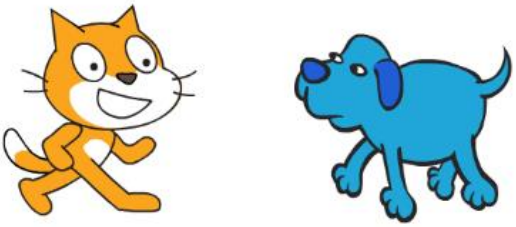
Bùi Việt Hà

Tự học lập trình

We support approaches to coding that engage young people in thinking creatively, reasoning systematically, and working collaboratively -- essential skills for everyone in today's society.

Chúng tôi hỗ trợ một công cụ lập trình mới giúp trẻ suy nghĩ hợp lý hơn, hệ thống hơn, sáng tạo hơn, làm việc nhóm và rèn luyện các kỹ năng cần thiết trong xã hội hôm nay.

Start Scratch Programming Yourself



Hà Nội 2017

Mục lục

Vì sao Scratch?

Trên tay các bạn là cuốn sách **Tự học lập trình Scratch**, một môi trường, ngôn ngữ lập trình "kéo thả" rất mới đối với Việt Nam. Vì sao mọi người cần học môi trường lập trình này? Vì sao Scratch lại thích hợp cho lứa tuổi thiếu nhi, thiếu niên và phù hợp cho việc đưa các kiến thức lập trình cho các bậc học này?

Môi trường và ngôn ngữ lập trình Scratch do nhóm nghiên cứu Lifelong Kindegarden Group thuộc đại học MIT (Massachusetts Institute of Technology) thiết lập đầu năm 2008. Ý tưởng ban đầu của nhóm chỉ là thiết lập một ngôn ngữ lập trình mới, đơn giản, chỉ dùng kéo thả, dành cho trẻ con để thiết lập trò chơi, phim hoạt hình, ứng dụng đơn giản, kích thích sự sáng tạo trong môi trường làm việc nhóm của trẻ.

Tuy nhiên Scratch chỉ thực sự bùng nổ từ năm 2014 khi một số quốc gia như Anh, Mỹ đã đổi mới đột phá chương trình giảng dạy môn Tin học trong nhà trường, đưa nội dung kiến thức Khoa học máy tính vào nhà trường ngay từ cấp Tiểu học. Một trong những đề nghị quan trọng nhất của các chương trình này là cần đưa các ngôn ngữ lập trình đơn giản, dạng kéo thả như Scratch vào giảng dạy trong nhà trường ngay từ Tiểu học. Việc điều chỉnh chương trình môn Tin học này đã kéo theo sự gia tăng bùng nổ của Scratch trên phạm vi toàn thế giới. Số lượng học sinh đăng ký tham gia trang Scratch tăng đột biến cả về số lượng và chất lượng. Thực tế đã chứng minh tính hấp dẫn của các môi trường lập trình kéo thả như Scratch, sự đam mê lập trình của trẻ nhỏ. Scratch vô cùng thích hợp cho trẻ lứa tuổi từ 6 đến 14, tức là các cấp Tiểu học, THCS của Việt Nam. Chính vì vậy trong Chương trình đổi mới giáo dục của Việt Nam sau 2018, Bộ Giáo dục & Đào tạo cũng đã quyết định đưa nội dung kiến thức Khoa học máy tính trong môn Tin học vào ngay từ cấp Tiểu học, và những ngôn ngữ lập trình kéo thả như Scratch sẽ là một lựa chọn tốt cho các nhà trường và học sinh.

Scratch là gì?

Tóm tắt một vài ý để trả lời cho câu hỏi: **vậy Scratch là gì?**

- Scratch là 1 môi trường lập trình ứng dụng đặc biệt, trong đó việc "viết" lệnh sẽ được thực hiện bằng thao tác "kéo thả".

- Đầu ra của Scratch hỗ trợ các công nghệ và ứng dụng mới nhất của CNTT-ICT, do vậy các ứng dụng của Scratch rất phong phú, hấp dẫn, nhất là trẻ nhỏ.

- Scratch có sự phát triển bùng nổ 2 năm trở lại đây. Đặc biệt là sau khi một số quốc gia có tiềm lực khoa học kỹ thuật mạnh trên thế giới đã quyết đinh đưa Scratch vào giảng dạy trong nhà trường cho học sinh từ cấp Tiểu học.

- Scratch hoàn toàn miễn phí và có thể chia sẻ rộng rãi trong cộng đồng.

- Scratch rất thích hợp để tạo ra các ứng dụng đồ họa, animation, bài học, bài giảng, mô phỏng kiến thức, trình diễn, sách điện tử, trò chơi, … rất phù hợp với nhà trường, giáo viên, học sinh.

- Scratch là môi trường tốt nhất để dạy học sinh làm quen với tư duy máy tính, khoa học máy tính ngay từ lứa tuổi tiểu học.

Nội dung, đối tượng cuốn sách

Cuốn sách sẽ bao quát tất cả các chủ đề chính của môi trường lập trình Scratch bao gồm: chuyển động, đồ họa, âm thanh, hội thoại, cảm biến, biến nhớ, xử lý số - xâu ký tự - mảng số, thủ tục và clone. Đối tượng của sách có thể là giáo viên tin học, giáo viên thường, học sinh tất cả các cấp từ Tiểu học, THCS, THPT.

Về định hướng nội dung của cuốn sách này sẽ là một trung dung giữa ứng dụng thuần túy thực tế và kiến thức hàn lâm của khoa học máy tính. Chúng tôi không đi quá sâu vào học thuật sẽ gây nhàm chán, khó hiểu với học sinh, nhưng cũng không sa đà quá nhiều vào các kỹ năng thiết kế trò chơi, phim hoạt hình, … Tuy nhiên trong cuốn sách này sẽ có vừa đủ 1 số kiến thức về thuật toán, dữ liệu, hệ đếm và cả 1 số kỹ năng và ý tưởng thiết kế trò chơi, phần mềm.

Nội dung sách sẽ được chia thành nhiều bài học nhỏ. Mỗi bài học đều có chung 1 cấu trúc nhất định. Người học có thể tự học hoặc học, thực hành dưới sự hướng dẫn của giáo viên.

Cấu trúc 1 bài học sẽ thống nhất sẽ bao gồm các phần sau.

Mục đích

Giới thiệu ngắn mục đích, yêu cầu cần đạt về kiến thức và năng lực của người học.

Bắt đầu

Phần mở đầu của mỗi bài học sẽ thường bắt đầu bằng những câu hỏi, đặt vấn đề liên quan đến bài học để người đọc suy nghĩ và chuẩn bị, trước khi bước vào phần chính thức. Mô hình của phần khởi động này chính là khơi dậy nguồn cảm hứng của người học, giúp người học sẽ luôn chủ động trong quá trình học tập và luyện tập.

Nội dung bài học

Nội dung chính của mỗi bài học sẽ bao gồm một dãy các hoạt động hoặc trải nghiệm dành cho người học. Người học có thể tự học, đọc hoặc làm theo hướng dẫn của giáo viên để thực hiện các hoạt động này. Các hoạt động sẽ dần dần dẫn dắt người học khám phá và từng bước nắm bắt kiến thức, dựa trên đó sẽ hình thành năng lực theo yêu cầu của bài học.

Câu hỏi và bài tập

Các bài tập, bài luyện trắc nghiệm hoặc lập trình mở rộng sẽ giúp người học củng cố kiến thức đã được học và rèn luyện kỹ năng lập trình, tư duy thuật toán và giải quyết vấn đề. Đa số các bài tập là đơn giản. Rất nhiều bài tập là các phát triển cụ thể của ý tưởng hoặc thiết kế đã có trong các hoạt động của bài học.

Mở rộng

Đây là phần thực sự có ý nghĩa vận dụng, mở rộng, tìm tòi trải nghiệm dành cho người học. Đa số các vấn đề, bài tập trong phần này là những bài khó, yêu cầu người học phải suy nghĩ, thử nghiệm, thực hành nhiều hơn. Nếu người học làm được tất cả các bài toán, vấn đề được đặt ra trong phần này thì chứng tỏ đã xuất sắc hoàn thành mục tiêu, mục đích của bài học.

Các bài tập hoặc ý tưởng của phần mở rộng sẽ đóng vai trò các bài tập lớn, bài tập nhóm và trải nghiệm thực hành của học sinh.

Vì sao học sinh cần học lập trình?

1. Cái nhìn lệch lạc hiện nay về việc dạy lập trình cho học sinh phổ thông

Từ khi có môn Tin học trong nhà trường phổ thông đến nay, chưa bao giờ việc dạy kiến thức và kỹ năng lập trình được coi trọng, thậm chí ngược lại. Chúng ta hãy cùng điểm qua các nguyên nhân của hiện tượng trên.

- Trong chương trình môn Tin học hiện nay trong nhà trường phổ thông, thời lượng và yêu cầu kiến thức, kỹ năng của phần lập trình là rất ít. Bản thân môn học Tin học không được coi là môn quan trọng và không có trong danh sách các môn thi tốt nghiệp hay thi vào các trường đại học, cao đẳng.

- Chương trình môn Tin học không được thiết kế một cách liên tục, liên thông từ cấp Tiểu học, THCS lên THPT. Điều này là hệ quả của việc trong suốt thời gian qua, môn Tin học chỉ chính thức và bắt buộc với cấp THPT, còn nó là tự chọn cho các cấp Tiểu học, THCS.

- Một nguyên nhân nữa cần nhắc đến là trình độ giáo viên tin học. Hầu như tất cả giáo viên môn Tin học hiện nay đều chưa được trang bị kiến thức đầy đủ về chuyên ngành Khoa học Máy tính (Computer Science), đặc biệt là kỹ năng lập trình, chưa đủ kiến thức về thuật toán để có thể truyền đạt và gây hứng thú khi dạy lập trình cho học sinh.

- Trong một bối cảnh như vậy nên dễ hiểu trong suốt thời gian qua, phần dạy lập trình trong khung môn tin học hầu như không được coi trọng. Học sinh không thích học, giáo viên không muốn dạy. Phần kiến thức quan trọng này gần như bị bỏ qua hoặc coi nhẹ trong quá trình dạy môn học này.

2. Lập trình sẽ giúp các em những kiến thức gì, năng lực gì, kỹ năng gì?

Hiện nay Bộ GD&ĐT đang thiết kế lại Chương trình đổi mới giáo dục phổ thông, trong đó có môn Tin học. Môn Tin học sẽ là môn học bắt buộc trong nhà trường suốt từ lớp 1 đến lớp 12. Trong cấu trúc của chương trình môn Tin học mới, mảng kiến thức Khoa học Máy tính (CS - Computer Science) sẽ đóng vai trò trung tâm, và trong kiến thức CS này, lập trình sẽ có vai trò rất lớn. Chúng ta cùng tìm hiểu nhanh vai trò và ý nghĩa của việc học lập trình trong phạm vi môn Tin học nói riêng và trong toàn bộ chương trình giáo dục phổ thông nói chung.

Học lập trình sẽ giúp các em trải nghiệm và rèn luyện các kỹ năng, kiến thức, năng lực tư duy sau:

- Tư duy logic chặt chẽ (tư duy thuật toán).

- Tư duy thuật toán, hay biết sử dụng tư duy máy tính để giải quyết vấn đề.

- Kỹ năng làm việc nhóm, trao đổi, hợp tác để giải quyết vấn đề.

- Năng lực sáng tạo, làm việc độc lập, kiên trì theo đuổi mục đích.

- Kỹ năng tính toán, logic, chặt chẽ của toán học, kỹ thuật, công nghệ.

- Năng lực và khả năng tự học, tự nghiên cứu, sáng tạo trong công việc.

- Năng lực thẩm mỹ, nghệ thuật thông qua việc thiết kế giao diện của các sản phẩm.

Chúng ta đều đã biết, Tin học, và cái lõi của nó là Khoa học máy tính, có ứng dụng lớn nhất và mạnh mẽ nhất hiện nay. Các ứng dụng của CNTT len lỏi vào khắp hang cùng ngõ hẻm của cuộc sống, vào từng gia đình. Nhưng với Khoa học máy tính thì toàn bộ phần "thực nghiệm" của nội dung này đều từ việc lập trình.

3. Vì sao lập trình hiện nay trở nên hấp dẫn hơn, dễ học hơn, dễ dạy hơn

Trước đây khi sự phát triển của máy tính còn sơ khai thì công việc lập trình là 1 việc vô cùng nặng nhọc, khó khăn. Do vậy việc đưa nội dung lập trình vào trường phổ thông gặp nhiều khó khăn là dễ hiểu.

Tuy nhiên, các ứng dụng của CNTT ngày càng phát triển. Các công cụ lập trình, ngôn ngữ lập trình cũng phát triển rất nhanh, đáp ứng đa dạng các nhu cầu học và dạy.

Gần đây nhất đã xuất hiện các ngôn ngữ và phần mềm lập trình dành riêng cho học sinh từ lứa tuổi từ tiểu học, thậm chí từ mẫu giáo. Các môi trường lập trình này đều có 1 đặc điểm chung là không cần "lập trình" theo nghĩa cổ điển, có thể chỉ bằng các động tác kéo thả đơn giản trên màn hình để tạo ra chương trình.

Các công cụ lập trình mới này thực sự là 1 cuộc cách mạng trong việc học và dạy tin học. Giờ đây học sinh có thể rất dễ dàng để học 1 ngôn ngữ lập trình và thiết lập các chương trình, phần mềm máy tính hoàn chỉnh chỉ sau 1 thời gian học ngắn. Các ngôn ngữ lập trình mới này lại rất bắt mắt, đẹp, đơn giản, thực sự rất hấp dẫn đối với học sinh ngay từ nhỏ.

Scratch chính là một môi trường lập trình mới như vậy: rất hấp dẫn, khuyến khích sáng tạo, làm việc nhóm, và có đủ mọi đặc tính của một ngôn ngữ lập trình hoàn chỉnh.

GIỚI THIỆU NHANH NỘI DUNG SÁCH

Cuốn sách Tự học lập trình Scratch bao gồm 5 chương, 25 bài học và 2 phụ lục cuối sách. Phần cuối cùng là Index và Tài liệu tham khảo.

Chương 1. Làm quen với Scratch

Chương 1 chỉ bao gồm 2 bài đầu tiên giới thiệu và làm quen với khái niệm "tư duy máy tính" và làm quen với giao diện lập trình Scratch. Mục đích chính của chương này là:

- Giới thiệu cho học sinh biết và hiểu được điều quan trọng sau: máy tính "suy nghĩ" như thế nào? Con người đang điều khiển hoạt động của máy tính như thế nào? Thông qua việc này học sinh sẽ tiếp cận nhưng khái niệm đơn giản như lệnh, chương trình, thuật toán để giải quyết 1 vấn đề được đặt ra.

- Làm quen với giao diện và phần mềm Scratch. Học sinh sẽ biết được những thao tác đầu tiên với Scratch và hiểu thế nào là "lập trình" trên môi trường Scratch.

Các bài học cụ thể của chương 1.

Bài 1. Tư duy máy tính là gì.

Bài 2. Làm quen với Scratch.

Chương 2. Bắt đầu lập trình Scratch

Chương 2 bao gồm 6 bài học tập trung vào các kỹ năng và các lệnh lập trình cơ bản nhất của Scratch. Đó là 3 nhóm kỹ năng lập trình chính trong Scratch: **chuyển động, đồ họa** và **âm thanh**. Mỗi nhóm lại chia thành 2 bài học, 1 bài cơ bản và 1 bài nâng cao.

Chương này có thể coi là những bài học nhập môn lập trình Scratch. Chúng tôi tập trung vào 3 mảng lệnh chính, cơ bản nhất của Scratch là **chuyển động (moving)**, **đồ họa (pen)** và **âm thanh (sound)**. Cũng trong chương này sẽ giới thiệu hầu hết các cấu trúc lệnh điều khiển trong Scratch bao gồm các lệnh lặp (**repeat, repeat until, forever, wait until**), các lệnh điều khiển sự kiện. Có thể coi phần kiến thức được trình bày trong chương này là 1 thay thế rất tốt (với rất nhiều mở rộng) cho phần kiến thức lập trình LOGO hiện đang có trong chương trình Tin học Tiểu học lớp 4, 5. Cũng trong chương này sẽ trình bày các thao tác bổ sung, tạo mới nhân vật, nền sân khấu và giới thiệu khả năng lập trình song song của Scratch.

Cấu trúc các bài học được thiết kế theo mô hình xoắn ốc qua các mức từ đơn giản (mức 1) và nâng cao (mức 2).

Danh sách các bài học của chương 2 bao gồm:

Bài 3. Chuyển động 1.

Bài 4. Vẽ hình 1.

Bài 5. Âm thanh 1.

Bài 6. Chuyển động 2.

Bài 7. Vẽ hình 2.

Bài 8. Âm thanh 2.

Chương 3. Tìm hiểu sâu hơn Scratch

Chương 3 có chức năng đi sâu hơn vào những kỹ năng lập trình đặc trưng và chuyên biệt của Scratch.

- Học sinh sẽ bắt đầu làm quen với khái niệm biến nhớ, cách khởi tạo và vai trò của biến nhớ trong chương trình.

- 2 công nghệ cơ bản và sâu sắc nhất của Scratch được trình bày trong chương này là khái niệm thông điệp **truyền thông** và giới thiệu các lệnh **cảm biến**. Các kỹ năng này là cơ sở của việc tạo quan hệ giữa các nhân vật, giữa nhân vật và sân khấu, là các công cụ lõi để thiết kế các chương trình, hoạt cảnh animation, phần mềm hoàn chỉnh trên nền Scratch.

Như vậy chương 3 sẽ kết thúc phần học **Cơ bản** của Scratch và bắt đầu đi sâu vào phần **Nâng cao** và **Chuyên sâu** của môi trường lập trình này. Học hết chương này, học sinh sẽ có thể thiết lập các chương trình, phần mềm tương đối phức tạp, hoàn chỉnh có ý nghĩa thực tế.

Chương này sẽ bao gồm 3 bài học như sau:

Bài 9. Hội thoại.

Bài 10. Hội thoại và truyền thông.

Bài 11. Cảm biến.

Chương 4. Scratch nâng cao

Chương 4 có thể coi là phần trung tâm, dài nhất và quan trọng nhất của cuốn sách. Phần này sẽ cung cấp một số kiến thức lõi về xử lý dữ liệu, thuật toán và giải quyết vấn đề. Phần kiến thức này được xếp vào mức nâng cao vì nó chỉ phù hợp với học sinh từ cấp THCS trở lên. Chương này sẽ bao gồm 10 bài học và được tách theo các nhóm nội dung riêng biệt: các thuật toán và bài toán xử lý số, xử lý xâu ký tự, làm việc với mảng số, thủ tục và cuối cùng là clone. Cũng trong chương này sẽ lần đầu tiên đưa khái niệm Sơ đồ hoạt động chương trình như 1 sơ đồ khối thiết kế hệ thống cho 1 chương trình hoàn chỉnh.

- Các thuật toán với số như các phép chia, chia có dư với số nguyên, khai triển số nguyên tố, chuyển đổi số thập phân sang nhị phân và ngược lại. Thuật toán tính ƯSCLN và BSCNN của 2 số tự nhiên.

- Các bài toán và thuật toán với xâu ký tự: tìm xâu con, hoán vị ngẫu nhiên xâu ký tự, tách và ghép xâu, tìm xâu đối xứng.

- Các bài toán và thuật toán với dãy số: tìm phần tử và giá trị Max, Min của dãy, hoán vị ngẫu nhiên dãy số, các thuật toán sắp xếp thứ tự dãy, tìm dãy con theo các tiêu chí khác nhau.

- Khái niệm thủ tục và thủ tục có tham số. Biến nhớ tổng thể và địa phương. Bước đầu học sinh được làm quen với thủ tục đệ qui.

- Giới thiệu mô hình và kỹ thuật lập trình Clone. Clone là mô hình mô phỏng lập trình đối tượng rất đặc sắc của Scratch. Sử dụng thành thạo kỹ thuật này, học sinh sau này sẽ dễ dàng tiếp cận với các ngôn ngữ lập trình bậc cao.

Danh sách các bài học trong chương này:

Bài 12. Xử lý số 1.

Bài 13. Xử ký số 2.

Bài 14. Xử lý xâu ký tự 1.

Bài 15. Xử lý xâu ký tự 2.

Bài 16. Làm việc với List 1.

Bài 17. Làm việc với List 2.

Bài 18. Thủ tục 1.

Bài 19. Thủ tục 2.

Bài 20. Clone 1. Phân thân của nhân vật.

Bài 21. Clone 2. Thuộc tính của Clone.

Clone (phân thân) là một tính chất công nghệ đặc biệt của Scratch. **Clone** được dùng trong rất nhiều bài toán lập trình Scratch. Là một trong những kiến thức khó nhất và sâu sắc nhất của Scratch, **clone** là hình ảnh mô tả rõ nhất những khái niệm lập trình hướng đối tượng mà bình thường rất khó truyền đạt cho học sinh. Khi các em hiểu rõ được vai trò và ý nghĩa của **clone** thì sau này khi học sang các ngôn ngữ lập trình bậc cao khác, những khái niệm như thừa kế, đa hình sẽ trở nên rất dễ hiểu.

Chương 5. Thiết kế trò chơi và phần mềm giáo dục

Chương 5 là chương đặc thù và khó nhất của cuốn sách. Trong chương này chúng tôi trình bày các ý tưởng, một số kỹ thuật chính và hướng dẫn chi tiết cách thiết kế một số chương trình, trò chơi, phần mềm giáo dục cụ thể. Đây có lẽ là nội dung dạy thiết kế phần mềm chi tiết đầu tiên của Việt Nam. Cái hay, cái đặc biệt của Scratch là chỉ cần học sau 1 thời gian ngắn, mỗi học sinh đều có thể tự thiết kế cho mình các trò chơi, phần mềm hoàn chỉnh theo đúng nghĩa đen và nghĩa bóng của từ này. Tóm tắt nội dung của chương 5.

- Bài đầu tiên của chương sẽ trình bày, hướng dẫn 1 số kỹ thuật thường gặp trên thực tế khi thiết kế phần mềm, trò chơi. Ví dụ như kỹ thuật tính điểm số, kỹ thuật đếm ngược, kỹ thuật thể hiện lượt chơi bằng hình ảnh các ngôi sao, kỹ thuật hiện số và chữ bằng hình ảnh. Tất cả các kỹ thuật này được mô tả chi tiết thông qua 1 trò chơi hoàn chỉnh, trò chơi **đánh bóng**.

- Bài tiếp theo sẽ trình bày chi tiết quá trình thiết kế từ ý tưởng, thiết lập nhân vật, thiết lập sơ đồ hoạt động của 1 số phần mềm, trò chơi tiêu biểu. Các trò chơi được thiết kế trong bài học này ít nhiều các em đã biết, nhưng lần đầu tiên được cùng tham gia thiết kế chi tiết, do vậy sẽ rất hấp dẫn đối với người học. Đặc biệt có những phần mềm rất nổi tiếng như Flappy Bird, Bắn súng, Xem bảo tàng, Trình diễn múa hát, Vẽ tự do, Hiệu ứng chim bay sẽ được mô tả chi tiết trong bài học này.

- 2 bài học sau cùng là phần chốt, khó nhất và phức tạp nhất của toàn bộ cuốn sách. Trong 2 bài học này, học sinh sẽ được cùng học và thiết kế các phần mềm giáo dục cụ thể nhưng khá phức tạp và cùng được tham gia thiết kế từ đầu đến cuối hoàn chỉnh. Bài 24 bao gồm 5 trò chơi, phần mềm liên quan đến số, bài 25 bao gồm 5 trò chơi, phần mềm liên quan đến chữ. Hầu hết các trò chơi này đều khá nổi tiếng trên thực tế và bây giờ học sinh có điều kiện cùng thiết kế từ đầu.

Danh sách các bài học có trong chương này.

Bài 22. Một số kỹ thuật thiết kế Games.

Bài 23. Thiết kế một số phần mềm giáo dục.

Bài 24. Các trò chơi với số.

Bài 25. Các trò chơi với chữ.

Các phụ lục

Phụ lục 1. Vẽ đồ họa trong Scratch.

Phụ lục mô tả chi tiết giao diện màn hình đồ họa của Scratch. Chức năng đồ họa này của Scratch cho phép người dùng tự thiết kế, vẽ các hình ảnh nhân vật hay nền sân khấu đơn giản.

Phụ lục 2. Các tập lệnh Scratch.

Phụ lục này liệt kê và mô tả ý nghĩa của tất cả các lệnh hiện có của bản Scratch 2 hiện tại.

Index. Tài liệu tham khảo

Phần Index có hơn 100 đầu mục. Cuối sách có 15 tài liệu tham khảo.

CHƯƠNG 1: LÀM QUEN VỚI SCRATCH

Chương 1 bao gồm 2 bài đầu tiên giới thiệu và làm quen với khái niệm "tư duy máy tính" và làm quen với giao diện lập trình Scratch. Mục đích chính của chương này là:

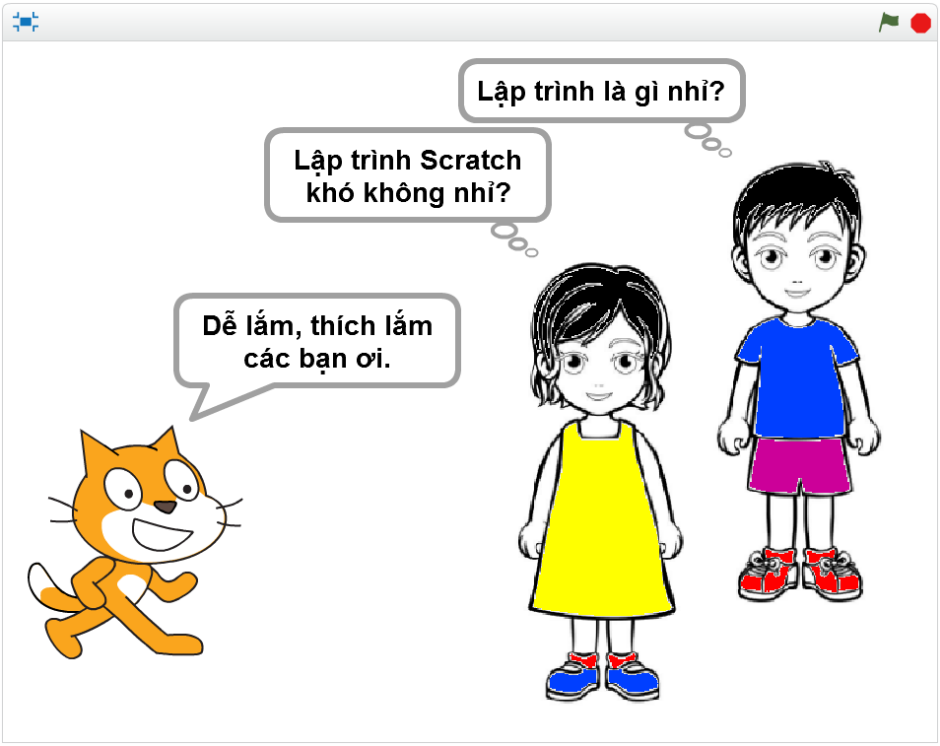
- Giới thiệu cho học sinh biết và hiểu được điều quan trọng sau: máy tính "suy nghĩ" như thế nào? Con người đang điều khiển hoạt động của máy tính như thế nào? Thông qua việc này học sinh sẽ tiếp cận nhưng khái niệm đơn giản như lệnh, chương trình, thuật toán để giải quyết 1 vấn đề được đặt ra.

- Làm quen với giao diện và phần mềm Scratch. Học sinh sẽ biết được những thao tác đầu tiên với Scratch và hiểu thế nào là "lập trình" trên môi trường Scratch.

Các bài học cụ thể của chương 1.

Bài 1. Tư duy máy tính là gì.

Bài 2. Làm quen với Scratch.



Bài 1. Tư duy máy tính là gì

Mục đích

Học xong bài này, bạn có thể:

- Hiểu được: để giải quyết 1 vấn đề, máy tính sẽ "tư duy", "suy nghĩ" như thế nào.

- Biết được để giải quyết 1 bài toán trên máy tính cần được thực hiện như thế nào, thông qua các ví dụ minh họa cụ thể.

Bắt đầu

1. Hàng ngày chúng ta vẫn được nghe, xem, quan sát các chương trình máy tính được điều khiển bởi con người. Hãy kể ra 1 vài chương trình như vậy mà em biết?

Gợi ý:

- Dây chuyền tự động lắp ráp máy tính.

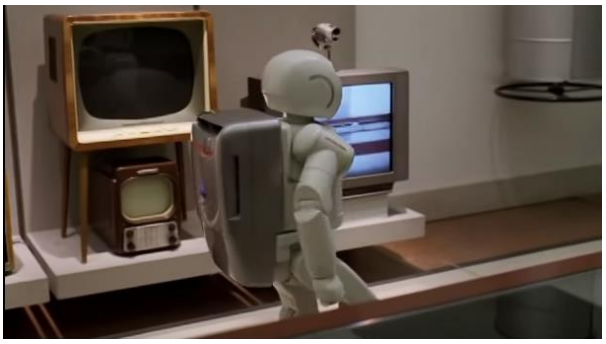
- Máy tính bỏ túi tự động tính toán các phép tính.

- Các trò chơi trên máy tính.

2. Em đã nghe được cụm từ "tuy duy máy tính" hay "máy tính nghĩ" ở đâu chưa? Hãy nêu ra 1 vài nguồn thông tin mà em biết. Tiếng Anh cụm từ này là "Computer thinking". Em hiểu thế nào về cụm từ này?

3. Quan sát Robot Asimo Nhật bản.

hoặc xem video:



https://youtu.be/NZngYDDDfW4

Em hãy trả lời các câu hỏi sau:

- Robot này có suy nghĩ giống người không?

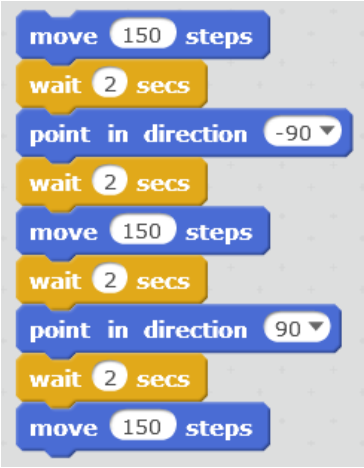
- Robot thực chất hoạt động như 1 máy tính, vậy máy tính có suy nghĩ không? Máy tính có tư duy không?

4. Em hãy quan sát kết quả của 1 chương trình máy tính đơn giản bằng cách mở và chạy chương trình Meo chay.sb2.

- Mô tả chuyển động của con mèo. Mèo đã thực hiện các công việc gì, theo thứ tự các bước như thế nào?

- So sánh điều em đã mô tả với hình sau:

Meo chay.sb2



- Em có rút ra được kết luận gì không?

5. Câu hỏi cho tất cả:

- Máy tính sẽ tư duy như thế nào để thực hiện các chương trình, công việc được lên kế hoạch trước?

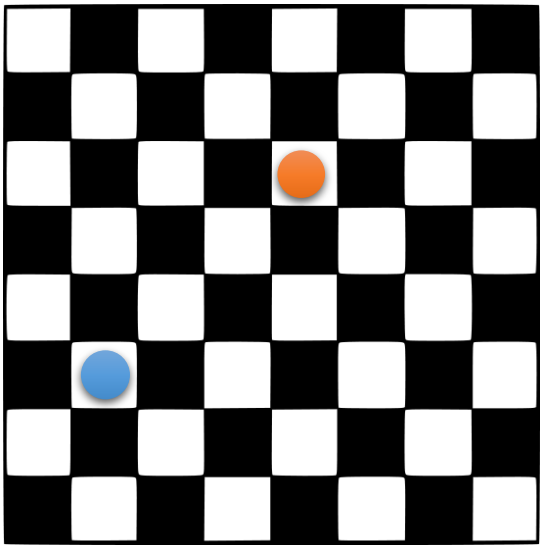
- Máy tính suy nghĩ và thực hiện công việc như thế nào?

- Vai trò của con người như thế nào đối với máy tính?

Nội dung bài học

1. Quân cờ trên bàn cờ vuông

Em hãy nhìn lên bàn cờ ô vuông sau. các quân cờ trên bàn cờ này mỗi lần chỉ đi được sang ô bên cạnh theo hàng ngang hoặc hàng dọc. Câu hỏi: quân cờ màu xanh cần phải đi bao nhiêu bước và đi như thế nào để đến được vị trí quân cờ màu đỏ?



Từ vị trí quân cờ màu xanh đến vị trí màu đỏ có nhiều cách đi khác nhau, nhưng cần chỉ ra các cách đi với ít bước nhất.

Trả lời: Đi bằng 6 bước và có thể đi theo nhiều cách để đến đích.

Trên thực tế tất cả các bài toán cần giải, các vấn đề cần giải quyết cũng phải được thực hiện theo từng bước.

Ghi nhớ: máy tính thực hiện công việc theo từng bước.

Em có thể tìm được rất nhiều các ví dụ khác về việc giải quyết vấn đề theo các bước trên thực tế. Hãy cùng xem các ví dụ sau, em cần liệt kê các bước cụ thể để thực hiện các công việc đó.

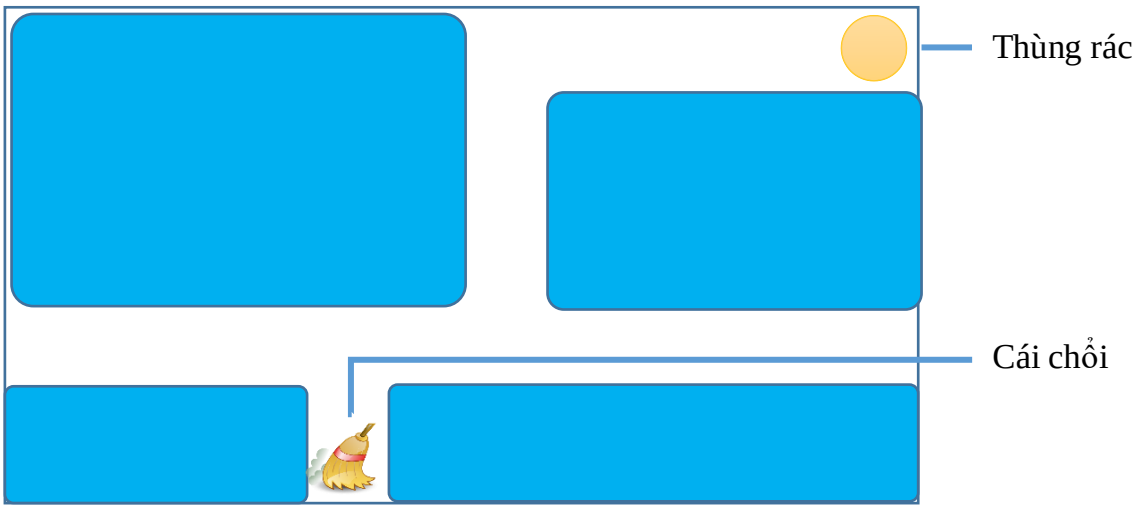
- Hàng ngày đi từ nhà đến trường, em phải đi qua các đường, phố nào, hãy liệt kê lần lượt các đường, phố đó.

- Các bước để thực hiện việc nấu cơm.

- Giải 1 bài toán theo các bước, ví dụ bài toán khai triển 1 số thành tích các số nguyên tố, bài toán giải phương trình bậc nhất, bài toán chứng minh một đẳng thức, bất đẳng thức.

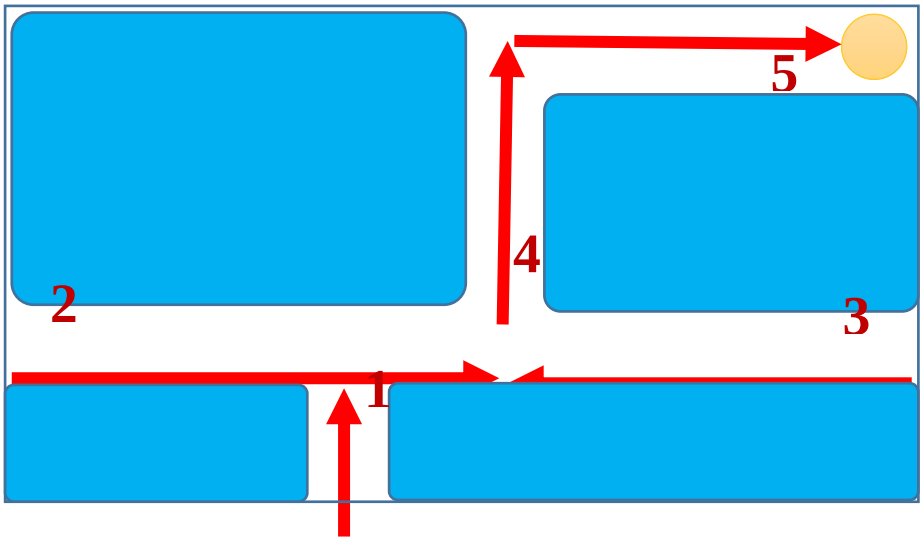
2. Quét nhà

Em quan sát căn phòng. Nhiệm vụ của em là cần quét rác và đưa rác đến góc nhà, vị trí có thùng rác. Em chỉ được dùng chổi quét theo các hướng ngang và dọc theo các lối đi trong phòng.



Hình ảnh dưới đây chỉ ra các công việc phải thực hiện đồng thời là thứ tự của chúng để hoàn thành công việc được giao trên.

Sơ đồ thực hiện là: 1 - 2 - 3 - 4 - 5.



Các em quan sát và trả lời các câu hỏi sau:

- Nếu thay đổi thứ tự các bước là: 3 - 1 - 2 - 4 - 5 thì công việc có hoàn thành không?

- Nếu chỉ thay đổi bước 1-2 thành: 2 - 1 - 3 - 4 - 5 thì kết quả công việc ra sao?

- Các bước 4, 5 có thể đổi chỗ cho nhau được không?

Kết luận

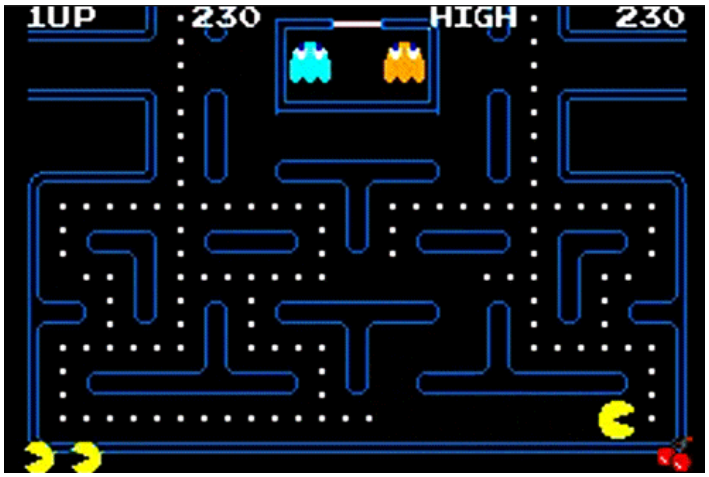
- Thông thường người ta thường giải quyết các bài toán thực tế thông qua các bước.

- Thứ tự các bước là quan trọng để hoàn thành nhiệm vụ.

3. Quan sát một chương trình máy tính mô phỏng giải quyết vấn đề

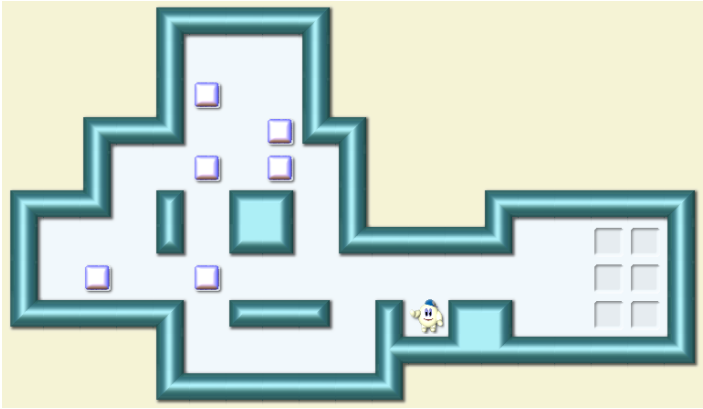
Có rất nhiều phần mềm hoặc trò chơi mô phỏng giải quyết vấn đề thông qua các bước.

- Trò chơi Pacman nổi tiếng một thời.



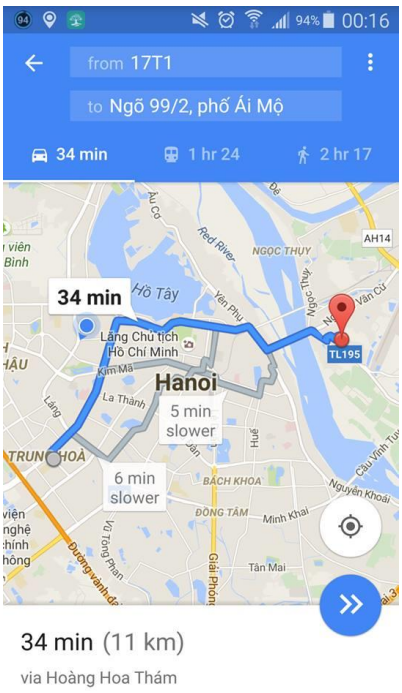
Trong trò chơi này, Pacman là người đào vàng sẽ có nhiệm vụ tìm vàng dọc theo các con đường trong 1 mê cung. Tuy nhiên trên đường đi thường xuất hiện các con quỉ sẽ tìm cách giết hại người đào vàng. Người đào vàng phải có nhiệm vụ tránh các con quỉ đồng thời đào được càng nhiều vàng càng tốt.

- Phần mềm trò chơi xếp đồ Socoban



Một bác công nhân có nhiệm vụ chuyển các thùng trong vườn đến vị trí đã được đặt trước. Mỗi lần bác công nhân chỉ có thể đẩy được 1 thùng đi 1 bước. Đường đi trong vườn rất khó vì có nhiều bức tường và nhiều thùng. Bác công nhân phải tìm ra được 1 cách dịch chuyển tối ưu nhất để hoàn thành nhiệm vụ.

- Tìm đường trên điện thoại



Bài toán tìm đường đi (tối ưu) giữa hai vị trí trong thành phố. Cho trước hai vị trí Đầu và Cuối, máy tính sẽ phải dựa trên bản đồ giao thông thành phố để tìm ra được 1 cách đi với khoảng cách hoặc thời gian tối ưu nhất.

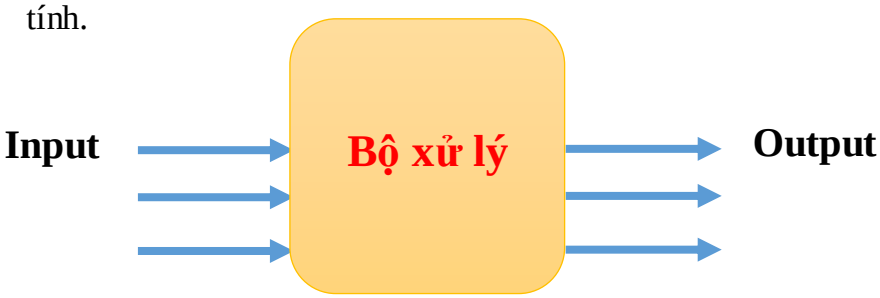
Trên màn hình máy tính sẽ hiện đường đi tối ưu đó với các thông tin có liên quan.

Các em có thể tìm hiểu và đưa ra nhiều ví dụ khác nữa.

Theo em có những điểm gì chung với các ví dụ trên?

4. Máy tính "nghĩ" như thế nào?

Em hãy quan sát sơ đồ sau hay được mô phỏng cho các công việc xử lý trên máy tính.



Input (đầu vào):

Đầu vào được hiểu là:

- Bài toán hay vấn đề cần giải quyết.

- Các thông số, dữ liệu ban đầu của bài toán.

Bộ xử lý:

Là chương trình máy tính, phần mềm hoặc thiết bị vi xử lý bất kỳ có chức năng tiếp nhận đầu vào, sau đó tiến hành giải quyếy bài toán.

Output (đầu ra):

Kết quả của bài toán, vấn đề được thể hiện ở đầu ra.

Ví dụ bài toán tìm đường đi tối ưu.

Input (đầu vào)	Xử lý	Output (đầu ra)
- Vị trí 2 điểm Đầu và Cuối. - Bản đồ đường giao thông thành phố. - Các qui định khác về giao thông, các tuyến đường 1 chiều, ….	Thực hiện tìm đường đi tối ưu từ vị trí Đầu đến vị trí Cuối. Máy tính thực hiện bài toán theo 1 qui trình đã được con người xác định (lập trình) trước theo các bước chặt chẽ, tối ưu, bảo đảm tìm ra được kết quả.	Thể hiện trên màn hình đường đi tối ưu này.

Máy tính **tư duy** như thế nào?

Như vậy "tư duy của máy tính" thực chất là việc con người cung cấp cho máy tính các dữ liệu đầu vào và cách xử lý vấn đề thông qua các chương trình, các bước, hay các lệnh một cách tuần tự, chặt chẽ, đảm bảo giải quyết được bài toán, vấn đề được đặt ra.

5. Con người sử dụng máy tính để giải quyết vấn đề gì và như thế nào?

Trên thực tế con người sử dụng máy tính để giải quyết các bài toán mà nếu không có máy tính sẽ rất khó giải quyết.

Qui trình con người sử dụng máy tính để giải quyết vấn đề thường theo các bước như sau:

- Xác định bài toán, vấn đề cần giải quyết (ví dụ: bài toán tìm đường đi trên bản đồ).
- Xác định các dữ liệu đầu vào ban đầu (vị trí 2 điểm Đầu và Cuối, bản đồ giao thông đường phố, …).
- Xây dựng chương trình để đưa vào máy tính xử lý (lập chương trình bao gồm các lệnh, đưa vào máy tính để xử lý).
- Máy tính chạy theo chương trình đã nạp và đưa ra kết quả, thể hiện là đầu ra trên màn hình máy tính.

Câu hỏi và bài tập

1. Em hãy liệt kê các bước để thực hiện 1 số công việc hàng ngày ở nhà em (như thổi cơm, quét nhà, giặt quần áo, ….).

2. Em hãy chỉ ra thêm các trò chơi và phần mềm mô phỏng giải quyết vấn đề.

3. Cách giải quyết bài toán sau là đúng hay sai?

Bài toán: cho trước 3 số a, b, c, tính tổng $a^2 + b^2 + c^2$ nếu biết rằng mỗi bước chỉ được phép thực hiện 1 phép toán 2 số.

Input: a, b, c.

Xử lý theo các bước sau:

B1. Tính $a^2 + b^2$

B2. Tính c^2

B3. Lấy kết quả B1 cộng với B2.

Output: in ra kết quả.

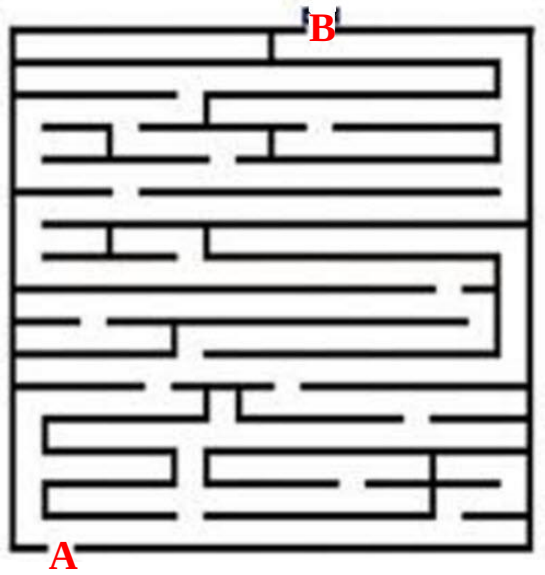
4. Cho trước số tự nhiên N (N>1). Mỗi bước em chỉ có thể thực hiện 1 trong các điều sau:

- Giảm N đi 1.

- Chia đôi N nếu N là chẵn.

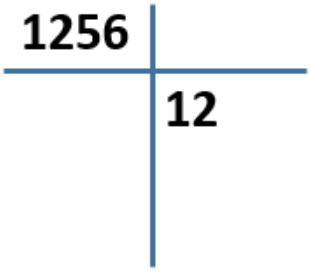
Giả sử cho N = 100. Hỏi em có thể thực hiện nhanh nhất là bao nhiêu bước để thu được số 1.

5. Em hãy tìm 1 đường đi từ vị trí A đến vị trí B.



6. Giả sử cần tính tổng a + b + c + d. Nhưng mỗi bước chỉ được phép tính phép cộng của 2 số. Hỏi em phải thực hiện bao nhiêu bước để tính được tổng trên.

7. Em hãy mô tả lại từng bước của phép chia hai số tự nhiên đã được học. Phép chia được thực hiện theo hàng dọc, ví dụ:



Chú ý: mỗi bước chỉ được phép mô tả 1 hành động, ví dụ:

- 1 phép tính +, -, x hay :

- Viết 1 số trên giấy.

8. Em hãy tìm các ví dụ minh họa cho các bước thực hiện của một bài toán cụ thể mà nếu thay đổi thứ tự các bước sẽ dẫn đến kết quả sai.

9. Qua 1 số ví dụ của bài học này, em có nghĩ rằng con người có thể tạo ra các máy biết suy nhĩ

10. Theo em, máy suy nghĩ khác gì so với người suy nghĩ?

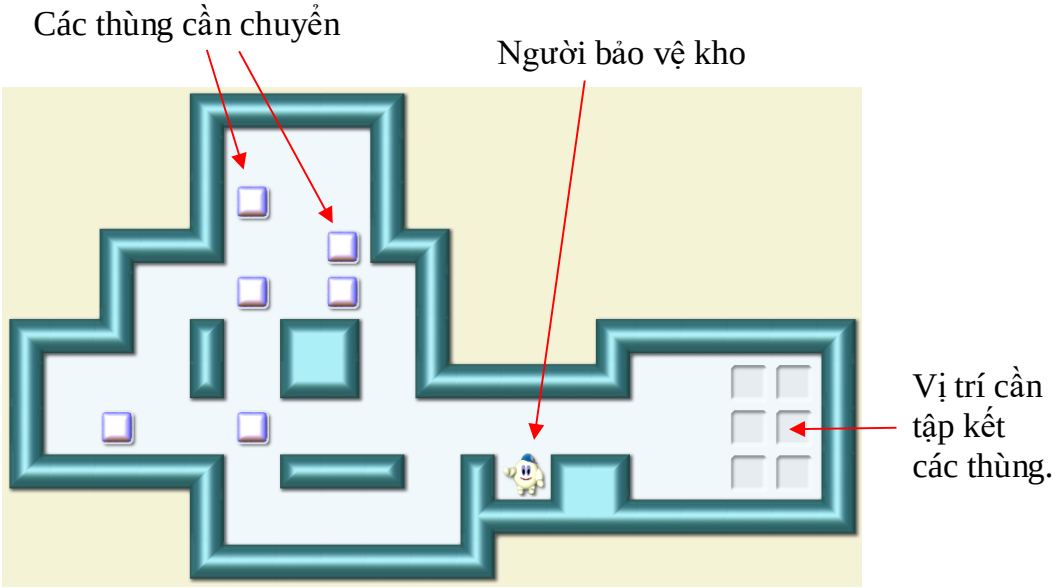
Mở rộng

Sử dụng cách mô tả mô hình xử lý vấn đề ở trên, em hãy viết và trình bày sơ đồ xử lý với 1 số bài toán sau (viết và mô tả Input, Output và Bộ xử lý).

1. Đầu vào là 3 số bất kỳ a, b, c, đầu ra là các số này nhưng đã được sắp xếp theo thứ tự tăng dần. Ví dụ đầu vào là 5, -1, 3 thì đầu ra của bài toán phải là -1, 3, 5. Biết rằng mỗi bước chỉ cho phép thực hiện đổi chỗ 2 số cạnh nhau hoặc có thể so sánh 2 số bất kỳ để kiểm tra xem số nào lớn hơn, số nào nhỏ hơn.

2. Bài toán tính tổng của 3 số bất kỳ cho trước a, b, c nếu biết rằng mỗi bước chỉ cho phép thực hiện: lấy tổng 2 số dương, đổi dấu của 1 số (từ âm thành dương và ngược lại), kiểm tra 2 số xem số nào lớn hơn, hoặc lấy 1 hiệu của 1 số dương lớn hơn trừ đi 1 số dương nhỏ hơn.

3. Giải bài toán chuyển đồ Socoban sau.



Người quản lý khi cần chuyển các thùng đồ (trong hình có 6 thùng) vào vị trí đúng ở góc bên phải. Hãy giúp người thợ thực hiện công việc này biết rằng người chỉ được phép đẩy 1 thùng đi theo từng bước.

Bài 2. Làm quen với Scratch

Mục đích

Học xong bài này, bạn sẽ biết:

- Làm quen và biết 1 môi trường ứng dụng mới: lập trình kéo thả.

- Lập trình tức là điều khiển máy tính hoạt động theo các bước, các lệnh tuần tự.

- Hiểu được qui trình hoạt động cơ bản của 1 chương trình trong môi trường Sratch.

Bắt đầu

Trong bài trước các em đã được làm quen với khái niệm "tư duy máy tính", tức là cách mà máy tính có thể "nghĩ" và "làm việc", "hành động" theo sự điều khiển của con người. Bây giờ chúng ta sẽ làm quen với một trong những hành động cụ thể đó của máy tính. Chúng ta sẽ được làm quen với một môi trường giao tiếp mới để con người có thể điều khiển máy tính có thể "suy nghĩ" và "làm việc", đó là môi trường Scratch.

1. Em hãy quan sát 1 chương trình chạy trên môi trường Scratch và đưa ra các nhận xét của mình. Chương trình được ghi trong tệp Dance.sb2.

Dance.sb2

Các gợi ý:

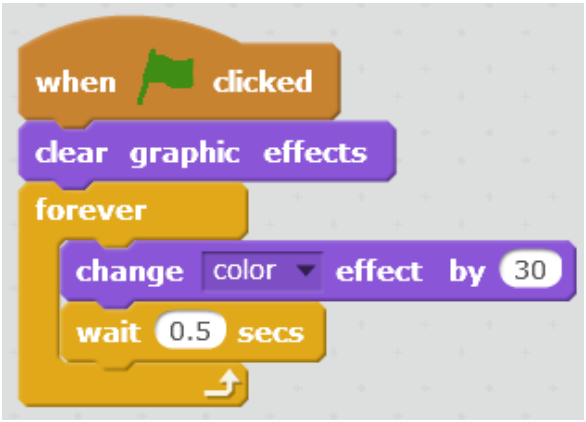
- Để chạy chương trình, em nháy lên hình lá cờ màu xanh phía trên cửa sổ chính. Em có nhận xét gì về hoạt động của cậu bé trên màn hình?

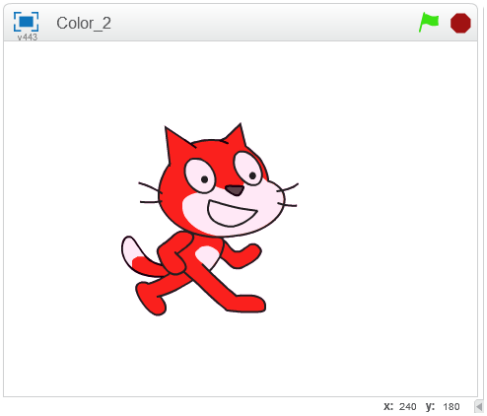
- Để dừng chương trình em bấm nút tròn màu đỏ phía trên (cạnh hình lá cờ).

- Em hãy chú đến các hoạt động của cậu bé, âm thanh trống và quan sát nền sân khấu.

2. Em mở tệp Color.sb2, chạy chương trình và viết lại nhận xét của em về hoạt động của nhân vật.

Color.sb2





Gợi ý:

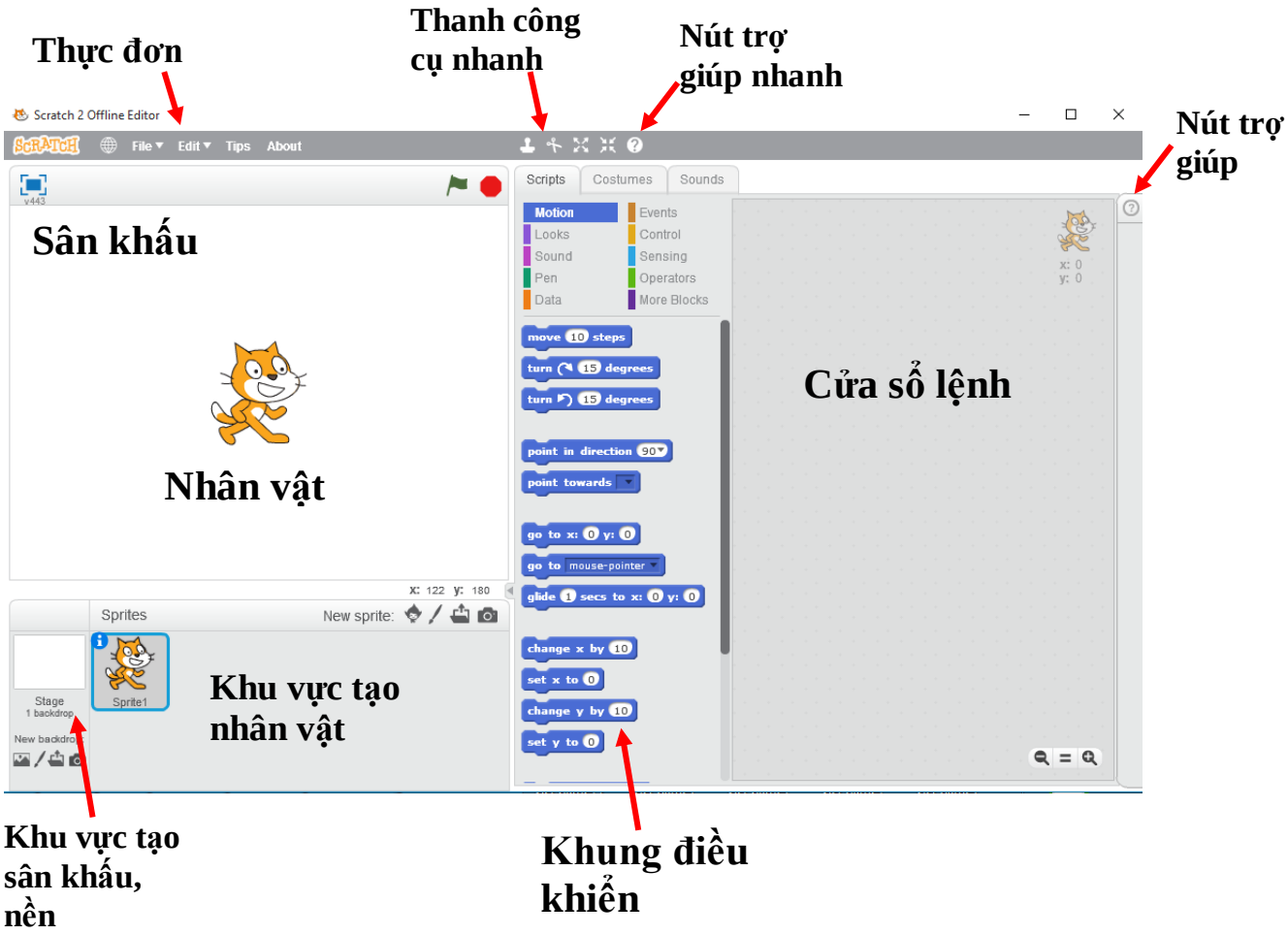
- Quan sát sự thay đổi màu sắc bên ngoài của con mèo.

- Em có thể giải thích vì sao mèo lại thay đổi màu của bộ lông của mình không?

Nội dung bài học

1. Làm quen với giao diện Scratch

Scratch là môi trường mà em sẽ được học trong cuốn sách này. Các em hãy quan sát giao diện chính của Scratch:



Chúng ta cùng tìm hiểu nhanh các vị trí quan trọng trong giao diện Scratch.

Sân khấu

Sân khấu là cửa sổ thể hiện chính của phần mềm. Khi phần mềm chạy chúng ta quan sát phần mềm thông qua sân khấu, tương tự như khi chúng ta xem biểu diễn ca nhạc, xem phim, xem Tivi.

Nhân vật

Nhân vật xuất hiện trên sân khấu, là đối tượng chính của các hoạt động. Có thể có nhiều nhân vật, đa dạng về kích thước và chủng loại. Khi lần đầu tiên chạy Scratch, nhân vật chính là 1 chú mèo xinh xắn.

Cửa sổ lệnh

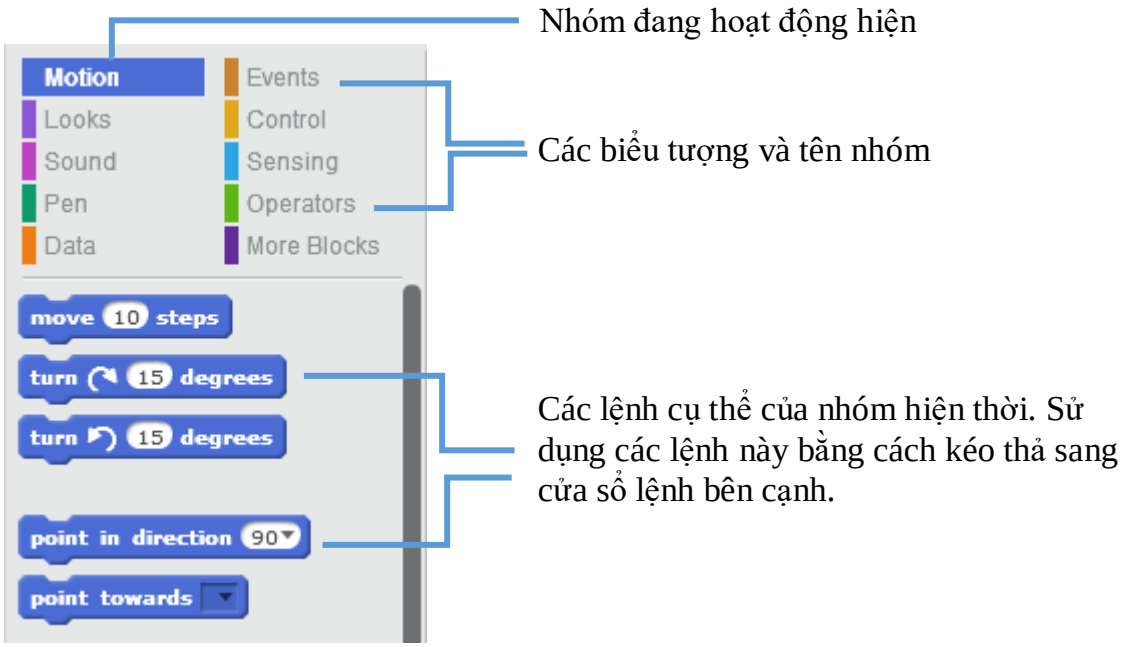
Cửa sổ lệnh chứa các "lệnh" để điều khiển hoạt động của nhân vật. Mỗi nhân vật có 1 cửa sổ lệnh riêng. Trong cửa sổ này không cần phải viết lệnh mà chỉ cần kéo thả các lệnh từ khung điều khiển bên cạnh sang. Vì vậy Scratch được gọi là môi trường lập trình kéo thả.

Khung điều khiển

Là nơi chứa các "công cụ" dùng để tạo ra chương trình. Tại khung này chứa các mẫu lệnh, người lập trình sử dụng các công cụ này để viết chương trình của mình.

Khung điều khiển này sẽ có 3 TAB thông tin: **Script** (Lệnh) , **Costume (Backdrop)** (Trang phục / Nền sân khấu) và **Sound** (Âm thanh).

Các lệnh trong Scratch được chia thành nhóm, mỗi nhóm bao gồm các lệnh có ý nghĩa gần tương tự nhau và thể hiện bằng 1 màu cụ thể. Có 10 nhóm các lệnh như vậy trong Scratch. Khi nháy lên 1 nhóm, khung phía dưới sẽ xuất hiện các lệnh của nhóm này.



Thực đơn

Thực đơn chứa các lệnh chính của Scratch.

Thanh công cụ nhanh

Thanh công cụ nhanh chứa một số lệnh làm việc nhanh với nhân vật và các lệnh.

Nút trợ giúp nhanh

Nút này có ý nghĩa như như: nháy chuột lên nút, sau đó nháy chuột lên 1 lệnh bất kỳ sẽ hiển thị nội dung mô tả của lệnh này.

Nút trợ giúp

Nháy nút này để hiện các trợ giúp nhanh của phần mềm.

Khu vực tạo nhân vật

Tại khu vực này em có thể thực hiện các thao tác như tạo thêm nhân vật, chỉnh sửa ngoại hình nhân vật (thay đổi trang phục), bổ sung âm thanh, ….

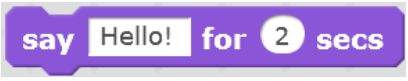
Khu vực tạo sân khấu, nền

Tại khu vực này em có thể thực hiện các thao tác với sân khấu như trang trí sân khấu, tạo thêm các cảnh sân khấu khác, tạo âm thanh nền cho sân khấu.

2. Điều khiển nhân vật bằng các lệnh

Các em hãy thực hiện các bước sau để hiểu cách điều khiển hoạt động của nhân vật con mèo.

(i) Nháy nút **Scripts**, nháy lên nhóm lệnh **Look** (màu tím)

(ii) Kéo thả lệnh 

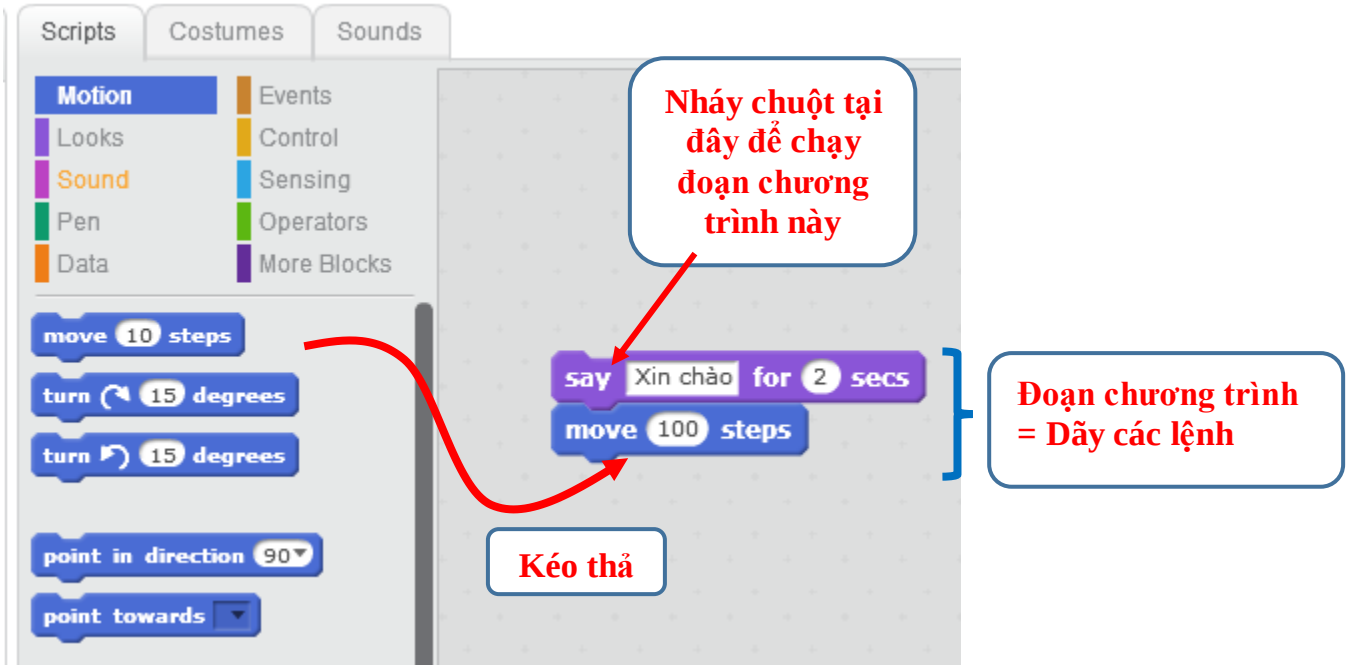
từ khung điều khiển phia dưới sang cửa sổ bên phải, sau đó sửa chữ "Hello!" thành "Xin chào".

(iii) Nháy lên nút nhóm **Motion** (màu xanh thẫm)

(iv) Kéo thả lệnh 

từ khung điều khiển phia dưới sang cửa sổ bên phải, sau đó sửa số 10 thành 100.

(v) Dùng chuột kéo 2 lệnh trên sát vào nhau như hình dưới đây, ta thu được 1 đoạn chương trình gồm 2 lệnh.



Bây giờ để chạy đoạn chương trình này em hãy nháy chuột lên vị trí bất kỳ của đoạn chương trình trên, và quan sát xem con mèo thay đổi như thế nào.

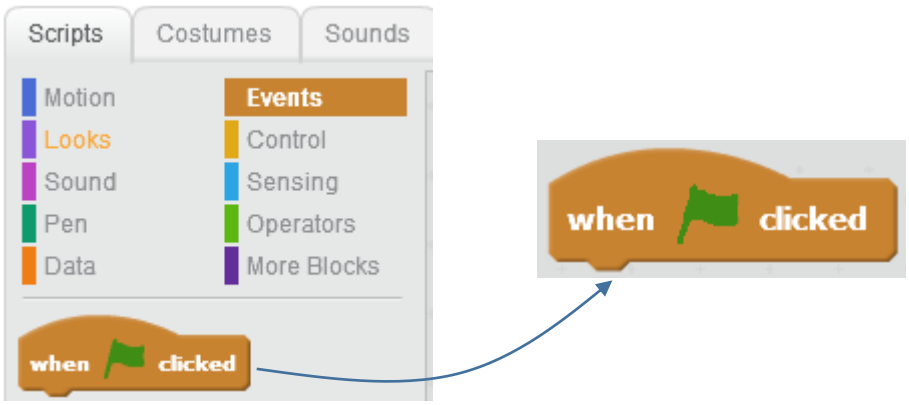
Các em sẽ thấy gì?

Nếu bây giờ em tách 2 lệnh trên rời ra và chạy thử em thấy có gì thay đổi?

3. Bắt đầu 1 chương trình đơn giản bằng sự kiện "Bắt đầu chương trình"

Bây giờ chúng ta sẽ cùng nhau hoàn thiện đoạn chương trình trên thành một chương trình hoàn chỉnh.

Chọn nút nhóm lệnh có chữ Events (sự kiện), kéo thả lệnh sau sang cửa sổ lệnh chính.



Sau đó em điều chỉnh để có 1 đoạn chương trình gồm 3 lệnh sau.

Lệnh điều khiển sự kiện khi nháy lên lá cờ xanh sẽ chạy đoạn chương trình dưới.

Nhân vật nói "Xin chào" trong 2 giây.

Nhân vật chạy về phía trước 100 bước.

Bây giờ chúng ta đã có 1 chương trình hoàn chỉnh trên Scratch. Em có thể phóng to cửa sổ chạy của chương trình và thực hiên lệnh chạy bằng cách nháy biểu tượng lá cờ xanh. Khi chạy chương trình, nhân vật của chúng ta sẽ nói "Xin chào" trong 2 giây, sau đó sẽ chuyển động về phía trước 100 bước.

Chương trình trên có thể viết lại dưới dạng các dòng chữ tiếng Việt cho dễ hiểu như sau:

```
Nếu nháy vào lá cờ xanh (nếu sự kiện này xảy ra)
     Thể hiện dòng chữ "Xin chào" trong 2 giây.
     Chạy về phía trước 100 bước.
```

2 lệnh này sẽ được thực hiện nếu sự kiện trên xảy ra.

Cách viết như trên thường được dùng trong tin học, khi thiết lập các vấn đề, bài toán và viết các bước thực hiện để giải quyết trên máy tính.

Một vài kết luận ban đầu.

- Các nhân vật có thể điều khiển bằng các lệnh trong cửa sổ lệnh.

- Lệnh có thể được kéo thả từ khung các mẫu lệnh, không phải viết từng lệnh trên màn hình.

- Nhóm các lệnh gắn liền nhau tạo thành 1 đoạn chương trình. Khi chạy các lệnh sẽ lần lượt chạy từ trên xuống dưới.

- Để hoàn thiện 1 chương trình hoàn chỉnh, cần đưa lệnh sự kiện 

lên đầu của đoạn chương trình.

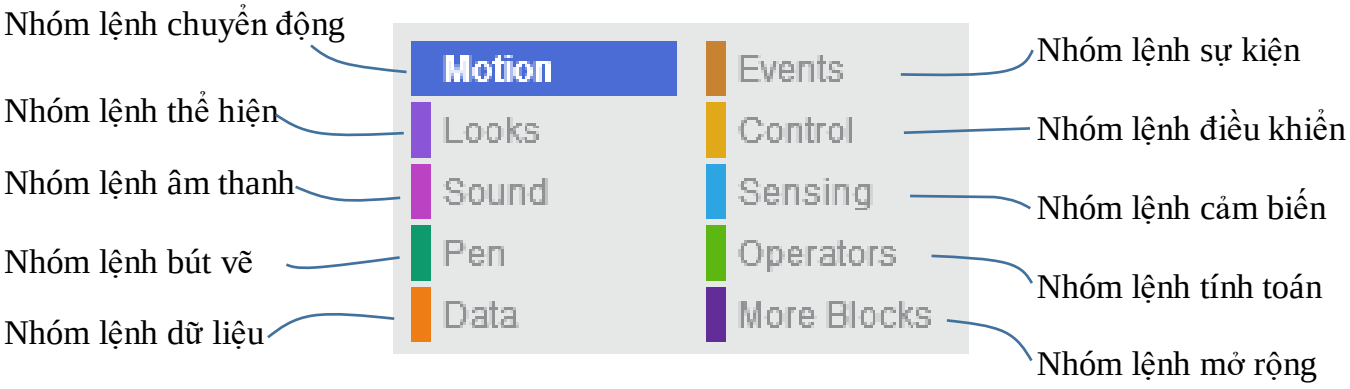
- Để chạy chương trình nháy lên nút có hình lá cờ xanh. Nháy nút tròn đỏ sẽ dừng chương trình đang chạy.

- Có thể phóng to cửa sổ chạy bằng cách nháy vào nút  góc trái trên màn hình.

4. Phân loại các nhóm lệnh điều khiển nhân vật

Em hãy quan sát vào khu vực khung điều khiển các lệnh của Scratch. Toàn bộ hệ thống lệnh được chia làm bao nhiêu nhóm? Mỗi nhóm có tên là gì và đặc trưng bởi màu sắc gì?

Em hãy nhìn vào sơ đồ sau để thấy toàn bộ các nhóm lệnh trong Scratch.



- Hãy liệt kê các nhóm lệnh của Scratch, màu sắc thể hiện của nhóm và nêu nhanh ý nghĩa các nhóm này.

- Với môi trường lập trình kéo thả Scratch chúng ta không cần học thuộc lòng và nhớ chi tiết từng lệnh. Chỉ cần hiểu ý nghĩa lệnh, cách sử dụng và kéo thả chúng sang cửa sổ lệnh là dùng được.

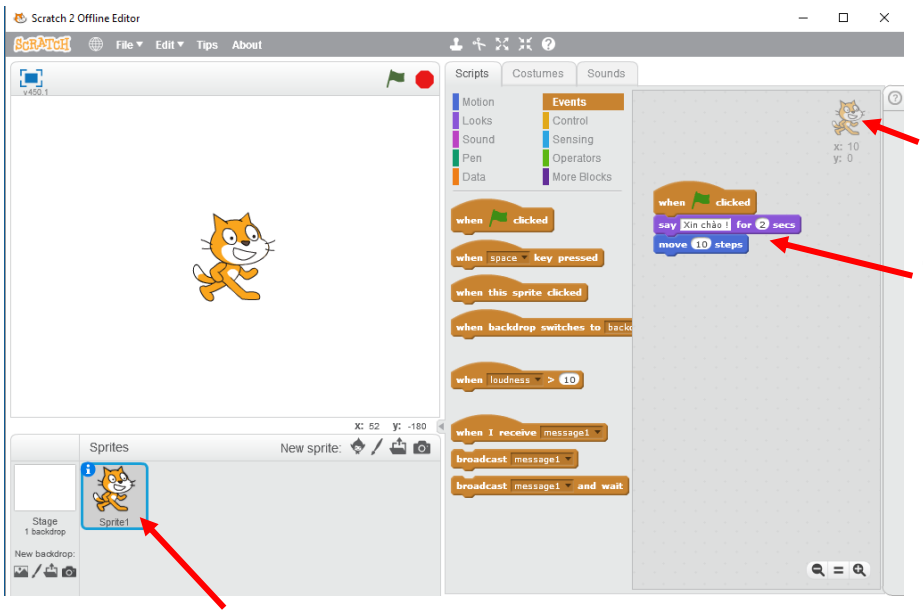
5. Cửa sổ lệnh của nhân vật và sân khấu

Trong phần trên em đã làm quen với cửa sổ lệnh ở bên phải màn hình. Em hãy quan sát góc phải trên của cửa sổ lệnh này.

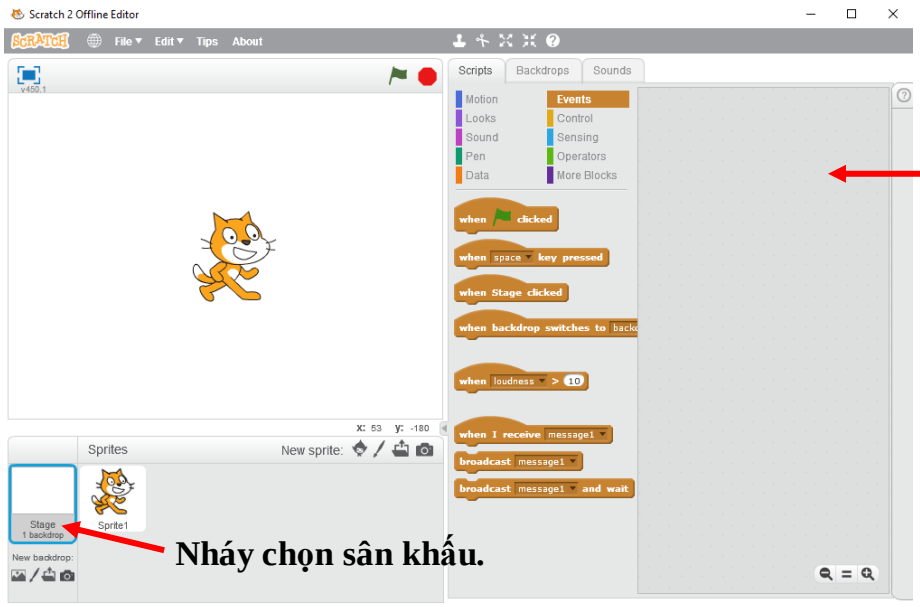
Em sẽ nhìn thấy hình ảnh của nhân vật (con mèo) tại góc này.

Như vậy đây là cửa sổ lệnh của mèo, hay nói cách khác, cửa sổ bao gồm các lệnh điều khiển mèo.

Mỗi nhân vật trong Scratch có 1 cửa sổ lệnh riêng biệt.



Bây giờ em hãy nháy chuột lên vị trí khung trắng chỉ sân khấu. Em sẽ thấy 1 cửa sổ lệnh mới, đây chính là cửa sổ lệnh của sân khấu. Các hệ lệnh của Scratch cũng có với sân khấu tương tự như đối với nhân vật, chỉ bớt đi một số lệnh và nhóm lệnh.



Như vậy:

> Mỗi nhân vật và sân khấu đều có 1 cửa sổ lệnh riêng. Nhóm các lệnh Scratch đối với sân khấu khác biệt với nhóm các lệnh với nhân vật.

Trong các bài tiếp theo em sẽ được làm quen với cả 2 nhóm lệnh này của Scratch.

Câu hỏi và bài tập

1. Trong ví dụ trên, nếu em đặt 2 lệnh tách ra như hình sau:

thì em có thể chạy 2 lệnh trên như 1 chương trình không?

Kết luận: các lệnh rời rạc nhau không tạo thành 1 chương trình hoàn chỉnh. Các lệnh cần kết nối với nhau.

2. Nút  nằm ở góc phải dưới của cửa sổ lệnh có ý nghĩa gì? Hãy thao tác và trả lời câu hỏi trên.

3. Hai nút nhỏ  này nằm trên thanh công cụ nhanh phía trên màn hình có tác dụng gì?

Để hiểu em hãy thực hiện thao tác sau:

- Nháy chuột lên 1 trong 2 nút trên.

- Sau đó em nháy chuột nhiều lần lên nhân vật.

- Em thấy điều gì xảy ra? Từ đó suy ra ý nghĩa của 2 nút trên.

4. Để tạo một chương trình mới, em thực hiện lệnh File / New.

Em hãy tạo 1 chương trình mới và tạo 1 đoạn chương trình đơn giản sau với nhân vật chính con mèo.

Chạy chương trình và quan sát xem con mèo thực hiện những hành động gì.

5. Để ghi lại chương trình đang làm việc, em thực hiện lệnh File / Save. Tệp chương trình Scratch cần có phần mở rộng là *.sb hoặc *.sb2.

6. Môi trường Scratch có thể chuyển đổi sang giao diện tiếng Việt. Em hãy thực hiện như sau:

Nháy nút  ở góc trên bên trái màn hình, sau đó tìm xuống dòng có ghi Tiếng Việt. Ví dụ nhóm các lệnh chính của Scratch trên giao diện tiếng Việt như sau.

Tên các nhóm lệnh Scratch bằng tiếng Việt.

Muốn đổi lại giao diện tiếng Anh, nháy vào nút  , sau đó chọn **English**.

7. Em khởi tạo 1 chương trình mới, vào cửa sổ lệnh của Mèo và sân khấu, nhập các dòng lệnh sau:

Cửa sổ nhân vật Mèo

Cửa sổ lệnh sân khấu

Em hãy chạy thử chương trình và nêu nhận xét của mình.

8. Tìm hiểu ý nghĩa nút lệnh nhỏ  tại dòng phía trên màn hình.

Giả sử em đã viết được 2 lệnh:

Bây giờ em hãy nháy lên nút  , sau đó em nháy lên vị trí của 2 lệnh trên. Em sẽ thấy gì? Hãy nêu ý nghĩa của nút lệnh  .

9. Tìm hiểu ý nghĩa nút lệnh nhỏ  tại dòng phía trên màn hình.

Giả sử em đã viết được 3 lệnh và ghép lại như hình sau:

Bây giờ em hãy nháy lên nút  , sau đó em nháy lên vị trí của 2 lệnh trên. Em sẽ thấy gì? Hãy nêu ý nghĩa của nút lệnh  .

10. Em hãy trả lời các câu hỏi sau:

- Dãy các câu lệnh của Scratch có cần kết dính lại với nhau không? Tại sao phải kết dính mà không tách rời?

- Có thể thay đổi thứ tự các lệnh trong một dãy lệnh được không? Vì sao?

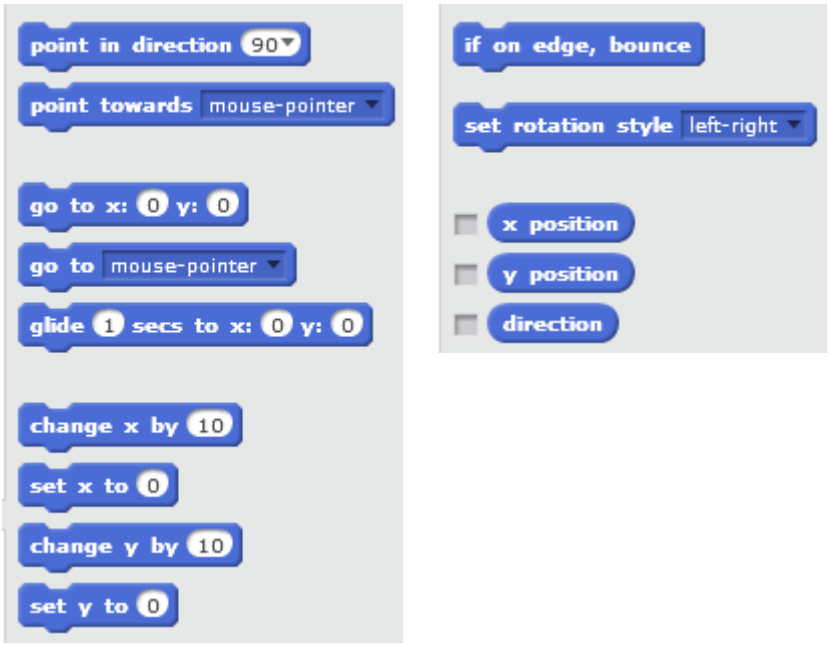
Mở rộng

1. Em hãy tìm hiểu các lệnh trong nhóm chuyển động (motion) và viết 1 chương trình hoàn chỉnh điều khiển nhân vật của em chuyển động trên màn hình.

a) Cho nhân vật con Mèo chạy 2 vòng xung quanh màn hình theo chiều kim đồng hồ.

b) Cho nhân vật con Mèo chạy 4 vòng xung quanh màn hình theo chiều ngược kim đồng hồ.

2. Em hãy quan sát và liệt kê tất cả các nhóm lệnh cùng các lệnh cụ thể của Scratch để xem trước. Ví dụ với nhóm lệnh Chuyển động (Motion), các lệnh sẽ được liệt kê như sau.



Em có nhận xét gì về nhóm lệnh này.

CHƯƠNG 2: BẮT ĐẦU LẬP TRÌNH SCRATCH

Chương 2 bao gồm 6 bài học tập trung vào các kỹ năng và các lệnh lập trình cơ bản nhất của Scratch. Đó là 3 nhóm kỹ năng lập trình chính trong Scratch: **chuyển động, đồ họa** và **âm thanh**. Mỗi nhóm lại chia thành 2 bài học, 1 bài cơ bản và 1 bài nâng cao.

Chương này có thể coi là những bài học nhập môn lập trình Scratch. Chúng tôi tập trung vào 3 mảng lệnh chính, cơ bản nhất của Scratch là **chuyển động (moving)**, **đồ họa (pen)** và **âm thanh (sound)**. Cũng trong chương này sẽ giới thiệu hầu hết các cấu trúc lệnh điều khiển trong Scratch bao gồm các lệnh lặp (**repeat, repeat until, forever, wait until**), các lệnh điều khiển sự kiện. Có thể coi phần kiến thức được trình bày trong chương này là 1 thay thế rất tốt (với rất nhiều mở rộng) cho phần kiến thức lập trình LOGO hiện đang có trong chương trình Tin học Tiểu học lớp 4, 5. Cũng trong chương này sẽ trình bày các thao tác bổ sung, tạo mới nhân vật, nền sân khấu và giới thiệu khả năng lập trình song song của Scratch.

Cấu trúc các bài học được thiết kế theo mô hình xoắn ốc qua các mức từ đơn giản (mức 1) và nâng cao (mức 2).

Danh sách các bài học của chương 2 bao gồm:

Bài 3. Chuyển động 1

Bài 4. Vẽ hình 1

Bài 5. Âm thanh 1

Bài 6. Chuyển động 2

Bài 7. Vẽ hình 2

Bài 8. Âm thanh 2

Bài 3. Chuyển động 1

Mục đích

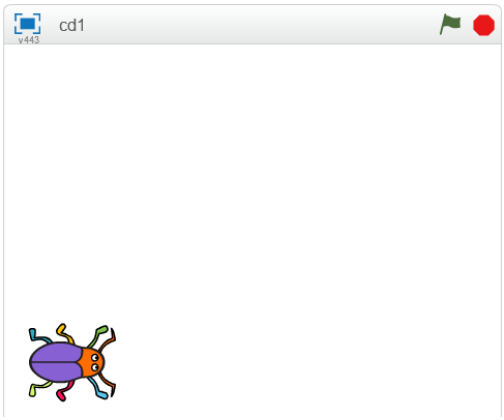
Học xong bài này, bạn có thể:

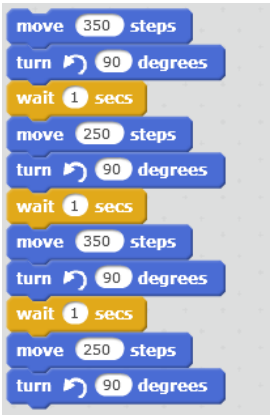
- Điều khiển nhân vật di chuyển trên màn hình (sân khấu) bằng dãy lệnh đơn giản.
- Thiết lập trang phục (costume, thể hiện bên ngoài, quần áo) cho nhân vật.
- Thay đổi nền sân khấu, màn hình.

Bắt đầu

Em hãy mở chương trình khoidong.sb2, quan sát và chạy chương trình.

khoidong.sb2





a) em hãy mô tả những gì xảy ra trên màn hình.

b) theo em các lệnh nào đã làm cho nhân vật (con cánh cam) chuyển động?

A. Lệnh  ?

B. Lệnh  ?

C. Lệnh  ?

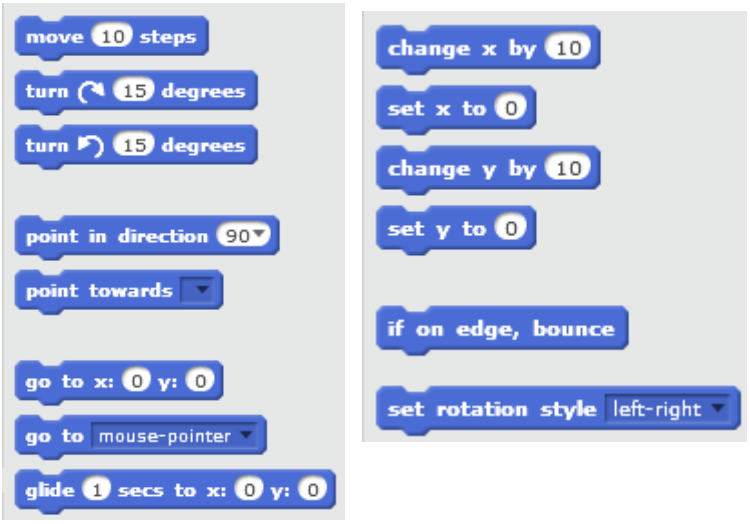
Em có nhận xét gì về màu sắc của các lệnh làm nhân vật chuyển động?

Nội dung bài học

1. Cùng quan sát nhóm lệnh Motion

Em hãy quan sát nhóm các lệnh trong nhóm chuyển động (motion) của Scratch.

Các lệnh trong nhóm chuyển động (motion).



Em có nhận xét gì về các lệnh này?

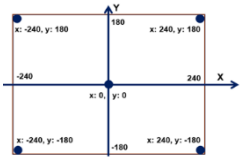
- Tất cả các lệnh trong nhóm Chuyển động đều có cùng màu xanh.

- Các lệnh này có cùng chức năng điều khiển hoạt động của các nhân vật trên màn hình.

2. Tọa độ nhân vật và kích thước sân khấu

Để có thể điều khiển chuyển động nhân vật dễ dàng chúng ta cần tìm hiểu thêm thông tin về sân khấu mà trên đó các nhân vật đóng vai chính.

Kích thước sân khấu



Sân khấu luôn có kích thước hình chữ nhật 480 x 360. Chiều ngang = 480 (điểm / pixel), chiều cao = 360 (điểm / pixel).

Tâm của sân khấu chính là tâm của hệ trục tọa độ màn hình.

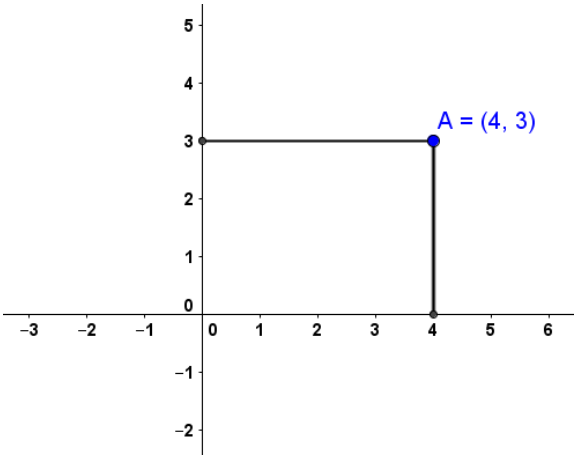
Mỗi vị trí trên sân khấu sẽ được đánh dấu bởi 2 số, 2 tọa độ (x, y). Ở đây:

x = khoảng cách từ vị trí này đến trục đứng (trục tung, trục Y); y = khoảng cách từ vị trí này đến trục ngang (trục hoành, trục X).

Cặp số (x, y) được gọi là tọa độ của vị trí, điểm đã cho.

Giá trị tọa độ x chạy từ -240 đến 240.

Giá trị tọa độ y chạy từ -180 đến 180.



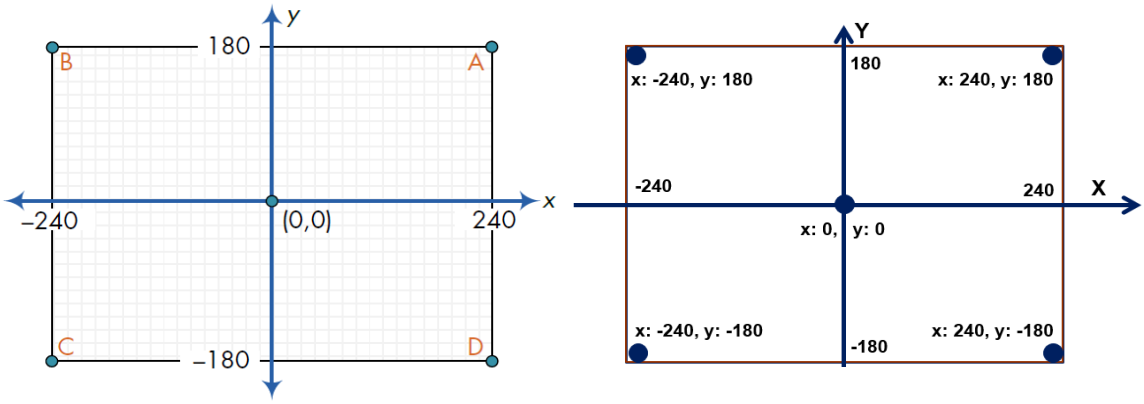
Tọa độ của điểm trên mặt phẳng có trục tọa độ và cách tính tọa độ.

Điểm A trên hình có tọa độ x=4 theo trục hoành (ngang) và y=3 theo trục tung (ngang).

Em hãy xác định trên sân khấu các vị trí có tọa độ sau:

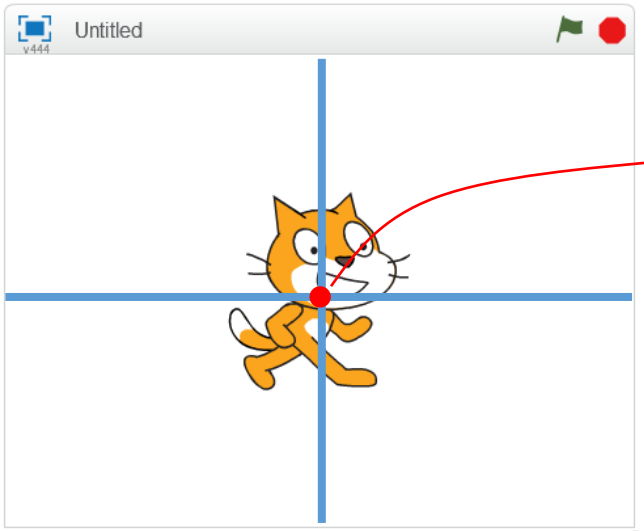
(0, 0); (10, -10); (100, 100).

Mô hình tọa độ và kích thước sân khấu của Scratch.



Tọa độ của nhân vật

Khi nhân vật chuyển động trên màn hình, chúng ta xác định nhân vật này bằng tọa độ của chúng. Tọa độ của nhân vật được tính là tọa độ của điểm chính tâm của nhân vật. Ví dụ điểm tính tọa độ của con mèo như hình dưới đây.



Vị trí này sẽ xác định tọa độ của nhân vật con mèo.

Em hãy kéo thả lần lượt các lệnh sau và kiểm tra xem nhân vật của chúng ta chuyển động như thế nào. Kết hợp quan sát và xem mô tả ở cột bên cạnh.

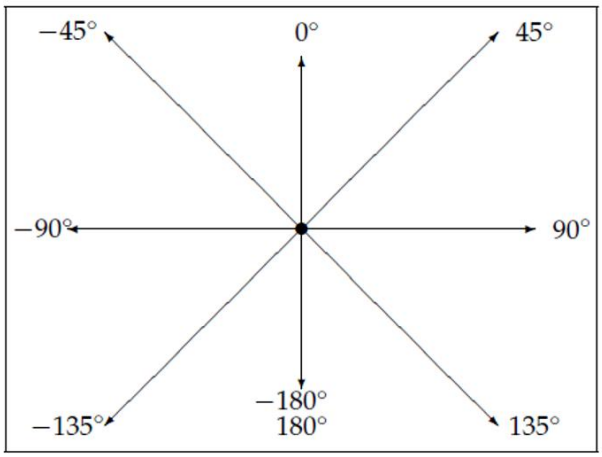
move 100 steps	Chuyển động nhanh về phía trước 100 bước.
go to x: 120 y: -50	Chuyển động nhanh (gần như tức thời) đến vị trí có tọa độ (120, -50).
glide 3 secs to x: 120 y: 120	Chuyển động trong thời gian 3 giây đến vị trí có tọa độ (120, -50).

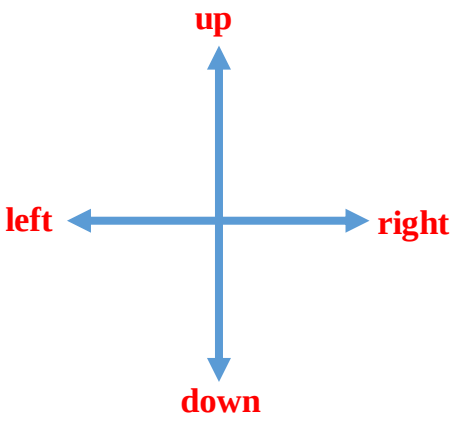
Biết được kích thước, tọa độ sân khấu chúng ta có thể điều khiển nhân vật chuyển động đến bất cứ vị trí nào chúng ta muốn.

3. Hướng chuyển động của nhân vật

Khi chuyển động bởi lệnh 

, nhân vật bao giờ cũng di chuyển theo 1 hướng nhất định. Trong Scratch, các hướng được qui định bởi 1 số đo góc, như chỉ ra theo sơ đồ sau.





Giá trị góc của hướng tính từ -180º đến 180º.

Lệnh 

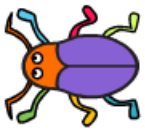
cho phép nhân vật quay theo chiều kim đồng hồ 1 góc cho trước (ví dụ 15 độ theo hình ảnh).

Lệnh 

cho phép nhân vật quay theo chiều ngược kim đồng hồ 1 góc cho trước (ví dụ 15 độ theo hình ảnh).

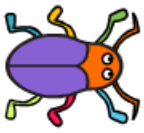
Khi 1 nhân vật mới được khởi tạo, nhân vật này luôn hướng theo hướng mặc định là hướng phải (right, 90º).

Nhân vật đang xoay theo hướng down (180º / -



Nhân vật đang xoay theo hướng left (-90º)

Mặc định nhân vật có hướng right (90º)



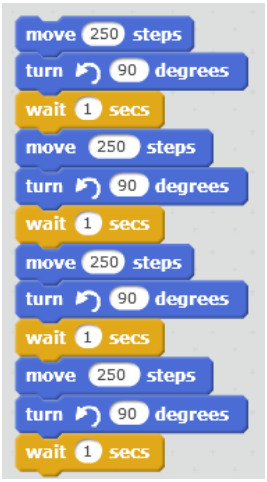
Nhân vật đang xoay theo hướng up (0º)

Hãy cho biết hướng các nhân vật này là hướng nào?

4. Hãy cho nhân vật chuyển động đơn giản trên sân khấu

Bây giờ các em hãy cùng thực hiện các chương trình đơn giản điều khiển các nhân vật chuyển động trên màn hình.

1. Thiết lập 2 chương trình sau và quan sát kết quả thực hiện.



Nhận xét:

- Kết quả thực hiện 2 chương trình này là như nhau (nhân vật chuyển động theo 1 hình vuông và quay về vị trí ban đầu), tuy nhiên với chương trình 1 rất khó quan sát vì tốc độ chuyển động nhanh quá, còn chương trình 2 thì dễ dàng quan sát sự chuyển động của nhân vật.

- Trong chương trình 2, lệnh 

(ý nghĩa tạm dừng thực hiện chương trình trong 1 giây) có ý nghĩa giúp chúng ta quan sát được sự chuyển động rõ ràng hơn. Kinh nghiệm này hay được sử dụng.

- Lệnh 

có màu vàng nhạt nằm trong nhóm lệnh Điều khiển (Control) mà các em sẽ được làm quen trong các bài học tiếp theo.

2. Bây giờ em sẽ cho nhân vật chính chuyển động theo chiều ngang trên sân khấu, ví dụ chạy từ trái sang phải và ngược lại.

Em hãy thiết lập chương trình có các lệnh sau:

Lệnh này có tác dụng cho nhân vật chính quay lại (xoay ngược kim đồng

Các em sẽ thấy gì?

Em sẽ thấy chú mèo khi quay ngược kim đồng hồ 180 độ sẽ bị lộn ngược!

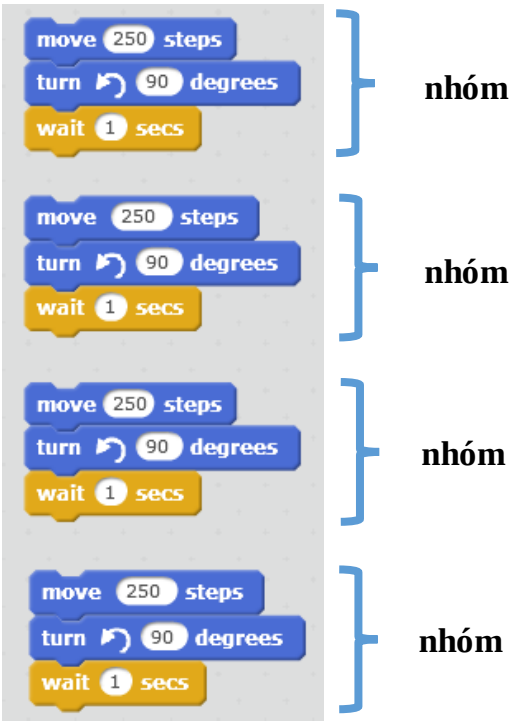
Để khắc phục lỗi này chúng ta sẽ dùng lệnh 

để điều chỉnh cách xoay người của nhân vật. Lệnh này có 3 lựa chọn bằng cách dùng chuột nháy chọn tại nút phía phải của lệnh. 3 lựa chọn quay là: chỉ **quay trái phải** (left-right); **không quay (don't rotate); và quay tròn (all around)**. Chúng ta nhìn thấy chú mèo bị lộn ngược là do lựa chọn quay tròn.

Bây giờ hãy viết lại đoạn chương trình trên:

5. Thêm lệnh lặp

Chúng ta sẽ quan sát kỹ hơn chương trình đã làm trong phần đầu của mục trên và chú ý đến 4 nhóm lệnh lặp lại giống nhau.



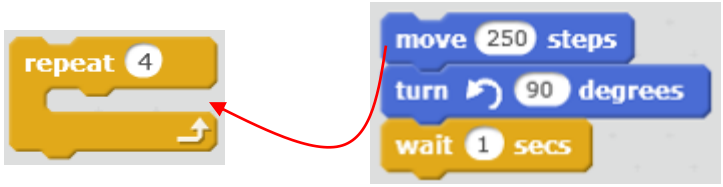
Để khỏi phải viết lại các nhóm lệnh giống nhau nhiều lần, trong Scratch (và trong mọi môi trường lập trình khác), người ta sử dụng các lệnh để điều khiển các cấu trúc lặp này.

Trong Scratch lệnh điều khiển này có dạng sau:

Kéo thả các nhóm lệnh muốn lặp vào vị trí này. Có thể thay đổi giá trị số 10 bằng 1 số tự nhiên.

Ý nghĩa của lệnh là cho phép thực hiện nhóm các lệnh nằm bên trong lệnh này có thể được thực hiện lặp lại 1 số lần. Số lần lặp có thể điều chỉnh trực tiếp trên lệnh.

Quay trở lại ví dụ trên, chúng ta có thể viết lại đoạn chương trình trên bằng 1 chương trình khác, ngắn gọn. Lệnh lặp được lấy từ nhóm lệnh Điều khiển.



1. Kéo thả nhóm lệnh này vào đây

2. Kết quả thu được, nhóm lệnh được thực hiện lặp 4 lần

Kỹ thuật lặp chương trình được sử dụng rất nhiều trong thực tế, trong tất cả các môi trường lập trình máy tính.

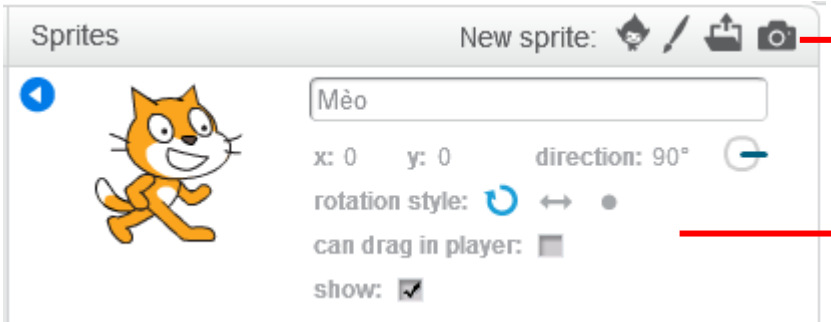
Em hãy viết lại đoạn chương trình lặp trên và chạy thử để kiểm tra kết quả.

6. Bổ sung nhân vật

Đối tượng điều khiển chính của 1 chương trình Scratch là nhân vật trên sân khấu. Chúng ta điều khiển nhân vật bằng chương trình được đưa ra (bằng cách kéo thả mẫu lệnh) trên Cửa sổ lệnh của nhân vật. Mỗi nhân vật có 1 cửa sổ lệnh (Script Window) của riêng mình. Điều đặc biệt thú vị là chúng ta có thể tạo ra nhiều nhân vật trên sân khấu.

1. Khu vực làm việc với nhân vật.

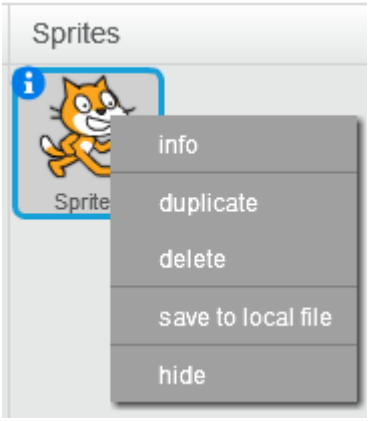
Khu vực phía dưới sân khấu chính là nơi tập trung thông tin của các nhân vật có trong chương trình. Nháy lên chữ i nhỏ sẽ xuất hiện các thông tin của nhân vật này. Khung thông tin nhân vật và các tính năng tương ứng như sau:



Các nút lệnh bổ sung thêm nhân vật.

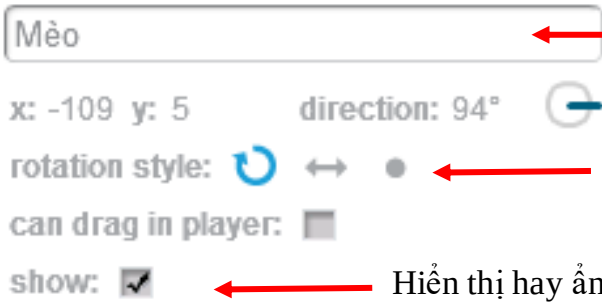
Cửa sổ thông tin của nhân vật. Tại đây có thể thay đổi, sửa trực tiếp thông tin của nhân vật.

Khi nháy chuột phải lên biểu tượng nhân vật sẽ có xuất hiện thêm 1 bảng chọn các lệnh nữa đối với nhân vật này.



Bảng chọn các lệnh với nhân vật khi nháy chuột phải lên nhân vật. Các chức năng từ trên xuống: **xem thông tin; tạo 1 bản sao; xóa; ghi hình ảnh ra file; ẩn không hiện trên màn hình.**

Tại cửa sổ thông tin của nhân vật chúng ta có thể thực hiện các thao tác sau.



Tên của nhân vật

Hướng xoay hiện

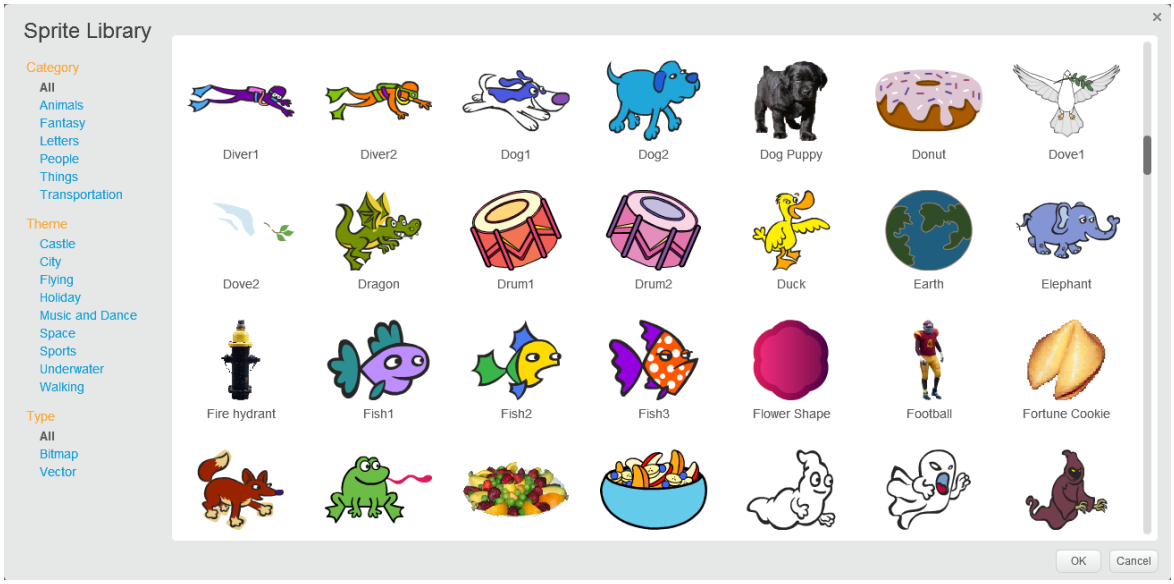
Lựa chọn cách xoay hướng nhân vật. quay tròn trái phải không quay

Hiển thị hay ẩn nhân vật.

2. Bổ sung thêm nhân vật từ thư viện dữ liệu

Để bổ sung thêm nhân vật từ 1 thư viện có sẵn, em hãy thực hiện các bước sau.

- Nháy nút  tại khung nhân vật.
- Xuất hiện cửa sổ các nhân vật trong thư viện có sẵn.



Kho thư viện này có rất nhiều hình ảnh người, con vật, đồ dùng, … có thể chọn làm nhân vật của chương trình.

- Nháy chuột chọn 1 hình và nháy nút OK để bổ sung hình này vào chương trình như 1 nhân vật mới.

Hình ảnh sau mô tả 1 chương trình có 4 nhân vật.

Danh sách các nhân vật hiện tại đây. Có thể bổ sung không hạn chế số lượng nhân vật cho mỗi chương trình Scratch.

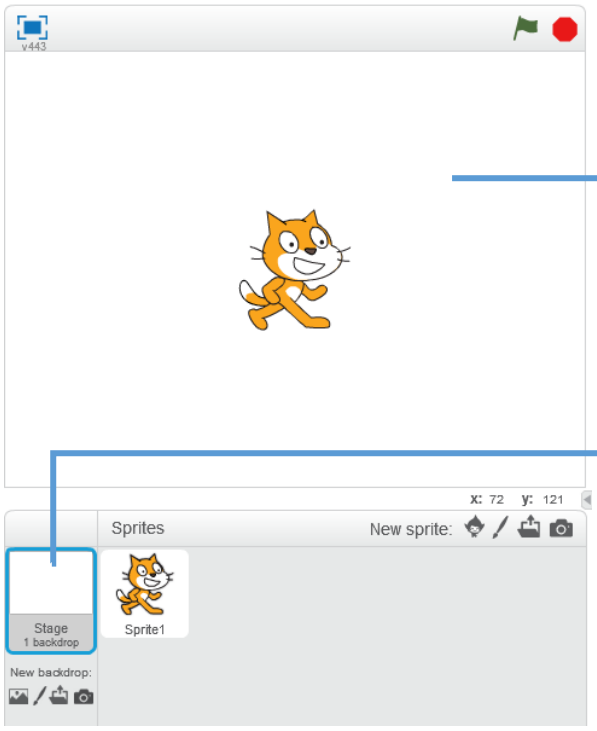
3. Hoạt động của em

Em hãy thiết lập 1 chương trình, bổ sung vào sân khấu từ 2 đến 5 nhân vật. Sau đó em viết các chương trình độc lập điều khiển các nhân vật này.

7. Thay đổi nền sân khấu

1. Nền sân khấu là gì?

Nền sân khấu là những gì thể hiện trên cửa sổ chính của chương trình. Các nhân vật đều hoạt động trên nền của sân khấu.

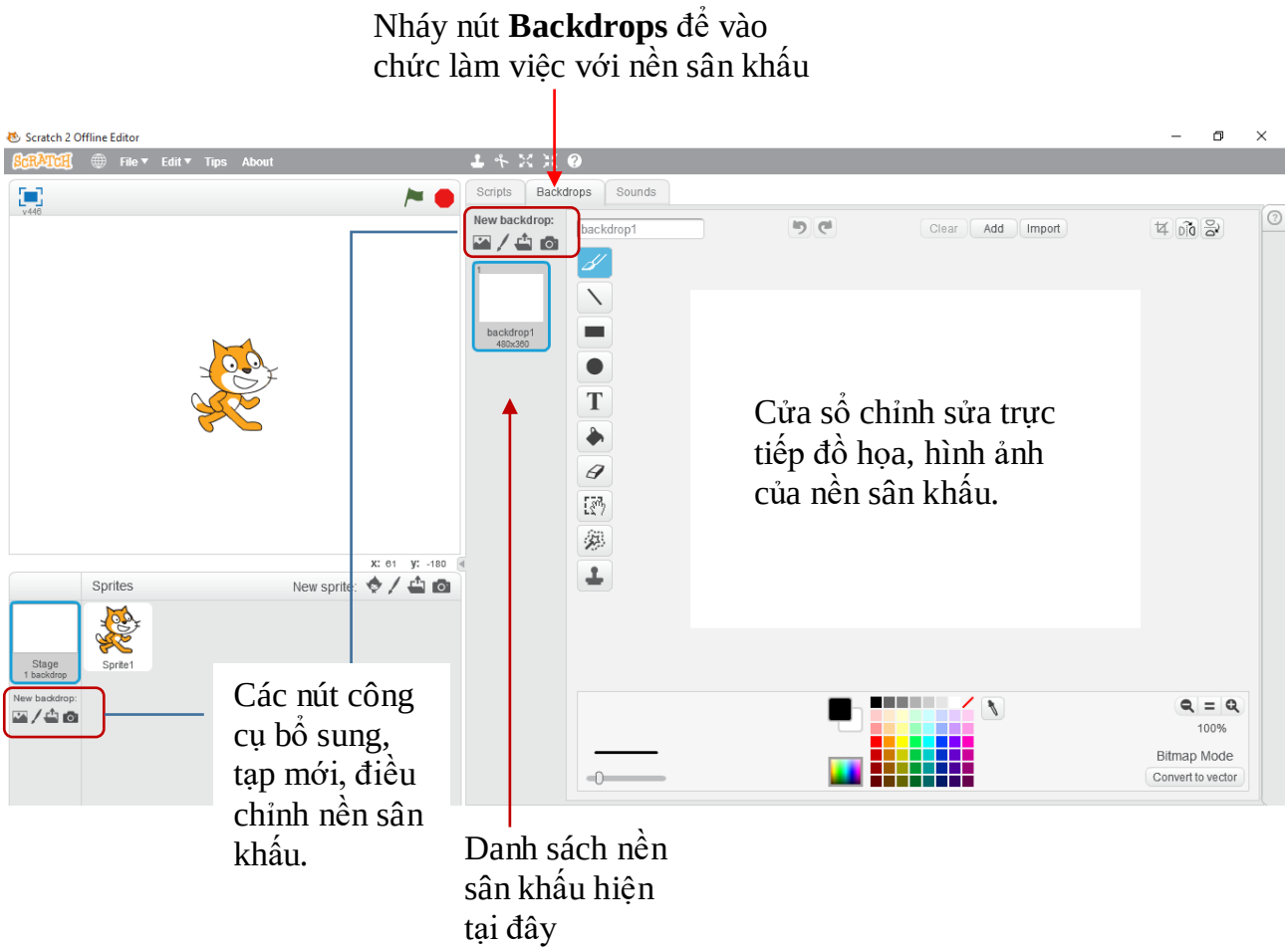


Hình nền sân khấu

Hình ảnh thu nhỏ của sân khấu và các chức năng với sân khấu ở đây

Trong hình trên, nền sân khấu là rỗng. Chúng ta có thể bổ sung thêm nhiều hình nền để đưa lên trang trí cho sân khấu.

Cửa sổ điều chỉnh, bổ sung nền sân khấu có dạng như hình sau.



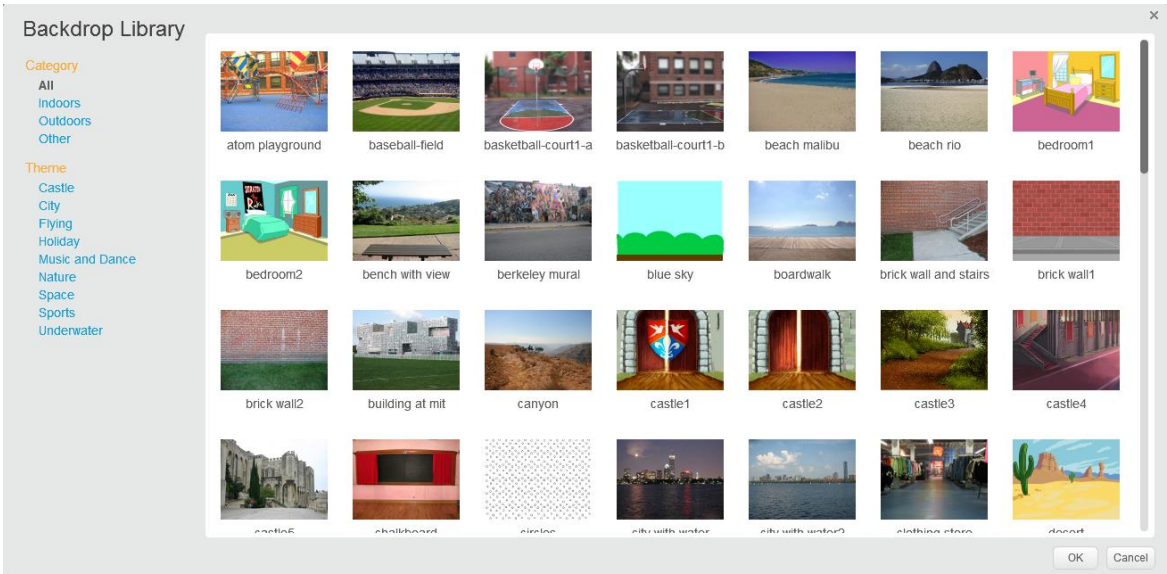
2. Bổ sung thêm nền sân khấu vào chương trình

Trong Scratch có 1 thư viện rất đa dạng các hình ảnh nền sân khấu khác nhau, em có thể sử dụng thư viện này.

Để bổ sung thêm hình ảnh sân khấu từ 1 thư viện có sẵn, em hãy thực hiện các bước sau.

- Nháy nút  tại khung thông tin sân khấu.

- Xuất hiện cửa sổ các hình nền sân khấu trong thư viện có sẵn.

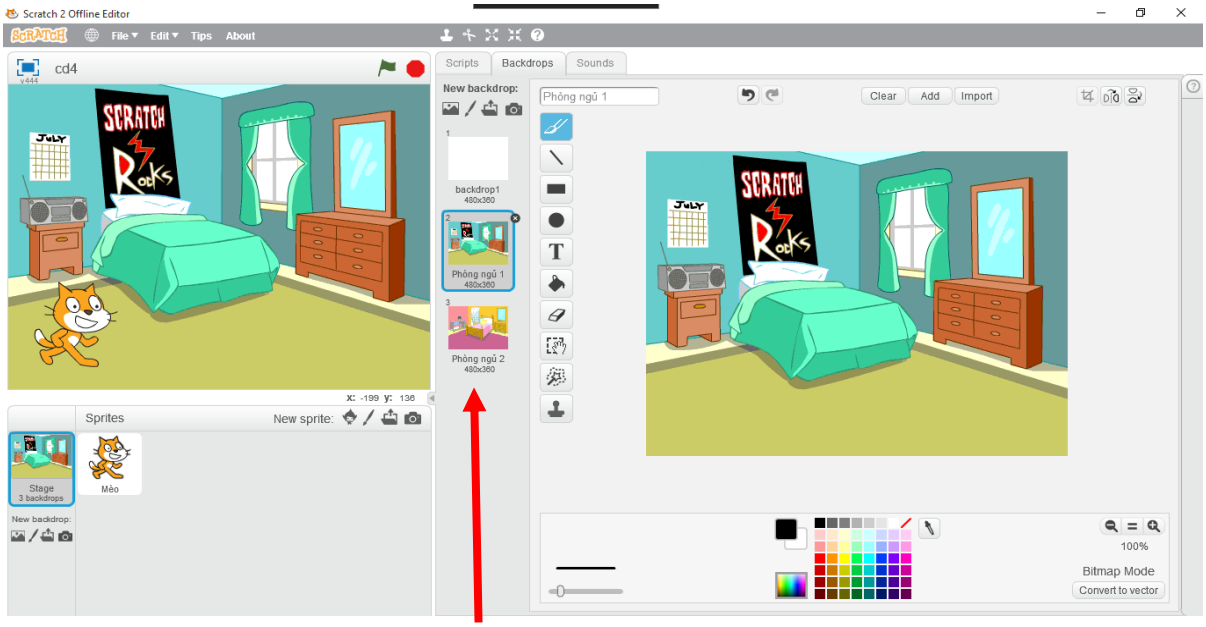


- Nháy chọn hình nền muốn đưa vào chương trình và nháy nút OK.

Hình nền sân khấu sẽ đưa vào chương trình tương tự hình sau.

Khung điều khiển nhân vật và sân khấu sau khi đã bổ sung thêm nhiều nhân vật và nền sân khấu.

Chú ý: khi nháy lên nút sân khấu phía dưới, cửa sổ **Backdrops** như hình sau xuất hiện cho phép chỉnh sửa trực tiếp hình ảnh đồ họa sân khấu. Có thể đặt tên các nền sân khấu trong cửa sổ này.



Dãy các hình nền sân khấu hiện ở đây

8. Cho nhân vật chào hỏi

Nhóm các lệnh Chuyển động chỉ điều khiển nhân vật di chuyển trên sân khấu, nhưng còn thể hiện bên ngoài của các nhân vật như ăn mặc, nói, hát thì phải dùng các lệnh của nhóm khác.

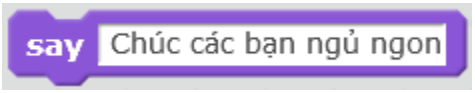
Chúng ta cùng làm quen với một vài lệnh đơn giản của nhóm lệnh **Thể hiện (Look)** của Scratch. Bắt đầu từ 1 chương trình như sau:

Chú mèo trước khi xuất phát sẽ chào khán giả: "Chào các bạn, tôi là Mèo con" trong 2 giây. Sau đó Mèo chạy vòng quanh sân khấu 1 vòng và quay trở lại vị trí cũ. Khi về đích, Mèo sẽ chào: "Chúc các bạn ngủ ngon".

Các lệnh được kéo thả vào chương trình như sau:

2 lệnh mới trong đoạn chương trình trên là:

Lệnh này thể hiện nội dung câu nói trên màn hình và dừng toàn bộ chương trình trong 1 thời gian nhất định (ví dụ 2 giây trong ví dụ trên). Hình ảnh thể hiện có ý nghĩa nhân vật đang nói.



Lệnh này thể hiện nội dung câu nói trên màn hình (không dừng toàn bộ chương trình). Hình ảnh thể hiện có ý nghĩa nhân vật đang nói.

Câu hỏi và bài tập

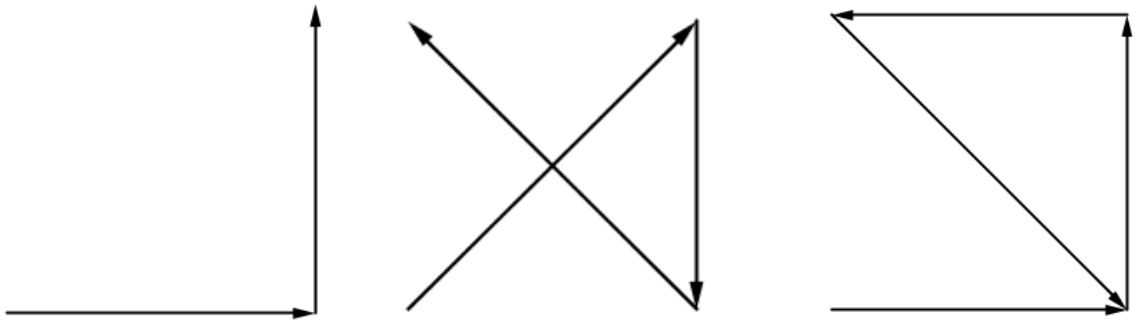
1. Viết chương trình điều khiển nhân vật thực hiện các công việc sau:

- Đi vòng quanh sân khấu 8 vòng theo chiều ngược kim đồng hồ.

- Đi vòng quanh sân khấu 5 vòng theo chiều kim đồng hồ.

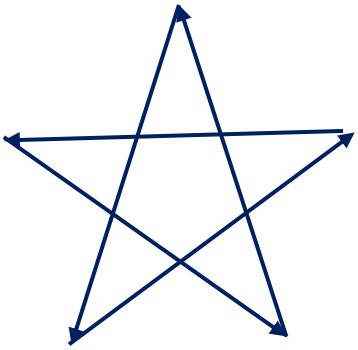
- Trước khi đi nói "Chào các bạn, tôi chuẩn bị đi". Sau đó nhân vật sẽ đi vòng quanh sân khấu 4 lần, mỗi lần đi về sẽ dừng lại và nói câu: "Tôi vừa hoàn thành công việc" trong khoảng thời gian 2 giây trước khi đi tiếp.

2. Em hãy thiết lập các chương trình để điều khiển nhân vật chuyển động theo các sơ đồ sau, chú ý đến mỗi ngã rẽ nên dừng lại vài giây để nghỉ ngơi.



A **B** **C**

3. Em hãy thiết lập các chương trình để điều khiển nhân vật chuyển động theo sơ đồ sau:



4. Thiết lập 1 chương trình có 2 nhân vật Chó và Mèo:

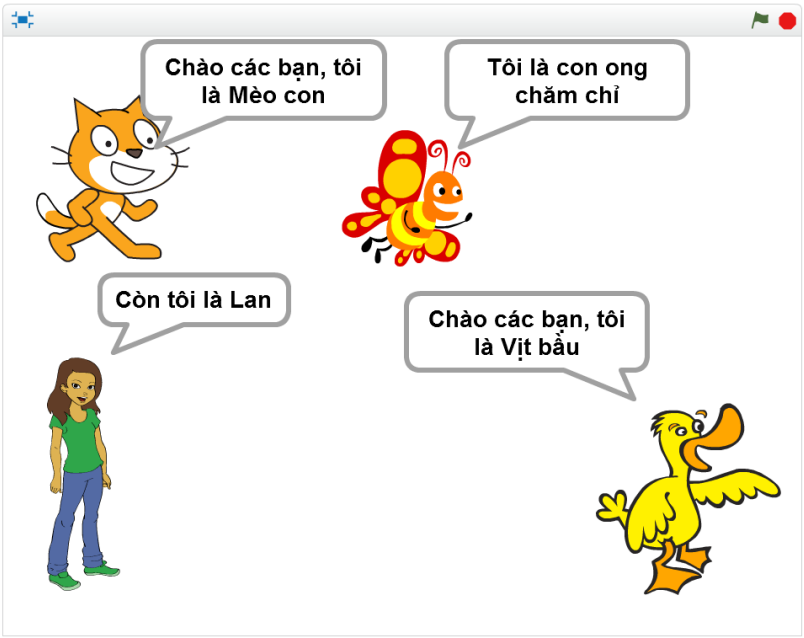
Lần lượt thiết lập chương trình cho 2 nhân vật này như sau:

Mèo

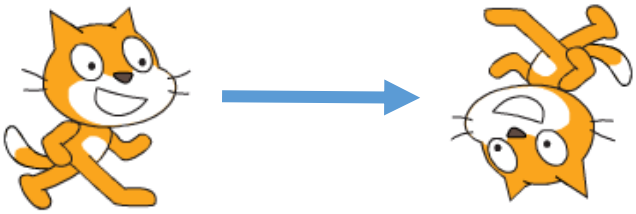
Chó

Em hãy chạy và mô tả kết quả của chương trình.

5. Em hãy viết 1 chương trình nhỏ tạo ra 4 nhân vật, khi chạy các nhân vật này chào hỏi như sau.



6. Có cách nào chỉ bằng 1 lệnh, em có thể cho con mèo xoay ngược thế này được không?



7. Muốn điều khiển nhân vật của chúng ta chuyển động chậm từ vị trí (-100,0) đến (100,0) thì cần viết những lệnh nào? 3 cách sau có đúng không? em chọn cách nào?

8. Giả sử vị trí ban đầu của nhân vật ở vị trí (-100, -100). Chỉ dùng các lệnh move, turn (phải hoặc trái) có thể điều khiển nhân vật đi đến vị trí (200, 100). Em hãy chỉ ra các lệnh thực hiện này.

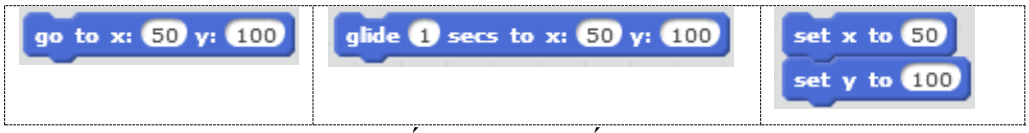
9. Em hãy trả lời các câu hỏi sau:

- Độ dài đường chéo chính của sân khấu là bao nhiêu (tính theo đơn vị bước của nhân vật)?

- Độ dài giữa 2 điểm có tọa độ (-10, -5) và (20, 35) là bao nhiêu?

10. Làm thế nào để biết 1 nhân vật đang đứng ở vị trí nào, tọa độ nào trên màn hình? Có lệnh của Scratch để tính tọa độ của nhân vật hay không?

11. Quan sát, làm thử và kiểm tra ý nghĩa của 3 nhóm lệnh sau, chúng có gì giống nhau, khác nhau.



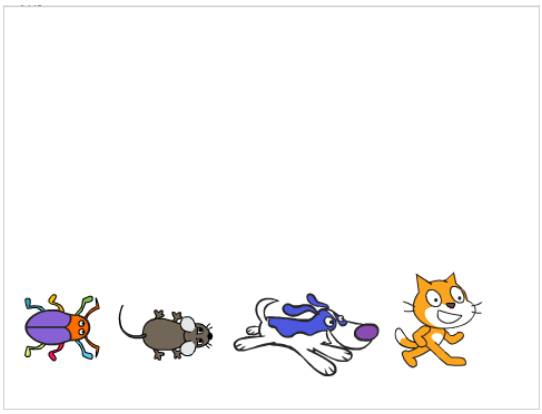
12. 4 nhóm lệnh sau sẽ cho kết quả như thế nào?

Mở rộng

1. Viết chương trình thực hiện các công việc sau:

- Thiết lập 4 nhân vật như hình dưới đây.

4 nhan vat chuyen dong.sb2



- Lần lượt cho mỗi nhân vật chuyển động quanh sân khấu theo chiều ngược kim đồng hồ 4 lần. Con này chạy sau con kia.

2. Yêu cầu điều khiển 4 con vật trên chạy chậm xung quanh màn hình, nhưng với điều kiện các con vật luôn bám sát nhau như trong hình ảnh trên.

Bài 4. Vẽ hình 1

Mục đích

Học xong bài này bạn sẽ biết:

- Sử dụng khái niệm bút vẽ để điều khiển nhân vật vẽ các đường, hình đơn giản trên màn hình.

- Thay đổi màu sắc, độ rộng, kiểu của nét vẽ.

- Làm quen với khái niệm cảm biến và các lệnh cảm biến đơn giản.

- Biết các câu lệnh điều khiển rẽ nhánh (selection).

Bắt đầu

1. Giả sử có 1 họa sĩ với 1 số bút vẽ, bảng màu và 1 tờ giấy trắng. Họa sẽ bắt đầu vẽ. Trong các từ sau, từ nào liên quan đến công việc vẽ của họa sĩ này:

- Trải giấy ra mặt bàn.
- Nhấc bút.
- Uống nước.
- Hạ bút.
- Hát.
- Nghỉ giải lao.
- Di chuyển bút trên giấy.
- Lia bút, rê bút.
- Chọn màu khác.
- Chọn bút khác.

2. Em đã từng làm quen với các phần mềm tập vẽ trên máy tính chưa? Hãy kể 1 vài chương trình như vậy.

3. Trong bài học này chúng ta sẽ được làm quen và học cách Scratch có thể vẽ được trên màn hình dưới sự điều khiển của chương trình. Chương trình sẽ điều khiển những công cụ sau đây:

- bút vẽ
- bảng màu
- nhấc bút, hạ bút

và nhiều tính năng nổi bật khác.

Hoạt động bài học

1. Chế độ vẽ theo chuyển động nhân vật

Trong Scratch, chế độ vẽ được thiết lập rất đơn giản theo nguyên tắc sau:

pen down: hạ bút, thiết lập chế độ vẽ.

pen up: nâng bút, hủy chế độ vẽ.

- Đặt chế độ hạ bút (**pen down**).

- Khi đó khi nhân vật chuyển động sẽ tạo ra vết trên đường đi của mình, đó chính là nét vẽ được tạo ra theo màu sắc, kiểu bút có sẵn của hệ thống.

- Trong quá trình vẽ có thể thay đổi màu vẽ, nét bút vẽ.

- Để kết thúc chế độ vẽ cần nhấc bút (**pen up**).

Các lệnh của nhóm vẽ (pen) đều có màu xanh lá cây.

Chúng ta chú ý các lệnh sau:

- Lệnh thiết lập chế độ vẽ: 

, Sau lệnh này mọi chuyển động của nhân vật sẽ đều để lại nét vẽ trên đường đi của mình.

- Lệnh hủy chế độ vẽ: 

. Chú ý sau lệnh này các hình vẽ không bị xóa trên màn hình.

- Lệnh thiết lập màu vẽ: 

. Cách xác định màu vẽ bởi lệnh này như sau: nháy chuột lên ô vuông của lệnh, sau đó nháy chuột lên 1 vị trí bất kỳ trên màn hình có màu sắc muốn xác định.

- Lệnh thiết lập động rộng nét vẽ: 

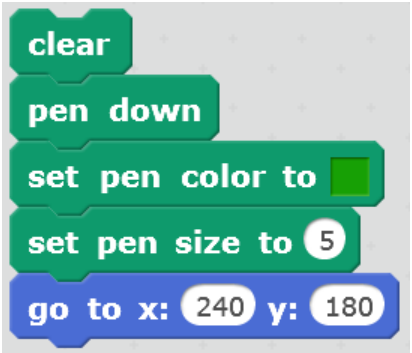
. Giá trị của lệnh là độ rộng nét bút tính theo pixel.

- Lệnh xóa tất cả các hình vẽ trên màn hình: 

.

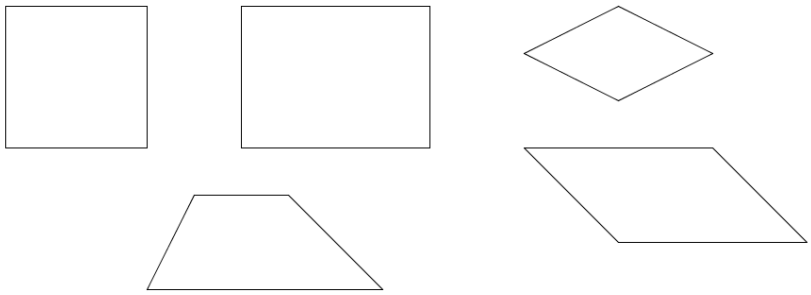
Em hãy thiết lập 1 chương trình vẽ đơn giản như mô tả dưới đây và quan sát kết quả.

```
Xóa các hình được vẽ trước đó.
Thiết lập chế độ vẽ: hạ bút.
Đặt màu vẽ là màu: xanh.
Thiết lập độ rộng nét bút = 5.
Cho nhân vật chuyển động lên
vị trí góc phải trên của sân
khấu.
```



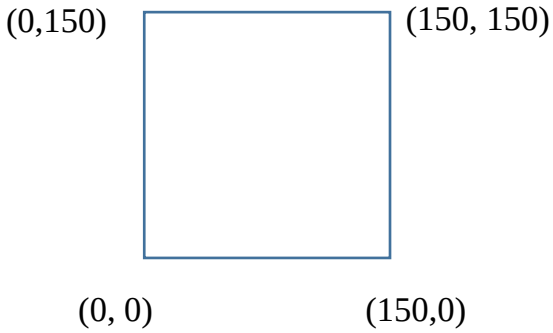
2. Vẽ 1 số hình hình học đơn giản

Các em sẽ cùng thực hành vẽ các hình hình học sau bằng Scratch: hình vuông, chữ nhật, hình thoi, hình bình hành, hình thang.



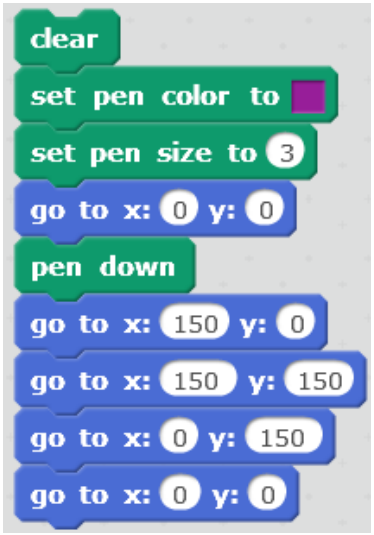
Để thực hiện các chương trình vẽ được các hình trên chúng ta cần xác định tọa độ các đỉnh của các hình này.

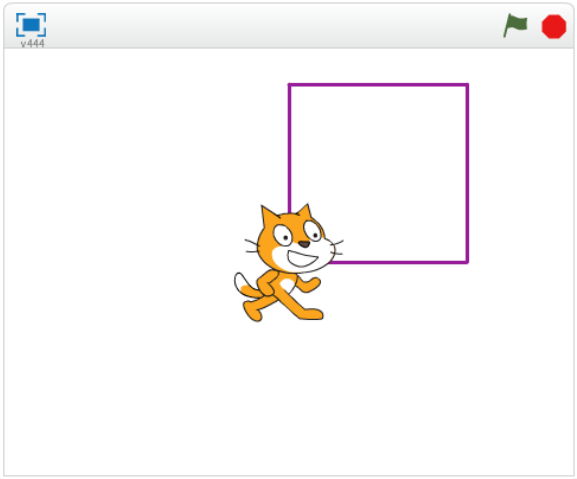
Ví dụ với hình vuông, chúng ta xác định tọa độ các đỉnh là:



Em viết chương trình điều khiển nhân vật bắt đầu chuyển động từ vị trí (0,0), sau đó lần lượt di chuyển đến các vị trí (150,0), (150,150), (0,150), cuối cùng quay trở lại vị trí ban đầu (0,0).

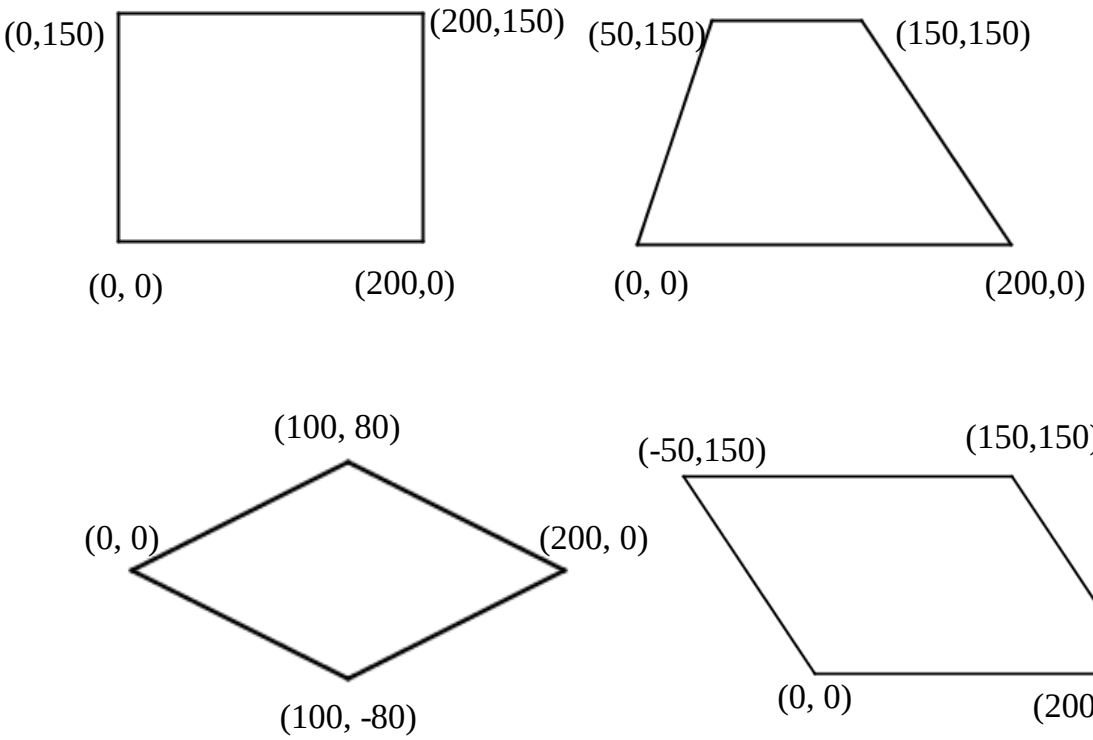
Chương trình có thể được thiết lập như hình dưới đây. Chú ý rằng lệnh "go to x:0 y:0" được đặt trước lệnh "pen down" chỉ ra rằng chỉ sau khi nhân vật dịch chuyển đến vị trí (0,0) mới bắt đầu quá trình vẽ.





Kết quả của chương trình được thể hiện ở hình phải.

Tương tự chúng ta thiết lập tọa độ của các hình còn lại và tạo chương trình vẽ tương ứng.



Ẩn nhân vật trong các bài toán vẽ hình

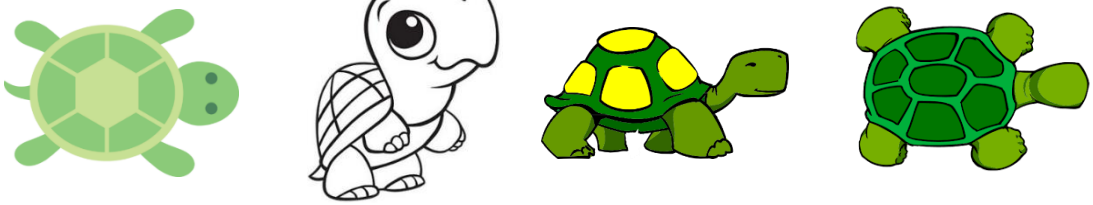
Trong 1 số bài toán vẽ hình nếu không có nhu cầu hiển thị nhân vật thì Scratch cho phép ẩn, không hiển thị (tạm thời) nhân vật bằng lệnh 

từ nhóm Hiển thị (Look). Ngược lại lệnh 

có tác dụng hiển thị trở lại hình ảnh nhân vật trên màn hình.

Sử dụng hình ảnh con rùa làm nhân vật của bài toán vẽ hình

Logo là phần mềm đầu tiên cho phép lập trình tạo bút vẽ trên màn hình. Trong phần mềm Logo sử dụng hình ảnh con rùa để vẽ. Chúng ta cũng có thể tạo ra các nhân vật hình ảnh Rùa trong các bài tập vẽ hình để tạo ra khung cảnh giống trong phần mềm Logo.

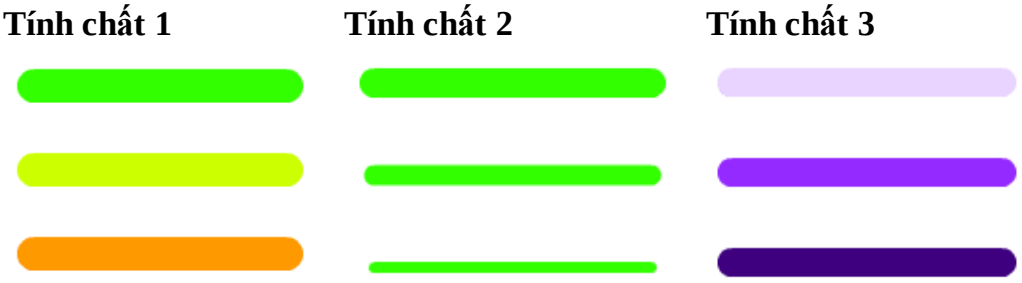


3. Thay đổi màu và nét bút

Quan sát các tính chất sau liên quan đến các tính chất của công việc vẽ.

3 tính chất quan trọng của bút vẽ:

- Màu vẽ.
- Độ rộng bút.
- Độ mờ.



Em hãy chỉ ra tên của các thuộc tính trong các cột của bảng trên.

Có 3 thông số sau liên quan đến bút vẽ trong Scratch: màu vẽ (**color**), độ rộng nét bút (**size**) và độ mờ nét bút (**shade**).

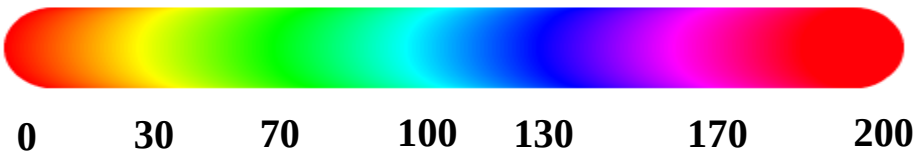
Màu bút vẽ (pen color)

Màu sắc của bút vẽ được thiết lập bởi lệnh 

. Giá trị màu sắc của bút vẽ trong Scratch được đánh số từ 0 đến 200 theo dải màu cầu vồng.

Dải màu theo sắc cầu vồng được thiết lập trong Scratch, lấy giá trị số từ 0 đến 200.

Dải các giá trị màu của bút vẽ từ 0 đến 200.



Ta có bảng giá trị màu ngắn sau:

Giá trị	Màu	Giá trị	Màu	Giá trị	Màu
0	Đỏ	**70**	Xanh lá cây	**130**	Xanh thẫm
30	Vàng	**100**	Xa da trời	**170**	Hồng

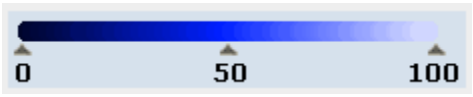
Độ rộng nét bút (pen size)

Độ rộng của nét bút được tính bằng pixel (điểm). Lệnh 

sẽ thiết lập kích thước của nét bút vẽ hiện thời.

Độ mờ nét bút (pen shade)

Độ mờ (hay độ đậm nhạt) của bút vẽ được đo bởi tham số shade có gía trị từ 0 đến 100. Giá trị 0 là đậm nhất (độ mờ = 0), giá trị = 100 là mờ hoàn toàn.



Lệnh 

thiết lập độ mờ của nét bút.

Ngoài 3 lệnh thiết lập thông số chính thức cho màu bút, kích thước và độ mờ nét bút, trong Scratch còn có các lệnh làm thay đổi các giá trị này.

Thay đổi màu của bút theo giá trị được nhập trong lệnh.

Thay đổi kích thước nét bút theo giá trị được nhập trong lệnh.

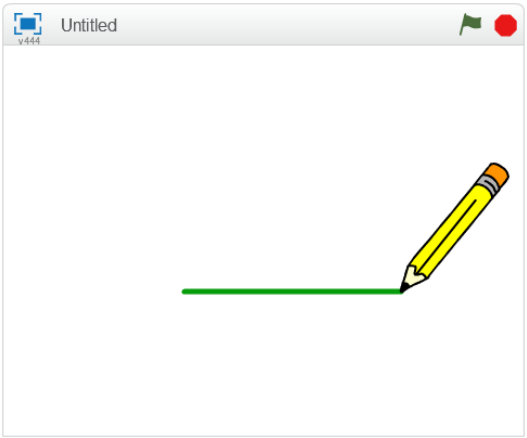
Thay đổi độ mờ bút theo giá trị được nhập trong lệnh.

Em hãy thiết lập các chương trình thực hiện kết quả thay đổi màu sắc, kích thước và độ mờ sử dụng các mẫu lệnh trên.

Em hãy viết lại các đoạn chương trình thể hiện trên màn hình các hình vẽ sau:

Test_Color_pen.sb2 Test_Size_pen.sb2 Test_Shade_pen.sb2

4. Thiết lập bút vẽ riêng

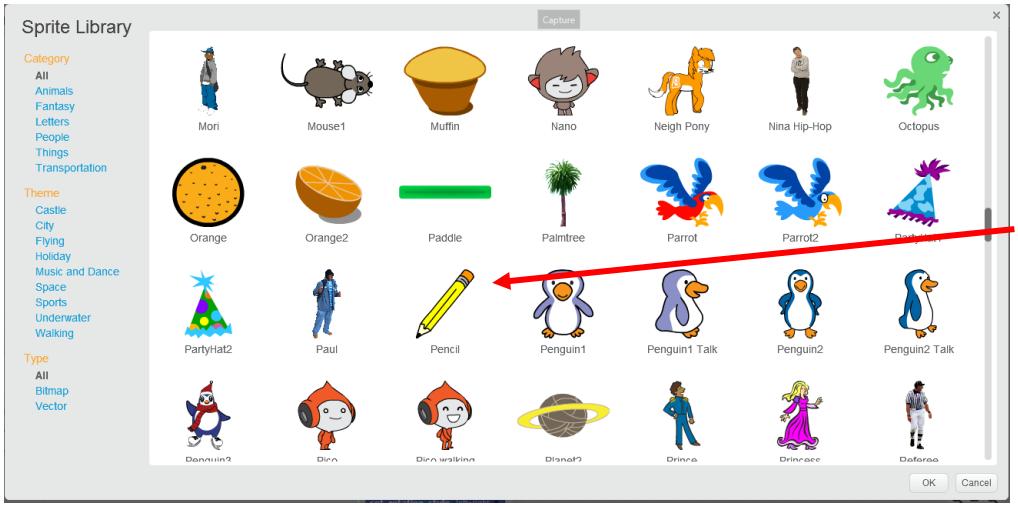


Mục đích của chúng ta trong bài luyện này là thiết lập một bút vẽ (như 1 nhân vật trên sân khấu) và có khả năng vẽ các nét bút đúng tại vị trí đầu bút như hình bên.

Chú ý nét vẽ của bút xuất phát từ chính đầu ngòi bút trong hình.

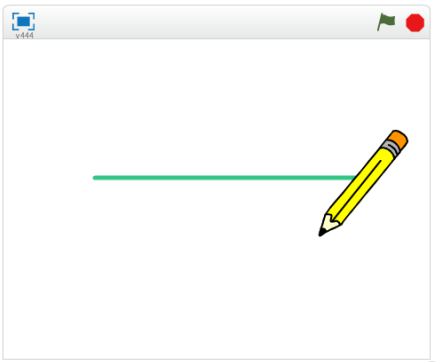
Em hãy thực hiện các bước sau đây:

- Bổ sung thêm 1 nhân vật vào chương trình. Lần này hãy chọn nhân vật bút chì (pencil) trong danh sách để đưa vào sân khấu.



Chọn bút vẽ theo hình ảnh này (pencil).

- Sau đó em có thể xóa nhân vật mèo bằng cách nháy chuột phải lên mèo tại cửa sổ điều khiển nhân vật và chọn lệnh **delete**.



Bây giờ Bút chì đã hiện như nhân vật duy nhất trên màn hình. Tuy vậy nếu thực hiện lệnh vẽ chúng ta sẽ thấy nét vẽ xuất phát từ điểm giữa của bút chì (xem hình ảnh bên).

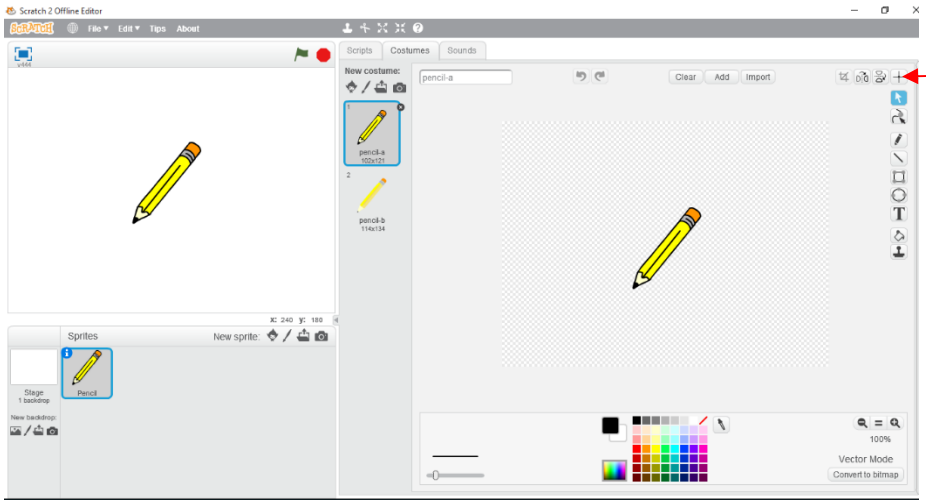
Trong Scratch, nét vẽ sẽ xuất phát từ **tâm** của nhân vật. Do vậy muốn nét vẽ đi qua đầu bút chì chúng ta cần chuyển tâm của hình về vị trí đầu bút chì.

Cách thực hiện như sau:

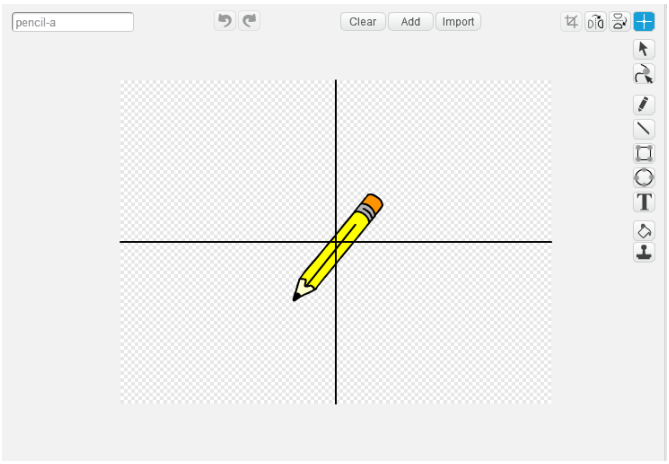
- Nháy chọn bút chì trong khung điều khiển nhân vật.

- Nháy lên TAB Costume. Sẽ xuất hiện màn hình chỉnh sửa hình ảnh trang phục của nhân vật hiện thời, cụ thể ở đây là bút chì.

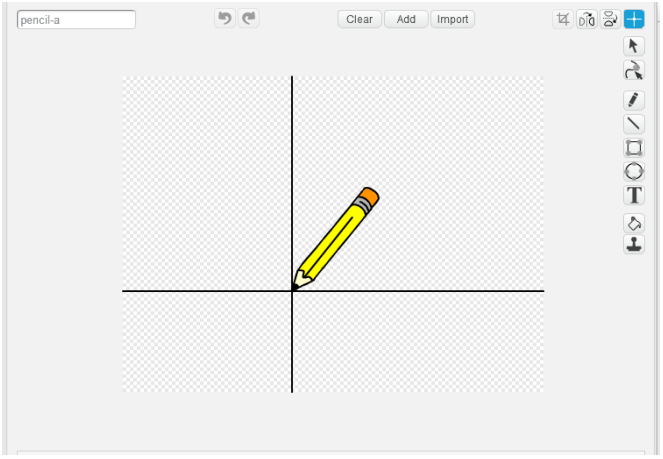
Bên phải là cửa sổ chỉnh sửa đồ họa của nhân vật đang chọn (ở đây là bút chì). Nhiệm vụ của chúng ta là chuyển tâm của nhân vật về đầu bút. Em thực hiện theo các bước sau.



1. Nháy chuột lên nút có hình dấu cộng (+) này. Đây là chức năng chọn tâm của hình nhân vật.

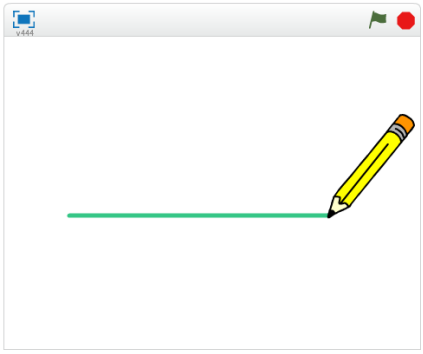


2. Xuất hiện 2 đường vạch ngang, dọc như hình bên. Em hãy nhấn giữ và rê chuột đến điểm đầu bút và nhả chuột.



3. Nhấn giữ và rê chuột đến điểm đầu bút và nhả chuột.

Bây giờ tâm của bút sẽ nằm ở đầu bút chì.



4. Công việc điều chỉnh vị trí tâm nhân vật bút chì đã hoàn thành. Em có thể kiểm tra để xem bút sẽ vẽ như thế nào.

Hãy sử dụng bút này để vẽ 1 hình vuông trên màn hình.

5. Vẽ theo điều kiện và sự kiện cảm biến

Những chương trình mà chúng ta đã biết đều là 1 dãy các lệnh bắt đầu bằng lệnh sự kiện 

. Lệnh này điều khiển đoạn chương trình gắn phía dưới chạy khi nháy chuột lên lá cờ.

Trong nhóm các lệnh sự kiện còn có 1 lệnh điều khiển theo sự kiện nháy lên 1 phím bất kỳ. Đó là lệnh 

. Lệnh này có ý nghĩa tương tự lệnh trên, chỉ khác là nhóm các lệnh gắn bên dưới sẽ thực hiện khi phím tương ứng được nháy.

Chúng ta hãy cùng thiết lập chương trình sau:

Nếu có sự kiện nháy lên lá cờ: khởi tạo chế độ vẽ đồ họa với các tham số ban đầu.

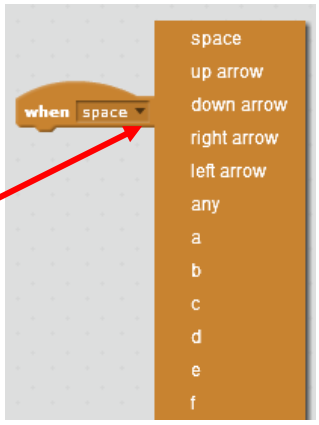
Nếu sự kiện một trong các phím ← → ↓ được bấm, nhân vật sẽ xoay các góc 90 độ trái, 90 độ phải và 180 độ.

Nếu phím ↑ được bấm, nhân vật sẽ tiến lên phía trước 10 bước.

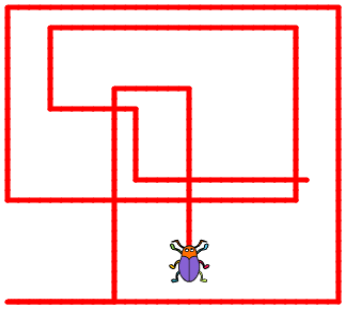
Chúng ta sử dụng lệnh 

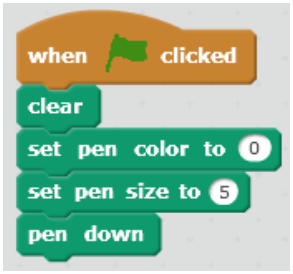
, trong đó phím tương ứng được chọn bằng cách nháy chuột lên nút hình tam giác và chọn tên phím tương ứng của sự kiện. Trên cửa sổ lệnh của nhân vật này sẽ đặt 5 chương trình như sau.

Nháy chuột lên vị trí này sẽ hiện ra tất cả các phím trên bàn phím. Lựa chọn 1 để thiết lập lệnh cho sự kiện nháy phím này.



Thiết lập chương trình như dưới đây và quan sát hoạt động của chương trình.





Các đoạn chương trình này được đặt trong cùng 1 cửa sổ lệnh của nhân vật, và được thực hiện đồng thời.

Chú ý: Trong các bài tiếp theo các em sẽ còn được học thêm cách điều khiển bàn phím bằng các công cụ cảm biến khác của Scratch.

6. Bổ sung cảm biến bàn phím

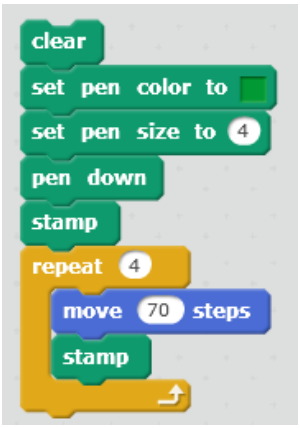
Mở rộng ví dụ trên, em hãy bổ sung thêm cảm biến như sau: trong khi điều khiển nhân vật chuyển động, vẽ trên màn hình, nếu bấm phím x: đổi màu vẽ thành xanh lá cây, phím v: màu vàng, phím d: màu đỏ.

Nhớ lại qui định các màu sắc như sau: Xanh: 130; Vàng: 30; Đỏ: 0 chúng ta sẽ bổ sung thêm các nhóm lệnh sau:

Em hãy viết hoàn chỉnh chương trình này.

7. Lệnh Stamp vẽ hình của nhân vật lên màn hình

Lệnh Stamp có chức năng in hình nhân vật lên màn hình tại vị trí và thời điểm thực hiện lệnh (với điều kiện đang ở chế độ hạ bút).

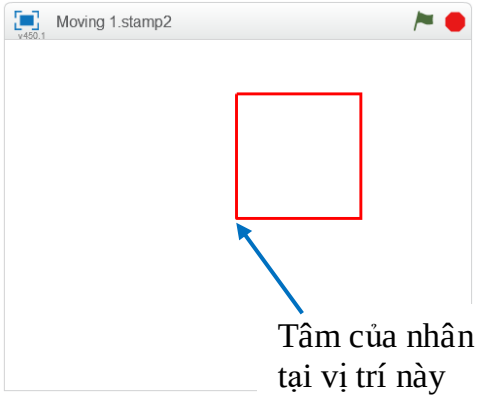


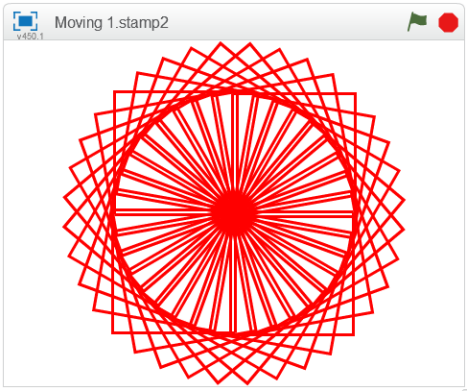
Em hãy thiết lập chương trình để có thể vẽ được hình dưới đây.

Em cần tạo một nhân vật mới có hình tròn màu xanh đặc, sau đó cần đặt tâm của nhân vật đúng vị trí tâm hình tròn.

Bây giờ em hãy thiết lập chương trình sau sử dụng lệnh **Stamp** để tạo các hình ảnh nghệ thuật.

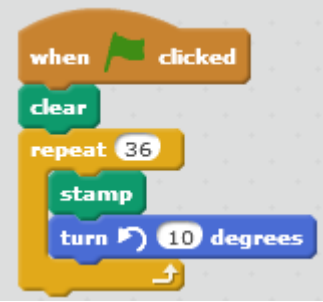
a) Thiết lập 1 nhân vật là 1 hình vuông với tâm chính là đỉnh trái dưới (xem hình bên dưới.





b) Em viết đoạn chương trình ngắn sau và kiểm tra kết quả.

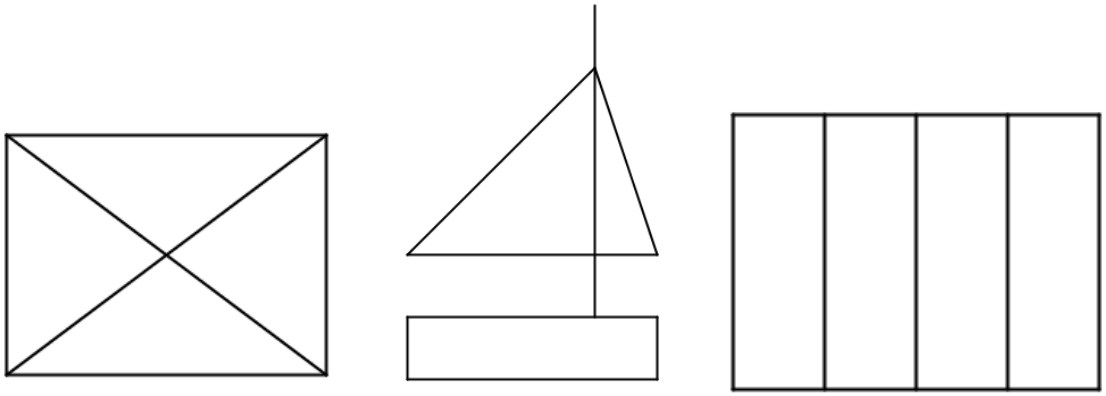
Vd_
Stamp3.sb2



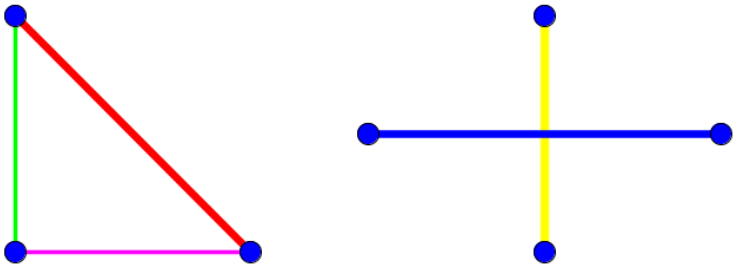
Chương trình sẽ cho nhân vật (hình vuông) xoay 1 vòng tròn 360 độ theo chiều ngược kim đồng hồ. Việc xoay này được thực hiện bởi lệnh lặp 36 lần, mỗi lần hình vuông xoay 10 độ và để lại hình ảnh của mình bằng lệnh stamp.

Câu hỏi và bài tập

1. Hãy viết chương trình để thực hiện vẽ các hình sau.



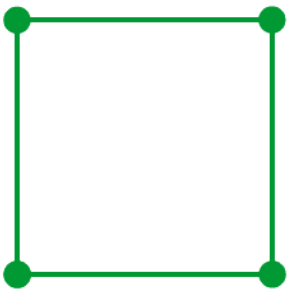
2. Hãy viết chương trình vẽ các hình sau, kết hợp các công cụ màu sắc và stamp.



3. Trong ví dụ của mục 5, em hãy thay đổi và bổ sung thêm yêu cầu sau:

- Khi nhân vật đi sang trái thì màu vẽ sẽ vàng, khi sang phải màu vẽ là xanh lá cây, đi lên với màu đỏ và đi xuống với màu xanh thẫm.

4. Sử dụng lệnh **Stamp** để viết chương trình vẽ hình sau:



5. Dựa trên ví dụ mẫu của mục 7, em hãy viết các chương trình để tạo ra các hình nghệ thuật khác nhau.

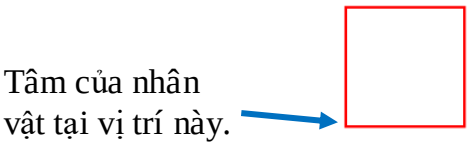
Một số hình ảnh gốc có thể dùng làm hình của nhân vật.

- Các hình đa giác đều.

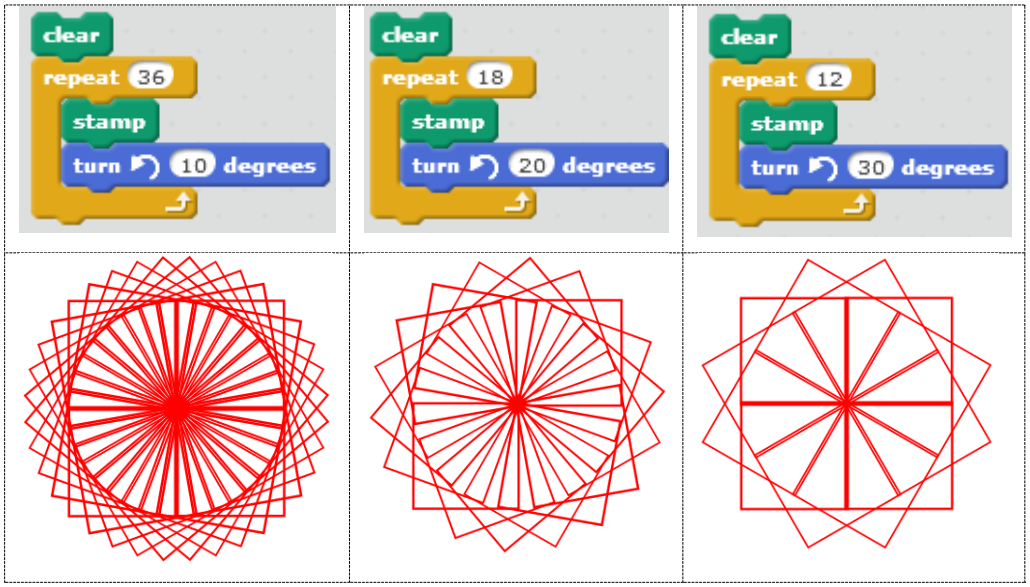
- Hình tròn.

- Các hình dạng sao.

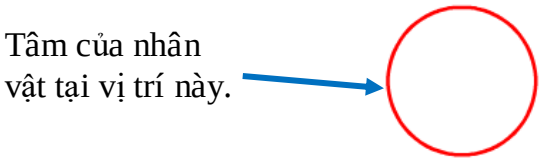
6. Thiết kế nhân vật là 1 hình vuông với tâm là đỉnh trái dưới:



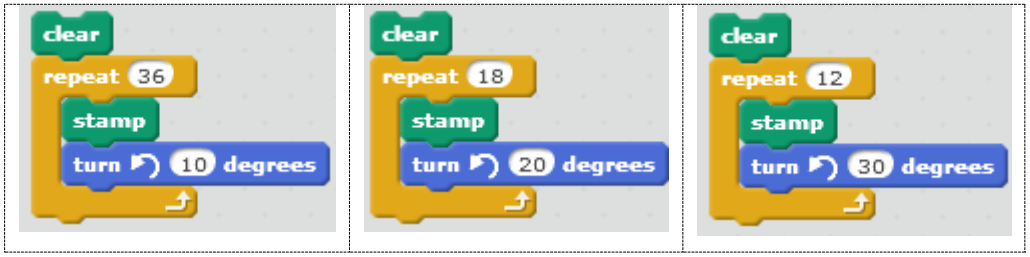
Em hãy thiết lập các đoạn lệnh như trong bảng sau và quan sát kết quả.

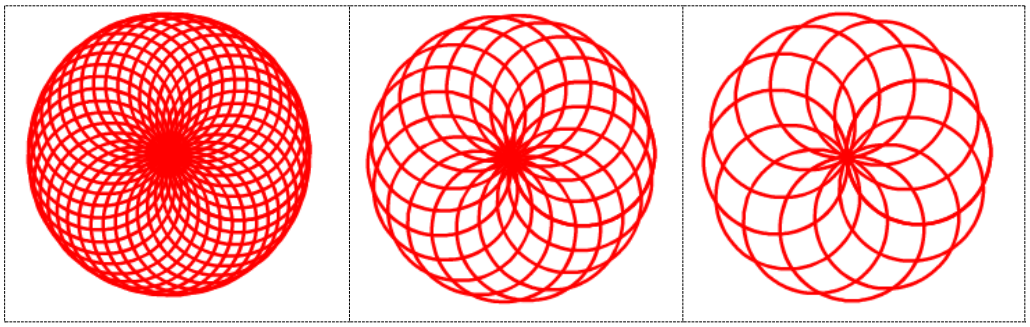


7. Thiết kế nhân vật là 1 hình tròn với tâm là vị trí ngoài cùng bên trái:



Em hãy thiết lập các đoạn lệnh như trong bảng sau và quan sát kết quả.



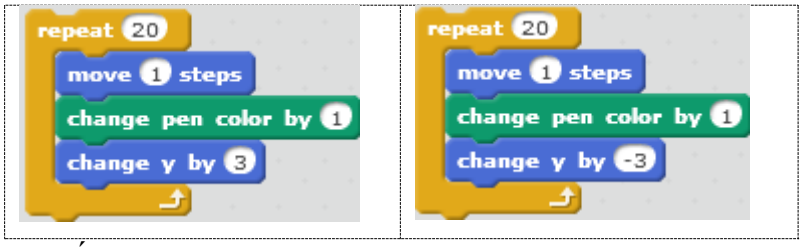


8. Thiết kế chương trình vẽ hình sau đây.

Gợi ý: Chương trình bao gồm các đoạn lặp, mỗi đoạn lặp lại chứa phần đưa bút vẽ lên, và phần tiếp theo đưa bút vẽ xuống. Trong suốt quá trình lặp, luôn thực hiện lệnh 

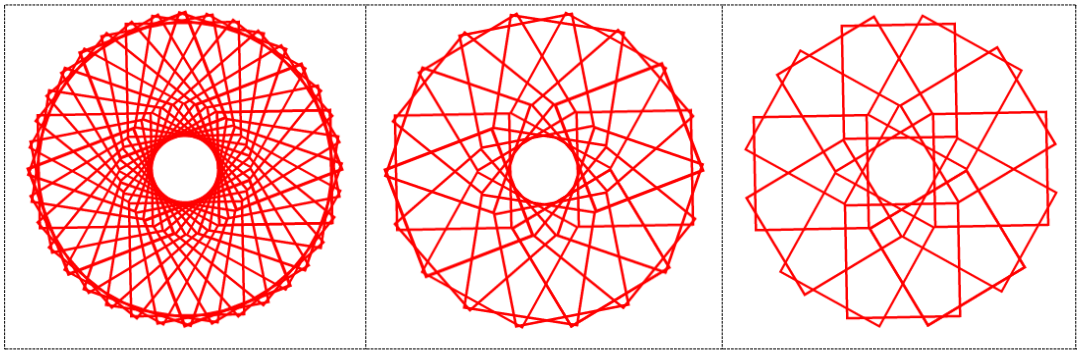
.

2 đoạn mô tả nét bút lên và xuống sẽ có dạng tương tự sau:



9. Viết chương trình vẽ hình sau:

10. Viết chương trình vẽ các hình sau:



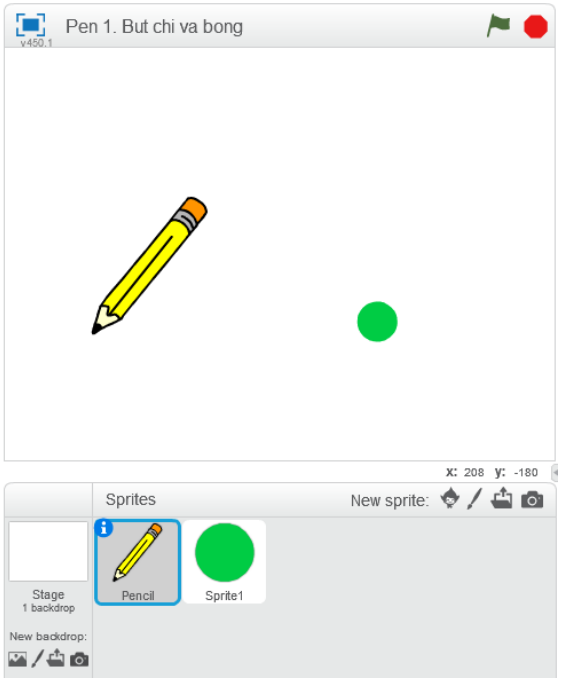
Mở rộng

Xây dựng chương trình có các chức năng sau.

- Có 2 nhân vật là bút chì và quả bóng hình tròn.

- Khi chạy chương trình, cả 2 bút chì và quả bóng đều vẽ lên 1 hình vuông ở 2 bên màn hình.

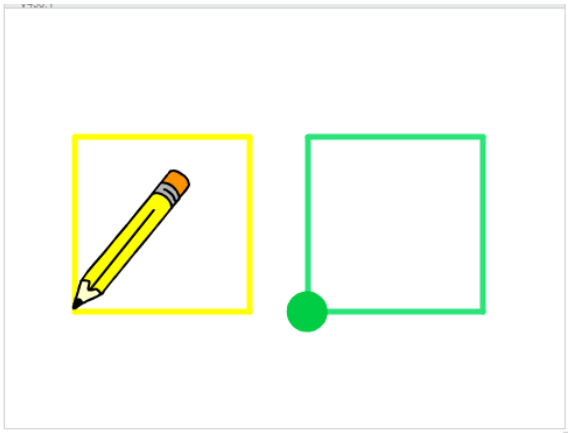
- Bút chì kẻ hình vuông màu vàng, quả bóng tạo hình vuông màu xanh.



Thiết lập 2 nhân vật như hình bên: bút chì và quả bóng tròn.

Lập trình riêng biệt cho 2 nhân vật này.

Kết quả của chương trình phải được như hình sau:



Bài 5. Âm thanh 1

Mục đích

Học xong bài này, bạn sẽ biết:

- Kết nối âm thanh với nhân vật. Nhân vật nói và thể hiện lời nói bằng chữ và âm thanh.

- Các lệnh tạo âm thanh, đánh trống và chơi nhạc trên nền sân khấu.

- Điều khiển nhân vật nhảy múa.

Bắt đầu

1. Chúng ta đã biết lệnh **say** có ý nghĩa như thế nào.

Lệnh 

sẽ làm cho mèo con kêu meo meo như sau

Tuy vậy chúng ta chỉ nhìn thấy mèo kêu mà không nghe thấy giọng nói của mèo. Vậy có cách nào để nghe được giọng của mèo, hay của bất kỳ một nhân vật nào khác hay không?

2. Trong bài học này, em sẽ được làm quen và học được các lệnh liên quan đến âm thanh. Có rất nhiều điều thú vị đang ở phía trước.

- Âm thanh, giọng nói có thể phát ra từ nhân vật hay không? Nhân vật sẽ "nói" như thế nào?

- Nhạc nền của sân khấu có thể bật lên được hay không?

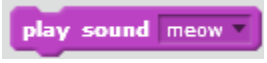
- Nhân vật của chúng ta có thể hát được không?

- Nhân vật có thể chơi các nhạc cụ, đánh trống, thổi kèn được hay không?

Nhưng câu hỏi như vậy em sẽ được lần lượt làm quen trong bài học này.

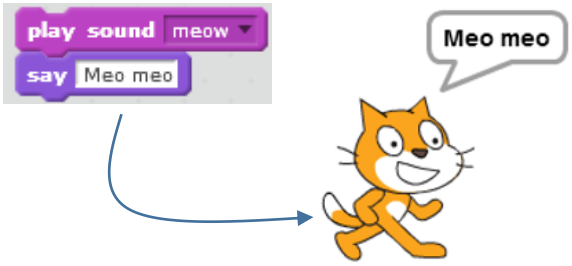
Nội dung bài học

1. Nhân vật có thể nói, hát

Chúng ta hãy bắt đầu từ nhóm lệnh Sound (âm thanh) và thực hiện lệnh 

trong nhóm lệnh này.

Chương trình ngắn sau đây sẽ cho chúng ta cảm giác nhân vật có thể nói, hội thoại và giao tiếp bằng âm thanh trong Scratch.



Em sẽ nghe và nhìn thấy gì?

Em sẽ nghe thấy tiếng kêu meo meo của Mèo và nhìn thấy dòng chữ Meo meo trên màn hình.

Như vậy các nhân vật trong Scratch không những có thể "nói" bằng chữ trên màn hình, mà còn có thể hát, nói bằng âm thanh thật của mình thông qua loa của máy tính.

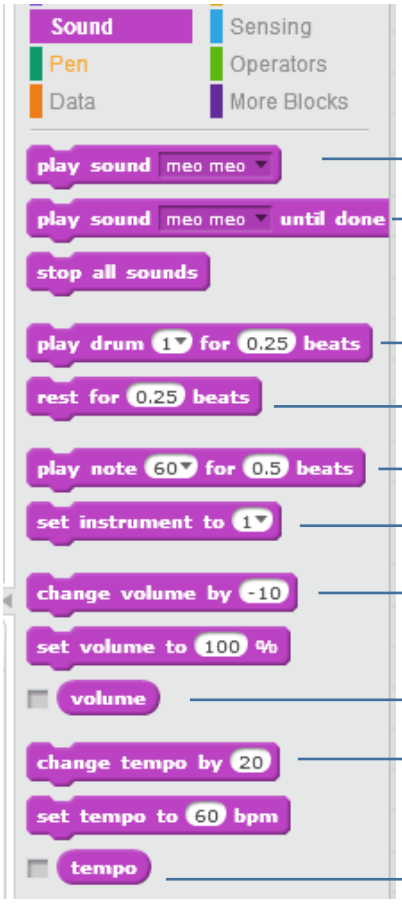
Mỗi nhân vật trong Scratch đều có thể có một hay nhiều âm thanh, giọng nói đi kèm. Số lượng âm thanh của mỗi nhân vật là không hạn chế.

Lệnh **play sound** cho phép nhân vật thể hiện âm thanh của mình trực tiếp trên máy tính.

Trong lệnh play sound cho phép hiển thị và lựa chọn âm thanh tương ứng của nhân vật. Chọn âm thanh cần thể hiện cho nhân vật này.

2. Nhóm lệnh âm thanh

Nhóm lệnh âm thanh (sound) có màu tím sẫm. Các lệnh làm việc với âm thanh rất đa dạng, chúng ta sẽ bắt đầu bằng nhóm lệnh đơn giản đầu tiên: các lệnh bật, tắt âm thanh của nhân vật.



Các lệnh bật âm thanh (sound) của nhân vật (hoặc sân khấu)

Các lệnh đánh trống và liên quan đến nhịp trống

Các lệnh chơi nhạc và liên quan đến nhạc cụ

Các lệnh làm việc với cường độ âm thanh

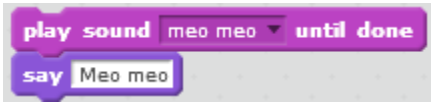
Các lệnh làm việc với nhịp trống

Chú ý quan trọng:

> Nhóm lệnh âm thanh của "nhân vật" và "sân khấu" hoàn toàn giống nhau. Như vậy sân khấu cũng có thể nói, hát giống như mọi nhân vật khác.
>
> Chúng ta có thể hiểu âm thanh của sân khấu chính là âm thanh được phát từ phía sau cách gà của sân khấu.

Em hãy thử và phân biệt sự khác biệt khi thực hiện của các nhóm lệnh sau:

Lệnh đầu tiên phát âm thanh "meo meo", ngay sau đó thực hiện lệnh **say** mà không cần đợi âm thanh kết thúc.



Lệnh đầu tiên phát âm thanh "meo meo" và đợi cho đến khi âm thanh kết thúc mời thực hiện lệnh **say**.

Lệnh **say** được thực hiện trước, sau đó đến lệnh **play sound**.

Em hãy thực hiện cả 3 nhóm lệnh trên và trả lời các mệnh đề sau đúng hay sai.

1. Cả 3 đoạn lệnh trên có tác dụng hoàn toàn giống nhau.

2. Các đoạn lệnh số 1 và 3 có tác dụng giống nhau.

3. Lệnh **play sound** sẽ thực hiện bật âm thanh có trong lệnh này đồng thời với các lệnh tiếp theo của lệnh này.

4. Lệnh **play sound** sẽ thực hiện bật âm thanh có trong lệnh, chờ nghe xong âm thanh sẽ thực hiện các lệnh tiếp theo.

5. Lệnh **play sound .. until done** sẽ thực hiện bật âm thanh có trong lệnh, chờ nghe xong âm thanh sẽ thực hiện các lệnh tiếp theo.

Chú ý: Lệnh 

sẽ dừng tất cả các âm thanh đang bật trong chương trình.

3. Âm thanh, nhạc nền sân khấu

Âm thanh không chỉ có ở nhân vật, mà sân khấu cũng có thể có âm thanh, nhạc nền của riêng mình. Chúng ta hãy thực hiện một hoạt động đơn giản sau để hiểu ý nghĩa của âm thanh sân khấu.

(a) Em hãy bổ sung nền sân khấu mới từ kho các hình ảnh đã có của phần mềm, ví dụ, lấy hình ảnh sau (bedroom).

(b) Làm ẩn đi nhân vật chính là chú mèo.

(c) Nháy chọn biểu tượng sân khấu trong khung điều khiển phía dưới và kéo thả đoạn chương trình như trong hình ảnh sau.

Kéo thả 2 lệnh này từ cửa sổ lệnh của sân khấu.

Làm ẩn đi nhân vật Mèo con

Bổ sung thêm hình ảnh sân khấu này.

Nháy tại đây để vào cửa sổ lệnh của sân khấu.

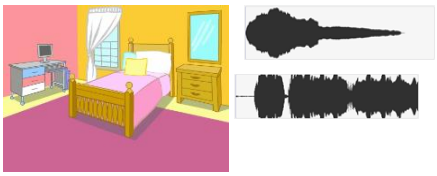
Chúng ta chú ý đến đoạn chương trình ngắn của sân khấu này.

Lệnh này nằm trong nhóm sự kiện (event) của sân khấu. Các lệnh bên dưới sẽ thực hiện mỗi khi nháy chuột lên sân

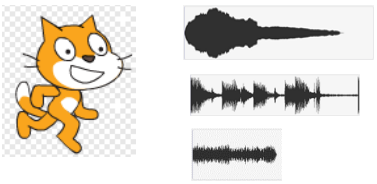
Bật âm thanh của sân khấu được ghi trong lệnh

Khi chạy chương trình này, em sẽ thấy mỗi khi nháy chuột lên sân khấu sẽ nghe được 1 âm thanh nhỏ phát ra.

Trên thực tế mỗi nhân vật và sân khấu nền của 1 chương trình đều có thể gán với một hay nhiều âm thanh khác nhau. Các âm thanh này rất đa dạng, có thể là lời nói, tiếng động, nhạc nền, bài hát. Các tệp âm thanh có thể dùng bao gồm các dạng wav, mp3, …..



Sân khấu có các âm thanh, nhạc nền riêng của mình.



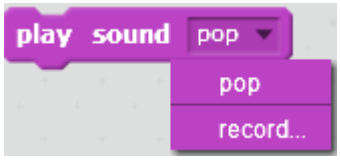
Nhân vật có âm thanh, nhạc, giọng nói riêng.

Số lượng âm thanh cho mỗi nhân vật là không hạn chế.

4. Thu âm trực tiếp âm thanh cho nhân vật

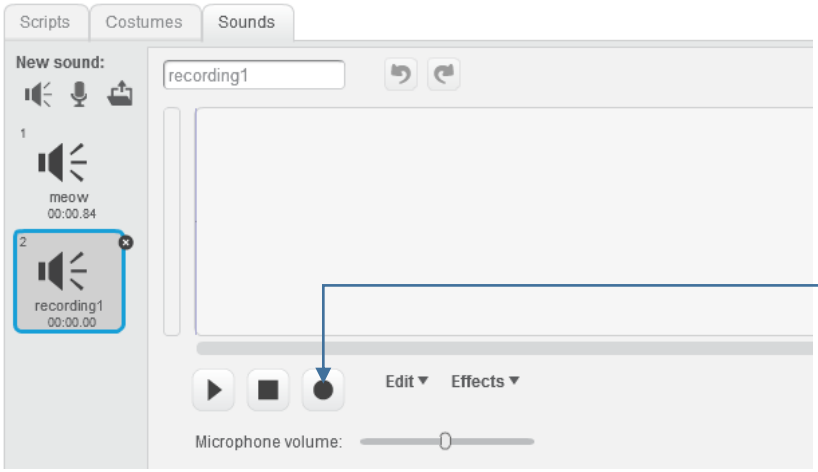
Trong hoạt động này, em sẽ thực hiện việc thu âm trực tiếp từ máy tính qua micro để tạo ra một âm thanh của nhân vật. Các bước thực hiện như sau.

(1) Lựa chọn **record** trong lệnh play sound của nhân vật.



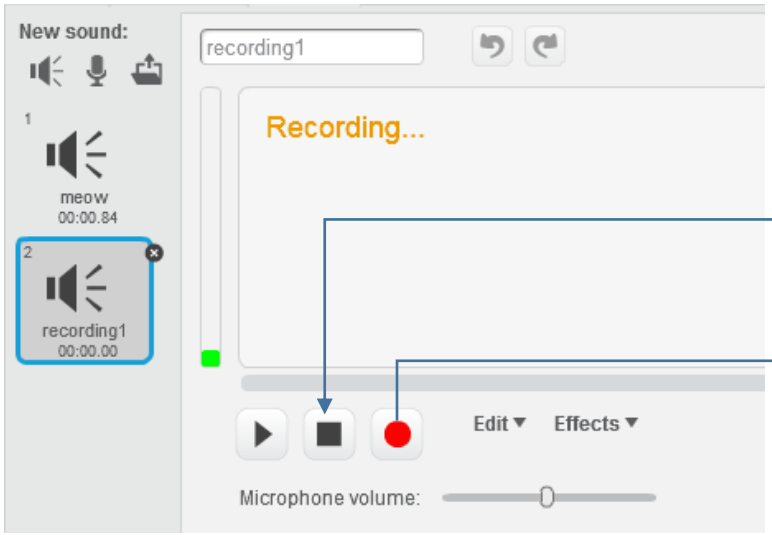
hoặc vào TAB **Sounds** của khung điều khiển và nháy lên biểu tượng micro  .

Cửa sổ sau sẽ xuất hiện để chuẩn bị thu âm trực tiếp.



Nháy nút này để bắt đầu thu âm.

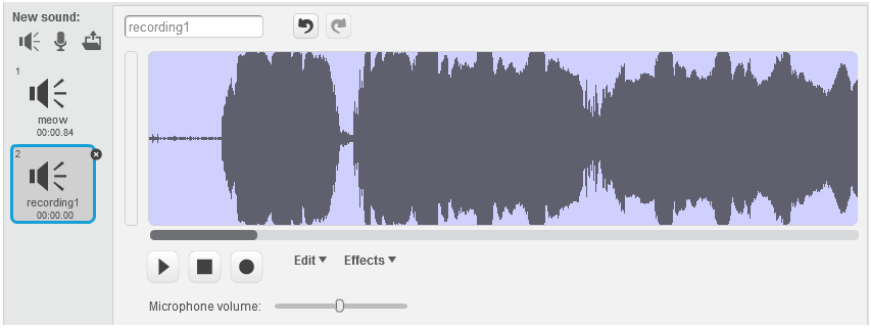
Trong quá trình đang thu âm, cửa sổ này có dạng như sau:



Nháy nút này để kết thúc thu âm.

Trong thời gian thu âm, nút này chuyển màu đỏ.

Khi kết thúc thu âm, âm thanh vừa được thu sẽ hiển thị trên cửa sổ như sau:



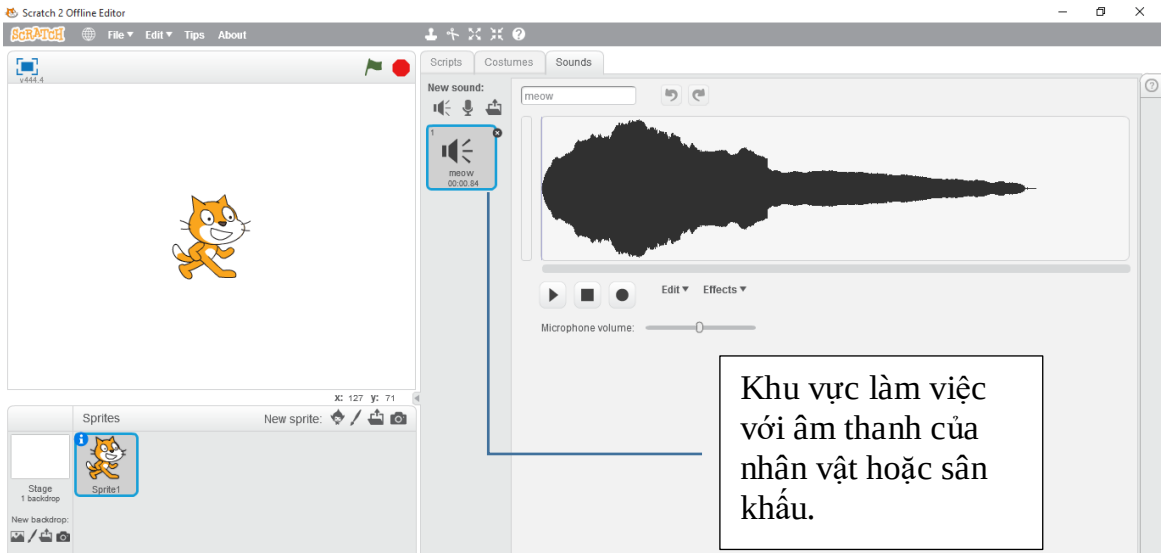
Bây giờ có thể thực hiện việc đặt tên cho âm thanh vừa thu âm.

5. Bổ sung âm thanh cho nhân vật và sân khấu

Trong hoạt động này, em sẽ thực hiện thao tác bổ sung âm thanh cho nhân vật (hoặc sân khấu) lấy từ 1 kho âm thanh có sẵn của phần mềm.

Cửa sổ làm việc với âm thanh của nhân vật hoặc sân khấu

Như chúng ta đã biết, trong Scratch, các nhân vật có thể nói, hát, phát ra tiếng nói của mình. Sân khấu cũng có thể có âm thanh nền. Trong màn hình Scratch nếu nháy lên nút Sound chúng ta sẽ vào chức năng cho phép bổ sung thêm âm thanh cho nhân vật. Cũng trên màn hình này sẽ hiện toàn bộ danh sách các âm thanh hiện đang có của nhân vật.

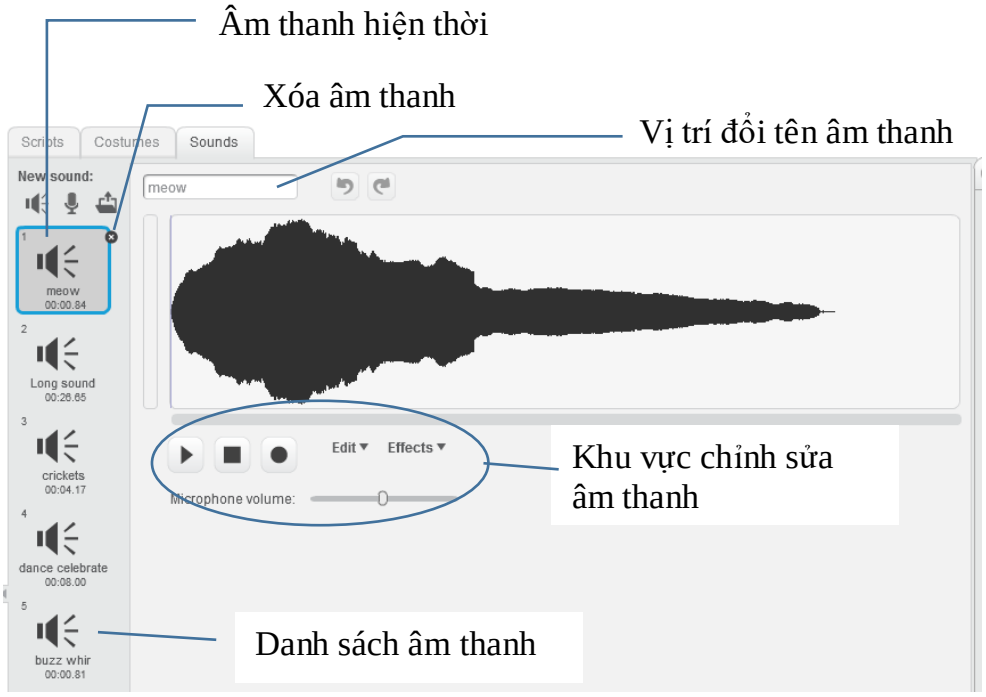


Với các âm thanh có sẵn chúng ta có thể thực hiện các thao tác sau:

- Chỉnh sửa âm thanh.

- Xóa âm thanh.

- Thay đổi tên âm thanh.

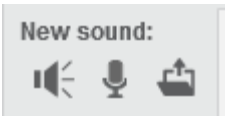


Mỗi nhân vật có thể đi kèm 1 hoặc nhiều âm thanh, em có thể bổ sung âm thanh từ các kho có sẵn, có thể thu âm trực tiếp hoặc lấy từ các tệp âm thanh trên máy tính.

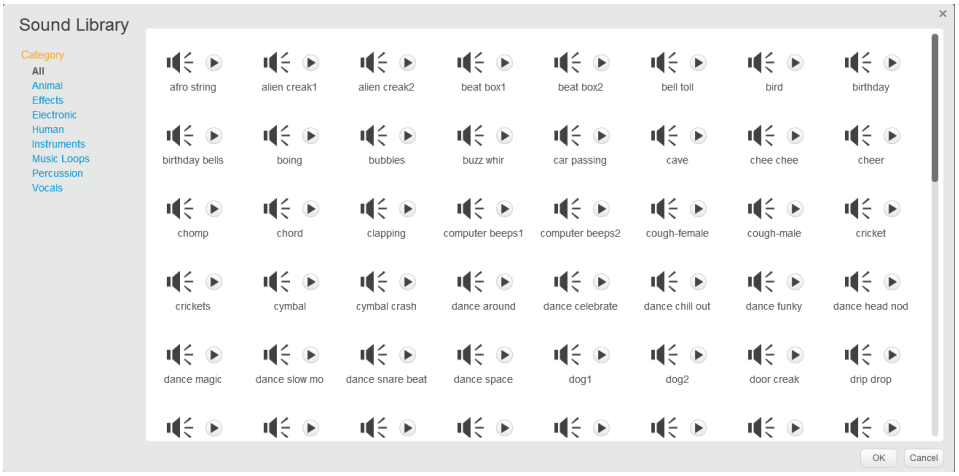
Em hãy thực hiện thao tác bổ sung âm thanh cho nhân vật bằng cách lấy từ 1 kho có sẵn.

Bổ sung âm thanh từ kho có sẵn

- Nháy vào nút hình loa  ở phía trên khung làm việc với âm thanh của nhân vật.

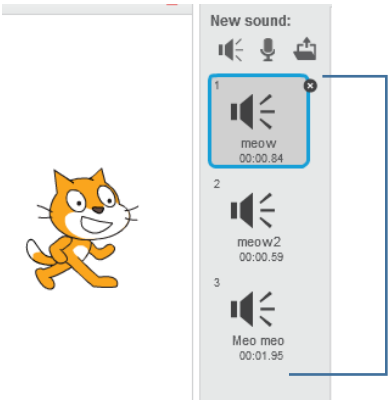


- Xuất hiện cửa sổ có danh sách các âm thanh.



- Em nháy chọn 1 âm thanh và nháy nút OK.

Em sẽ thấy âm thanh này đã được đưa vào khu vực âm thanh của nhân vật.

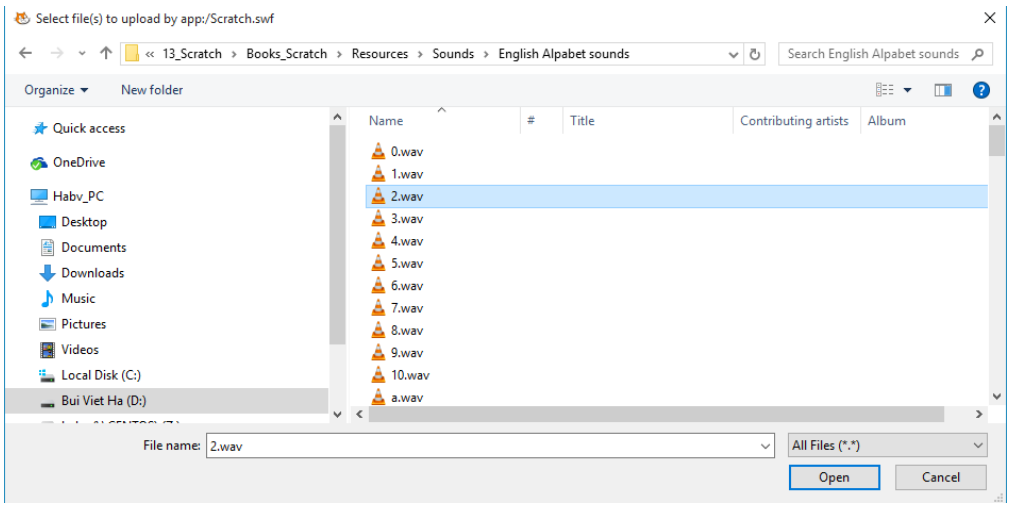


Danh sách các âm thanh của nhân vật hiển thị tại đây.

Bổ sung âm thanh từ tệp ngoài

- Nháy nút  .

- Xuất hiện hộp hội thoại mở tệp có dạng tương tự hình sau. Tìm kiếm tệp âm thanh ngoài.



- Chọn tệp âm thanh muốn bổ sung và nháy nút **Open**. Tệp âm thanh này sẽ được đưa vào danh sách âm thanh của nhân vật (hoặc sân khấu).

Câu hỏi và bài tập

1. Thiết lập nhân vật là thầy giáo, thu âm lời nói "Chào em" và viết đoạn chương trình thầy giáo nói và thể hiện dòng chữ "Chào em" trên màn hình.

2. Thiết lập nhân vật là học sinh, thu âm lời nói "Em chào thầy ạ" và viết đoạn chương trình em học sinh chào thầy giáo, nói và thể hiện dòng chữ "Em chào thầy" trên màn hình.

3. Thiết lập 2 nhân vật là thầy giáo và học sinh. Thu âm giọng nói của thầy "Chào em", giọng nói học sinh "Em chào thầy ạ" và thực hiện thiết kế chương trình như sau:

- Thầy nói "Chào em" và thể hiện lời chào trên màn hình.

- Sau 1 giây, học sinh sẽ chào lại "Em chào thầy ạ" và hiện dòng chữ trên màn hình.

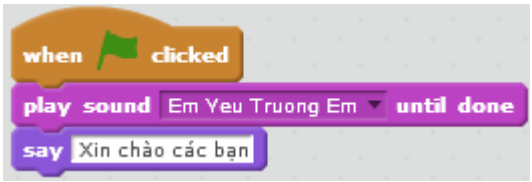
4. Thiết lập chương trình như sau:

- Nền sân khấu là hình ảnh một ngôi trường.

- Nhân vật chính là hai bạn học sinh nhỏ đang đi học.

- Em bổ sung bài hát "Em yêu trường em" cho nhân vật hai bạn nhỏ này.

- Cho chương trình chạy thì hát bài hát trên. Khi hát xong thì hai bạn nhỏ sẽ chào mọi người.



5. Thiết lập chương trình có 2 nhân vật là Mèo con và Chó cún như sau:

- Nhân vật: Mèo con, Chó cún.

- Em thu âm cho mèo con nói được "meo meo" và chó kêu được "gâu gâu".

- Lập trình cho mèo con và chó cún chạy xung quanh màn hình, vừa đi vừa kêu meo meo, gâu gâu và thể hiện các dòng chữ này trên màn hình.

6. Chỉnh sửa bài tập 4 theo hướng sau:

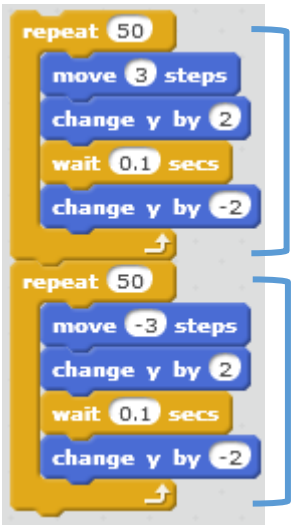
- Thay đổi nhân vật chính bằng 2 bạn nhỏ mới này.

Em yêu trường em.sb2

- Thay đổi lệnh điều khiển sự kiện, khi nháy chuột lên hình 2 bạn nhỏ thì 2 bạn mới cất tiếng hát bài **Em yêu trường em**. Sử dụng lệnh kích hoạt sự kiện sau:

7. Mở rộng bài tập 6 như sau: trong khi hát thì 2 bạn sẽ chạy, nhảy múa trong sân trường.

Đoạn chương trình mô tả việc nhảy múa có thể như sau:



Đoạn này mô tả nhân vật tiến về phía trước, mỗi lần bước 3 bước và nhảy lên, nhảy xuống 2 bước.

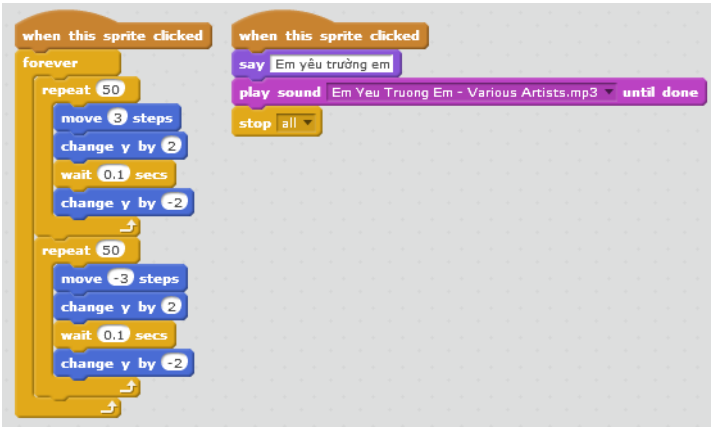
Đoạn này tương tự trên nhưng nhân vật đi theo hướng ngược lại, vừa đi vừa nhảy.

8. Hoàn thiện chương trình **Em yêu trường em** như sau:

- Nhân vật, 2 em học sinh sẽ nhảy múa suốt trong quá trình hát.

- Khi kết thúc bài hát thì dừng toàn bộ chương trình bởi lệnh 

.



Mở rộng

1. Em hãy viết 1 chương trình đơn giản có âm thanh với các yêu cầu sau.

- Nhân vật chính là 1 em bé.

- Em bé sẽ chạy vòng quanh sân khấu, mỗi bước chạy sẽ vọng lên tiếng thịch thịch của bàn chân.

2. Viết chương trình sau:

- Màn hình nền trắng, có 1 cái bút chì là nhân vật chính.

- Khi chạy chương trình, bút bắt đầu vẽ các hình đa giác đều trên màn hình, bắt đầu là tam giác, tứ giác và ngũ giác đều.

- Khi nháy chuột lên màn hình thì lời 1 bài hát quen thuộc vang lên. Em có thể tìm chọn một bài hát bất kỳ.

Bài 6. Chuyển động 2

Mục đích

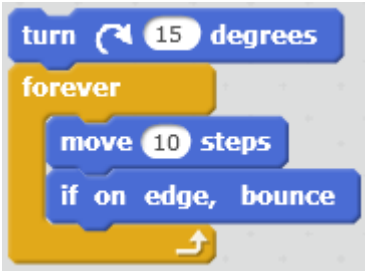
Học xong bài này, bạn sẽ biết:

- Các lệnh chuyển động phức tạp hơn trên màn hình.
- Giao tiếp đơn giản giữa nhân vật và máy tính.
- Sử dụng lệnh lặp đơn giản và lặp vô hạn.
- Bước đầu làm quen với lệnh điều khiển if … then …
- Làm việc với 2 hoặc nhiều nhân vật. Chương trình song song.

Bắt đầu

1. Quan sát chương trình sau, bạn có nhận xét gì về chuyển động của nhân vật.

2. Quan sát tiếp chương trình sau.



- Bạn có nhận xét gì về chuyển động của nhân vật? Có điểm gì giống và khác nhau so với chương trình trên.

Nội dung bài học

1. Điều khiển nhân vật khi chạm biên màn hình

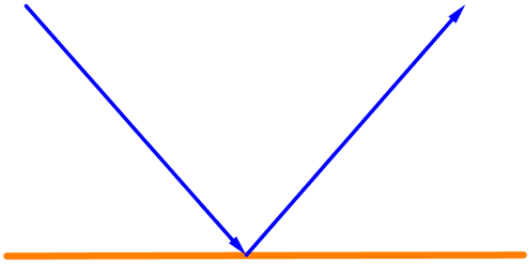
Có bao giờ em nghĩ rằng nếu chúng ta cho các nhân vật chuyển động ra khỏi phạm vi màn hình thì sẽ như thế nào không? Hoạt động này sẽ giúp em điều khiển nhân vật tránh được hiện tượng trên. Trong Scratch có 1 lệnh riêng như vậy, đó là lệnh

,

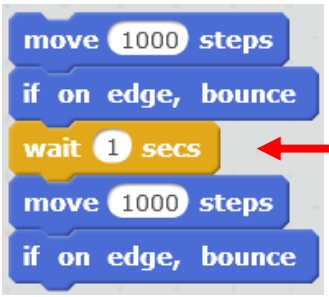
Lệnh này có tác dụng như sau: khi nhân vật chuyển động chạm cạnh sân khấu, nhân vật sẽ tự động dừng và quay lại theo hướng đối xứng gương với hướng ban đầu. Hướng ban đầu và hướng quay được chỉ ra trong hình bên.

Chú ý: lệnh này chỉ có trong Scratch và thường được đặt sau các chuyển động của nhân vật để điều khiển chuyển động này .

Em hãy thử đoạn chương trình sau để kiểm tra tác dụng của lệnh trên.



Khi nhân vật chuyển động gặp cạnh sân khấu, nhân vật sẽ dừng lại và quay hướng theo góc đối xứng với hướng lúc đi đến.



Lệnh **wait (1) secs** có tác dụng cho nhân vật dừng lại trong thời gian 1 giây để quan sát được dễ hơn.

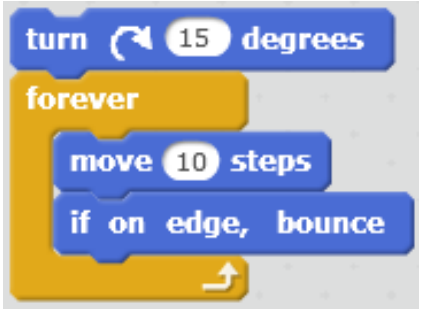
2. Lệnh lặp vô hạn

Chúng ta đã được làm quen với lệnh trong nhóm điều khiển dùng để làm cho 1 đoạn chương trình thực hiện lặp lại 1 số lần (lệnh repeat). Nếu chúng ta muốn 1 đoạn chương trình được lặp lại vô hạn lần thì cần sử dụng lệnh dưới đây.

Nhóm các lệnh nằm trong khung này sẽ được lặp lại vô hạn lần cho đến khi người sử dụng nháy nút đỏ trên màn hình hoặc có lệnh cho phép dừng vòng lặp hoặc chương trình.

Bây giờ thì em hãy thử lại và hiểu hai chương trình đã được nêu trong phần mở đầu, em có thể trả lời đươc câu hỏi đặt ra.

Nhân vật chuyển động liên tục, vô hạn dọc theo các cạnh của 1 hình vuông.



Nhân vật chuyển động liên tục, vô hạn, mỗi khi gặp cạnh sân khấu thì quay lại và đi tiếp.

3. Hội thoại giữa 2 nhân vật, chương trình song song

Trong hoạt động này chúng ta cùng thiết kế 1 chương trình đơn giản sau.

Nhân vật: Mèo và Cánh cụt.


Nội dung chương trình cần thực hiện.

```
Mèo (trông thấy Chim) chào
"Xin chào bạn Cánh cụt".
Chờ trong 3 giây.
Cánh cụt (thấy Mèo chào) chào "Chào Mèo con".
```

Em hãy phân tích yêu cầu của đầu bài và viết ra các công việc cần thực hiện của Mèo và Cánh cụt.

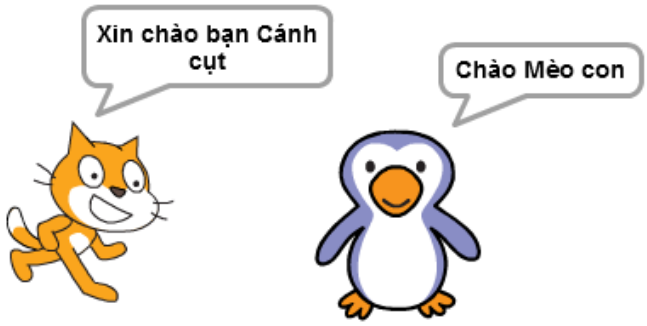
Công việc của Mèo	**Công việc của Cánh cụt**

Em cần thực hiện các bước sau:

- Bổ sung thêm nhân vật chim cánh cụt như hình trên và đổi tên 2 nhân vật thành "Mèo" và "Cánh cụt".

- Trong cửa sổ lệnh của Mèo và Cánh cụt, em viết 2 chương trình riêng biệt cho Mèo và Cánh cụt như hình dưới đây. Chú ý: hai chương trình sẽ thực hiện đồng thời, song song, nên cần để câu chào của Mèo trong 5 giây, còn Cánh cụt thì cần cần chờ đợi 3 giây đúng theo yêu cầu, sau đó mời chào Mèo.

- Kết quả thực hiện chương trình.



Em hãy bổ sung thêm các lệnh để mở rộng yêu cầu của bài toán trên.

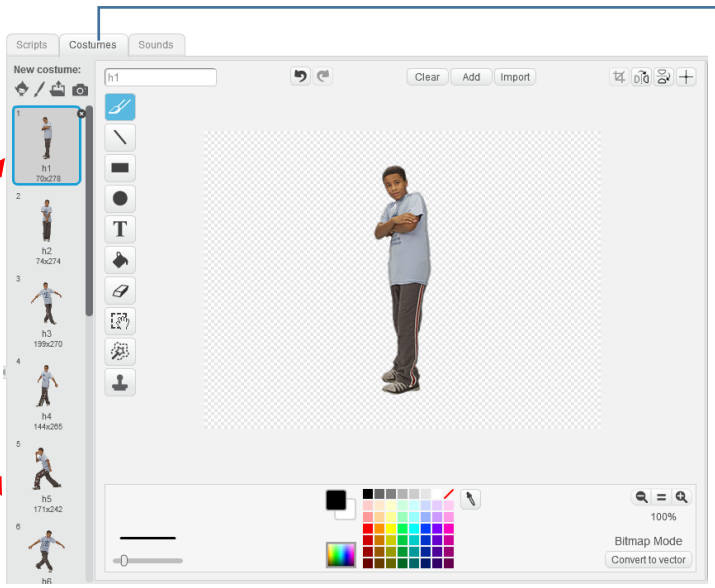
```
Mèo (trông thấy Chim) chào "Xin chào bạn Cánh cụt".
Đồng thời Mèo tiến 10 bước về phía Cánh cụt.
Chờ trong 3 giây.
Cánh cụt quay lại phía Mèo.
Cánh cụt chào "Chào Mèo con".
```

4. Chuyển động bằng thay đổi trang phục

Để mô tả chuyển động của nhân vật, ngoài các lệnh cho phép nhân vật dịch chuyển vị trí trên màn hình, chúng ta còn có thể sử dụng hiệu ứng thay đổi hình ảnh nhân vật.

Mỗi nhân vật sẽ có thể có nhiều hình ảnh minh họa, được gọi là trang phục. Nút Costume cho phép em bổ sung và chỉnh sửa các hình ảnh trang phục này. Ví dụ nhân vật Hip-hop sau có đến 12 bộ trang phục. Như vậy kết hợp thay đổi trang phục và dịch chuyển trên màn hình sẽ tạo ra các hình ảnh chuyển động thực của nhân vật. Phim hoạt hình cũng được thiết kế dựa trên nguyên tắc này.

Các trang phục của nhân vật. Scratch cho phép tạo ra nhiều hình ảnh - trang phục khác nhau của 1 nhân vật.

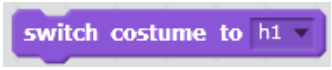


Cửa sổ vẽ, chỉnh sửa trang phục của nhân vật.

Trong nhóm lệnh **Thể hiện (Look)** có 2 lệnh điều khiển trang phục sau.

Lệnh 

có tác dụng cho nhân vật chuyển sang trang phục tiếp theo trong danh sách.

Lệnh 

có tác dụng cho nhân vật chuyển sang trang phục được chọn trong danh sách.

Bây giờ em hãy thiết kế đoạn chương trình cho nhân vật Hip-hop của chúng ta vừa dịch chuyển trên màn hình (dùng lệnh move), vừa thay đổi trang phục (dùng lệnh next costume).

Chương trình 1.

- Vòng lặp: 10 lần.
- Nhân vật thay đổi trang phục sau mỗi 10 bước.
- Sau khi thay trang phục thì dừng chương trình 1 giây.

Quan sát chương trình trên, em có nhận xét gì về chuyển động của chú bé Hip-hop?

Bây giờ chúng ta sẽ làm tốt lên chương trình trên bằng cách tăng số vòng lặp lên 100, nhân vật sẽ thay đổi trang phục chỉ sau 5 bước và thời gian chờ chỉ là 0.5 giây. Hiệu ứng chuyển động sẽ tốt lên.

Chương trình 2.

- Vòng lặp: 100 lần.
- Nhân vật thay đổi trang phục sau mỗi 5 bước.
- Sau khi thay trang phục thì nghỉ nhanh 0.5 giây.

5. Lệnh điều khiển có điều kiện

Lệnh **if ... then**

Trong hoạt động này em sẽ làm quen với 1 lệnh quan trọng của Scratch, lệnh điều khiển có điều kiện if … then (nếu …. thì). Đây là 1 lệnh, 1 tư duy quan trọng của việc điều khiển máy tính làm việc.

Trên thực tế chúng ta thường rất hay gặp tình huống mà hành động được quyết định tùy theo các điều kiện cụ thể. Ví dụ:

- **Nếu** trời nắng **thì** tôi sẽ đi chơi.
- **Nếu** bạn gặp không làm được bài **thì** tôi sẽ giúp bạn.

Mệnh đề trên được thể hiện tổng quát thành câu lệnh sau:

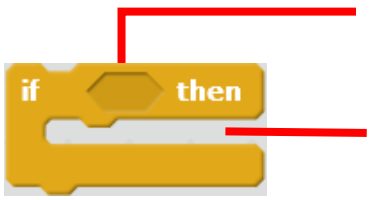
Nếu <điều kiện> **thì** <hành động>

Ở đây <điều kiện> là 1 biểu thức, hoặc mệnh đề luôn có giá trị Đúng hoặc Sai. Các mệnh đề, biểu thức chỉ mang ý nghĩa đúng, sai được gọi là biểu thức logic.

Ví dụ các biểu thức logic.

1 = 2 Nhận giá trị sai. 15 > 3.4 Nhận giá trị đúng.

Lệnh thực hiện lệnh theo điều kiện (logic) là lệnh If …. then, lệnh này nằm trong nhóm lệnh Điều khiển (Control) có màu nâu.



Điều kiện thực hiện lệnh mô tả bằng 1 biểu thức hay giá trị dạng **Đúng / Sai** (logic). Biểu thức logic thể hiện hình trám nhọn ở 2 đầu.

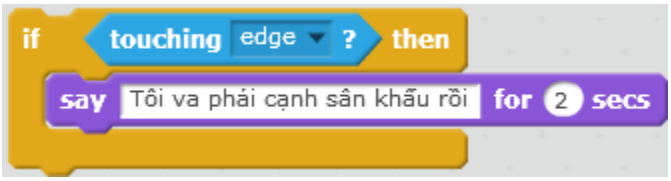
Nhóm lệnh trong vị trí này sẽ được thực hiện khi biểu thức logic trên là Đúng

Trong Scratch có 1 số lệnh là các mệnh đề logic, chỉ có giá trị đúng và sai. Chúng ta cùng tìm hiểu 2 lệnh này trong nhóm lệnh Cảm biến (Sensing).

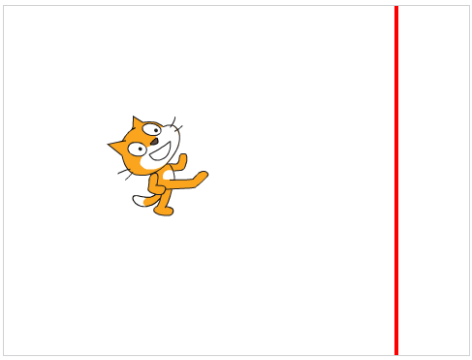
Lệnh này đóng vai trò 1 điều kiện logic. Lệnh chỉ trả lại giá trị Đúng nếu nhân vật hiện thời va chạm với 1 đối tượng khác chỉ ra trong lệnh (nhân vật khác hoặc cạnh sân khấu).

Lệnh này chỉ trả lại giá trị Đúng nếu nhân vật hiện thời tiếp xúc với màu sắc được chỉ ra ngay trong lệnh.

Ví dụ đoạn chương trình sau kiểm tra xem nhân vật có chạm vào cạnh sân khấu hay không, nếu đúng thì sẽ thông báo "Tôi va phải cạnh sân khấu rồi".



Bây giờ em hãy thực hiện chương trình với nhiệm vụ sau.



Vẫn là bài toán cho chú Mèo chuyển động tự do, vô hạn trên màn hình, nếu gặp cạnh sân khấu thì quay lại (xem ví dụ mục 1). Tuy nhiên bổ sung thêm yêu cầu sau:

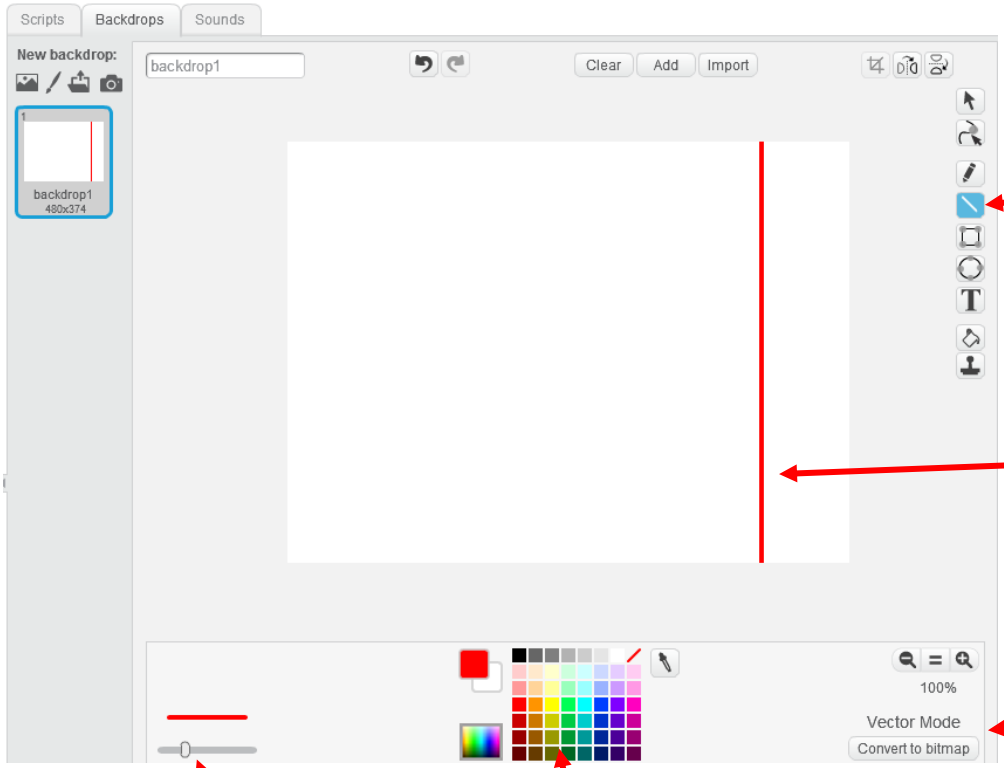
Trên sân khấu vẽ thêm 1 vạch đứng màu đỏ.

Mèo chỉ được phép chuyển động tự do ở vùng bên trái vạch đỏ. Nếu chạm vạch đỏ sẽ lập tức quay sáng trái với góc nghiêng 45 độ.

Em hãy cùng thực hiện chương trình này theo các bước sau.

1) Chỉnh sửa sân khấu, bổ sung thêm 1 đường thẳng đứng màu đỏ.

Em hãy nháy chọn biểu tượng sân khấu, nháy tiếp nút Backdrops, khi đó màn hình chỉnh sửa hình ảnh sân khấu như hình dưới đây. Em hãy thực hiện tiếp theo sơ đồ dưới đây.



2. Lựa chọn công cụ kẻ đoạn thẳng

5. Kẻ 1 đoạn thẳng đứng tại đây.

1. Lựa chọn kiểu hình ảnh Vector Mode

4. Lựa chọn độ dày của đường kẻ

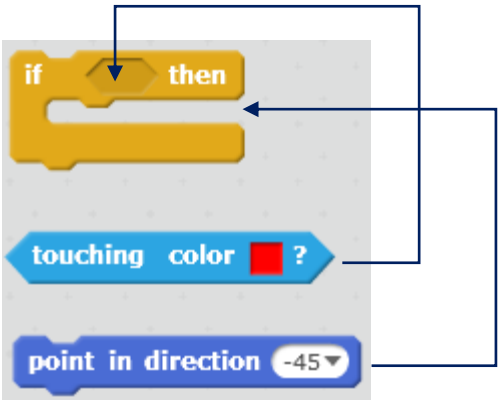
3. Lựa chọn màu đỏ cho công cụ vẽ

2) Bây giờ em thiết kế lại chương trình đã có trong mục 1.

3) Trong vòng lặp này, em cần đưa vào thêm 1 số lệnh để điều khiển nhân vật nếu va chạm với đoạn thẳng đứng màu đỏ thì phải quay lại. Đoạn chương trình này có dạng sau:

```
Nếu <nhân vật tiếp xúc với màu đỏ> thì
     Quay lại theo hướng 45 độ
```

Em kéo thả ra màn hình lệnh của nhân vật các lệnh sau, sau đó điền thông số và kéo thả chúng vào đúng vị trí.

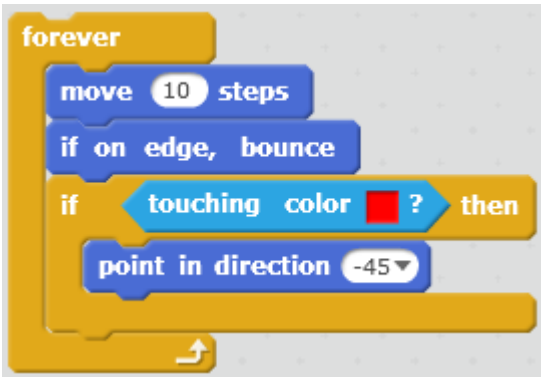


Để thiết lập màu đúng của lệnh - biểu thức logic này em làm như sau:

- Nháy chuột lên vị trí ô vuông chọn màu của lệnh.

- Di chuyển chuột ra sân khấu và nháy lên đoạn thẳng đứng. Màu của đoạn thẳng này sẽ tự động điền vào biểu thức của lệnh này.

4) Kết quả em thu được đoạn chương trình sau:



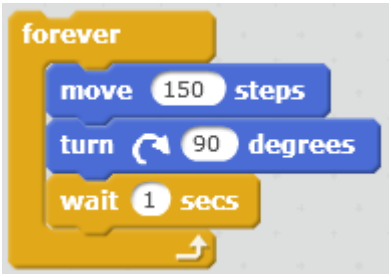
5) Chạy chương trình, kiểm tra kết quả và ghi lại thành tệp tin *.sb2.

Câu hỏi:

Em có thể sửa lệnh **point in direction -45º** để thiết lập kết quả nhân vật sẽ phản xạ tại vạch đứng màu đỏ theo hướng đối xứng với hướng đi đến không?

Câu hỏi và bài tập

1. Em hãy mô tả kết quả của đoạn chương trình sau:



2. Mở rộng đoạn chương trình trong mục 4, cho cậu bé Hip-hop chuyển động vô hạn từ trái qua phải, gặp cạnh sân khấu thì quay lại và cứ như vậy tiếp tục.

3. Các biểu thức logic sau là đúng hay sai?

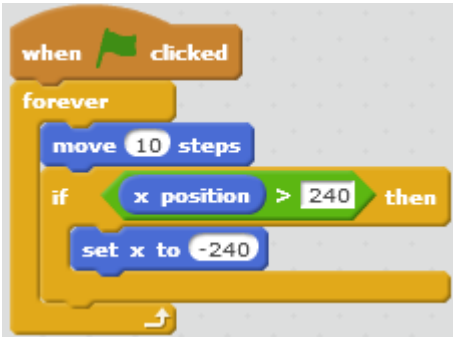
A. $4 < 5$.

B. $5 + 7 - 2 > 10$.

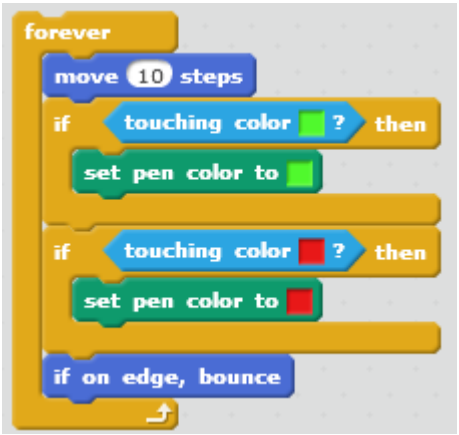
C. $2^3 = 3^2$

D. $\sqrt{2} > 1.4$

4. Em hãy chạy đoạn chương trình sau và giải thích kết quả chương trình.

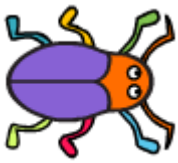


5. Đoạn chương trình sau có ý nghĩa gì?

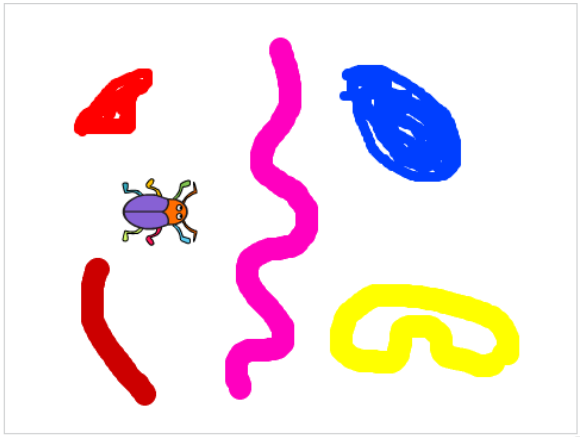


6. Em hãy viết chương trình, trong đó sử dụng lệnh cảm biến:

Chương trình có 1 nhân vật là con cánh cam:

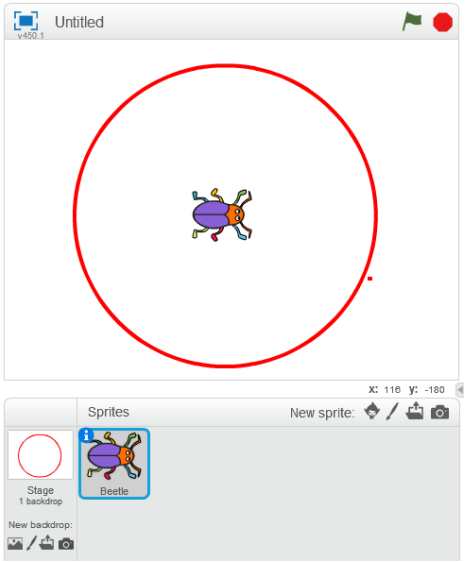


Trên màn hình có nhiều đám đất với các màu sắc khác nhau, ví dụ như hình sau:



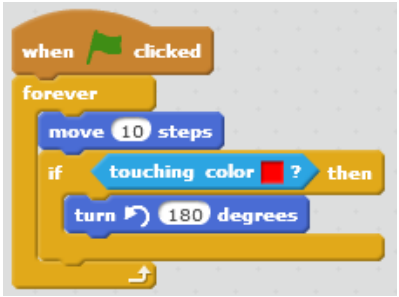
Chương trình như sau: người chơi sẽ điều khiển con cánh cam chuyển động trên màn hình theo các 4 hướng: →, ←, ↑, ↓. Nếu cánh cam đi qua đám đất có màu sắc nào thì sau đó nó sẽ để lại vết có màu đó.

7. Quan sát hình sau.

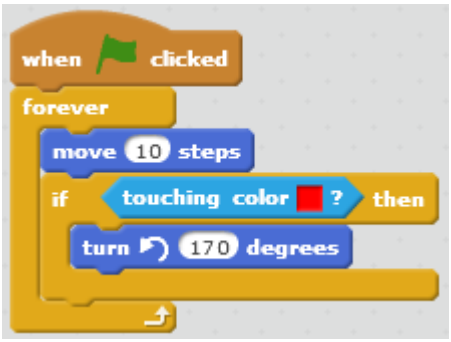


Chương trình sau sẽ điều khiển con cánh cam làm gì?

Chuyen dong cam bien 1.sb2



Chương trình sau sẽ điều khiển con cánh cam làm gì? So sánh với chương trình trước.



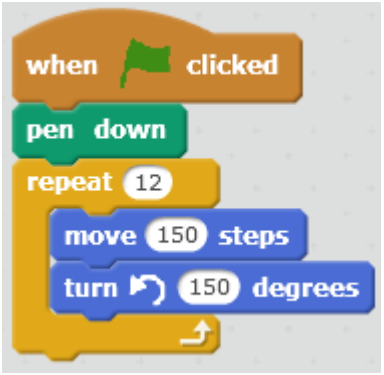
8. Viết chương trình nhỏ mô tả yêu cầu sau:

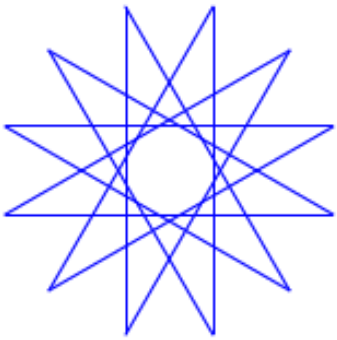
Nhân vật chính là con mèo chuyển động vô hạn trên màn hình, nếu gặp cạnh màn hình thì bật lại. Nếu người dùng gõ 1 phím Space thì mèo sẽ kêu meo meo.

9. Viết chương trình có 2 nhân vật Mèo và Chó. Thu âm các âm thanh meo meo và gâu gâu cho mèo và chó.

Lập trình thực hiện yêu cầu sau: nếu nháy chuột lên mèo thì mèo kêu meo meo, nếu nháy chuột lên chó thì chó kêu gâu gâu.

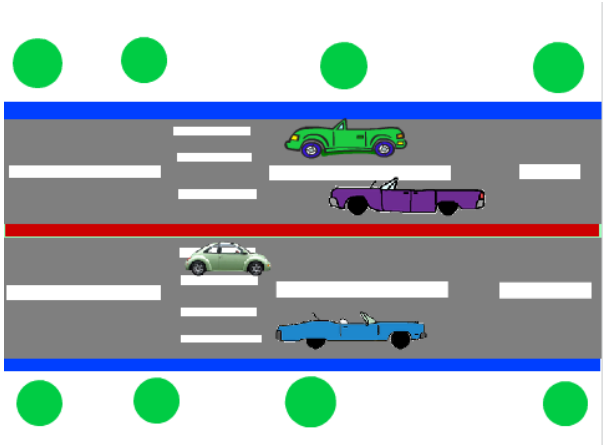
10. Thực hiện đoạn chương trình sau với chế độ pendown, hãy quan sát và giải thích kết quả chương trình.





11. Thiết kế chương trình đơn giản mô phỏng giao thông đường phố như hình sau:

Giao thong.sb2



Yêu cầu mô tả chuyển động của các ô tô như sau: khi đi hết 1 chiều của sân khấu sẽ xuất hiện trở lại từ phía đối diện.

Mở rộng

1. Thiết kế 1 trò chơi đơn giản

Em hãy thiết kế 1 trò chơi đơn giản "Thả bóng" theo các yêu cầu sau:

Nhân vật: quả bóng và thanh ngang.

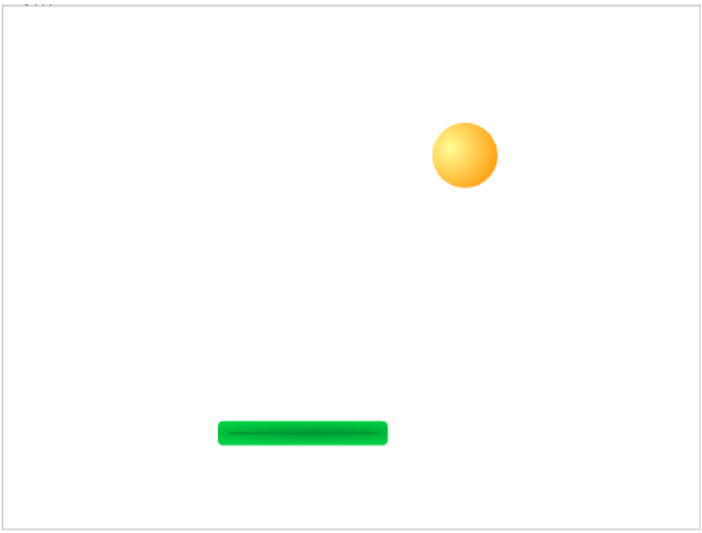
Quả bóng sẽ rơi tự do từ trên trần xuống. Nếu gặp cạnh sân khấu sẽ tự động bật lại và chuyển động liên tục.

Nếu quả bóng chạm thanh ngang, nó sẽ bật lên 1 góc 15 độ và chuyển động ngược lên trên.

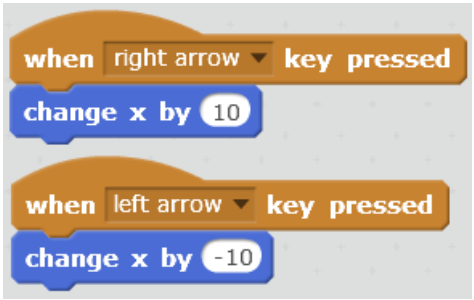
Người dùng điều khiển dùng các phím phải, trái để điều khiển quả bóng sao cho bóng không bị rơi xuống phía dưới.

Gợi ý cách thực hiện.

Bước 1. Thiết lập 2 nhân vật **Bóng** và **Thanh ngang** trên sân khấu. Các nhân vật này có thể chọn từ kho các hình ảnh có sẵn của phần mềm.



Bước 2. Thiết lập chương trình điều khiển thanh ngang bằng các lệnh sự kiện khi bấm phím phải, trái.



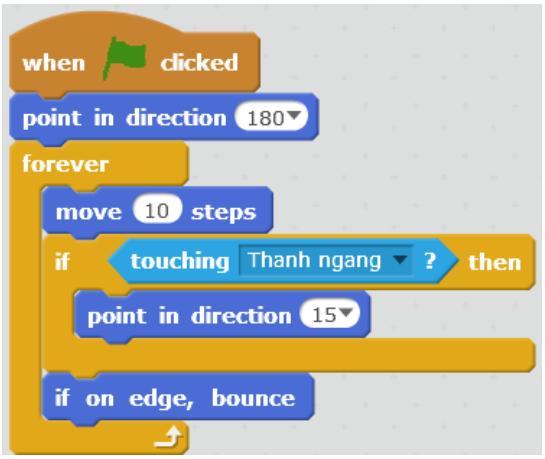
Bước 3. Thiết lập chương trình cho Bóng. Có thể hình dung công việc của Bóng như sau:

Quay hướng xuống dưới.

Lặp vô hạn lần

```
dịch chuyển 10 bước (về phía trước)
Nếu tiếp xúc với màu đỏ thì
     quay hướng lên trên 15 độ
Nếu gặp cạnh sân khấu, quay lại.
```

Đoạn chương trình có dạng sau.



2. Lảm tốt lên trò chơi này

Em hãy làm tốt trò chơi này bằng cách bổ sung yêu cầu sau:

Nếu để Bóng rơi xuống phía dưới thì dừng trò chơi.

Chú ý sử dụng các gợi ý sau:

- Có thể vẽ thêm 1 đường ngang phía dưới để kiểm soát khi Bóng rơi xuống.
- Sử dụng lệnh **Stop All** từ nhóm lệnh Điều khiển để dừng chương trình.

3. Viết chương trình mô phỏng **Mèo con chạy** như sau:

Yêu cầu điều khiển con Mèo chạy vòng quanh gian phòng như trong hình ảnh.

Meo con chay.sb2

Mèo phải chạy theo các đường thẳng màu đỏ, xuất phát từ vị trí ban đầu, đi 1 vòng và quay trở về vị trí cũ. Yêu cầu có thêm mô phỏng về khoảng cách: mèo càng ở xa càng bé đi, càng lại gần càng to ra.

Bài 7. Vẽ hình 2

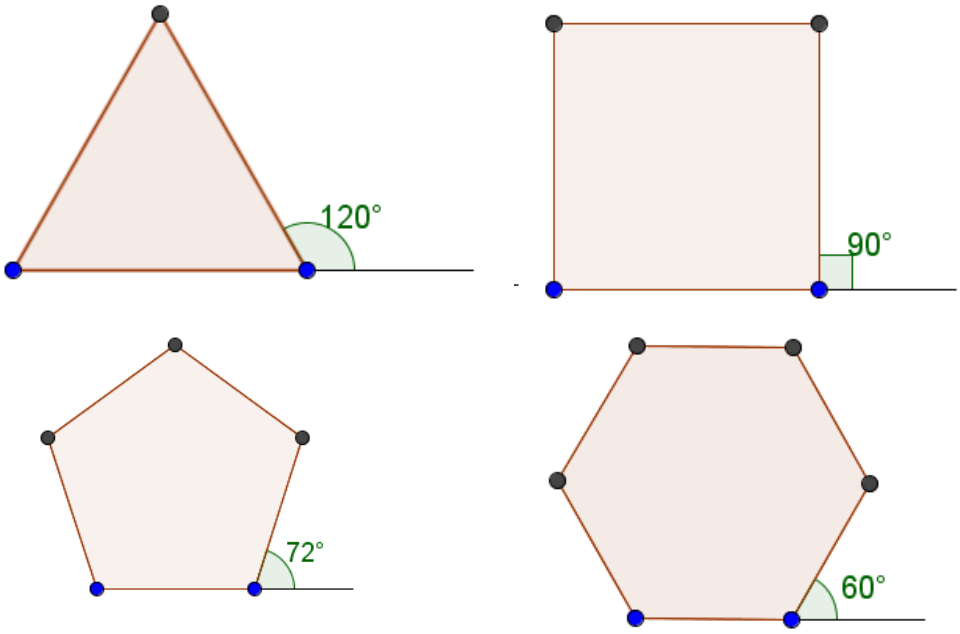
Mục đích

Học xong bài này, bạn có thể:

- Thực hiện vẽ hình phức tạp hơn: hình tròn, đa giác đều, xoắn ốc, hình nghệ thuật.
- Kết hợp màu sắc để vẽ hình nghệ thuật.
- Lệnh lặp cố định, lặp có điều khiển.

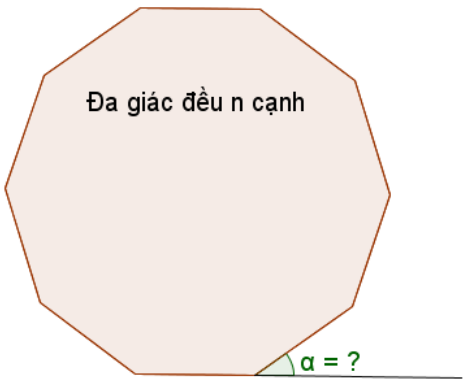
Bắt đầu

1. Quan sát các hình đa giác đều sau.



- Em có nhận xét gì về số đo các góc trong và ngoài của các đa giác này, nếu liên hệ với số cạnh của đa giác?

2. Em có thể tổng quát cách tính góc α với trường hợp đa giác đều n cạnh như hình dưới đây?

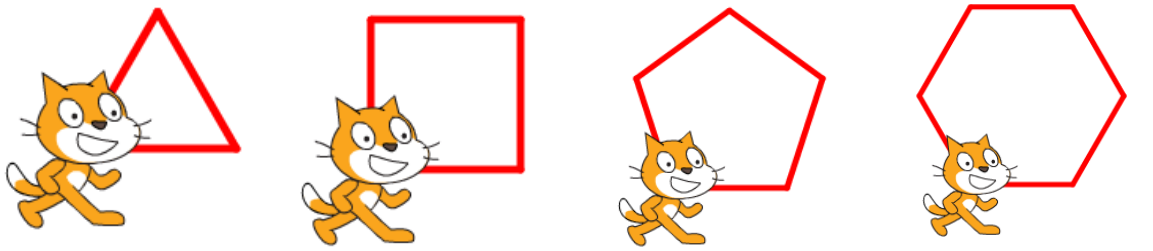


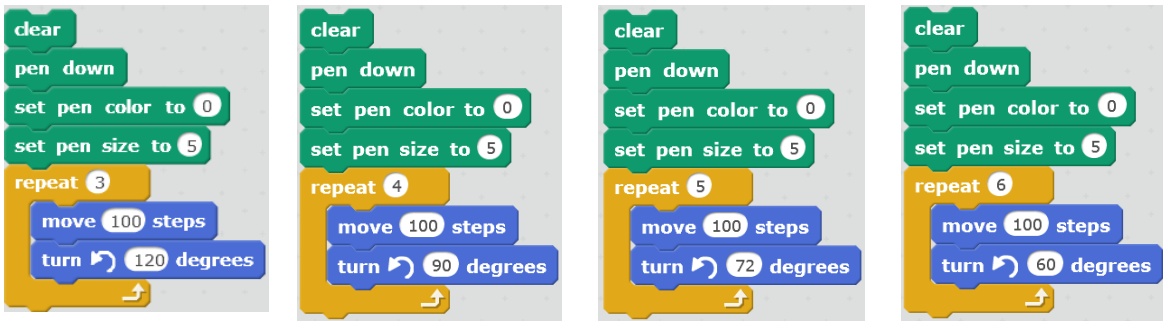
3. Em hãy trình bày theo cách riêng của em cách vẽ hình đa giác đều bằng Scratch.

Nội dung bài học

1. Thực hành vẽ 1 số hình đa giác đều

Dựa trên quan sát trong phần mở đầu, em hãy viết các chương trình đơn giản sau để vẽ các hình tam giác, tứ giác, ngũ giác và lục giác đều.

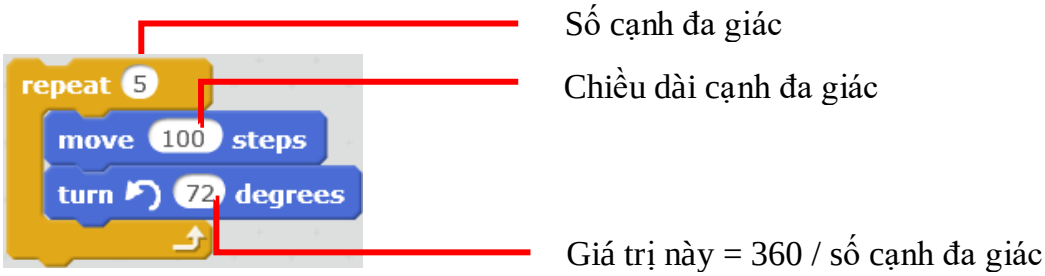




Em có nhận xét gì về các chương trình trên?

2. Tìm qui luật của cách vẽ đa giác đều

Nếu quan sát kỹ hơn các chương trình trên chúng ta sẽ tìm ra được quy luật (hay còn gọi là thuật toán) của các chương trình này.

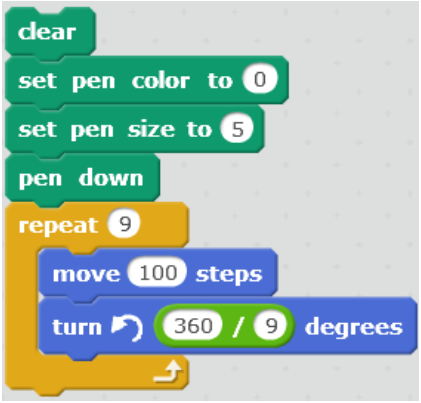


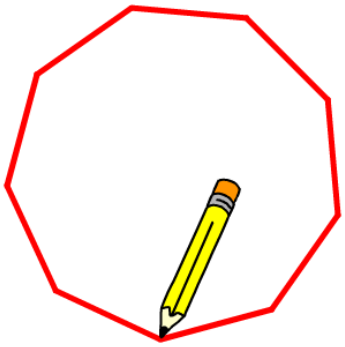
Chú ý: nếu số cạnh lớn thì có thể điều chỉnh độ dài cạnh để hình vẽ không vượt quá giới hạn sân khấu.

Em hãy áp dụng qui luật trên để vẽ các đa giác đều 9 và 11 cạnh, sử dụng nhân vật là bút chì.

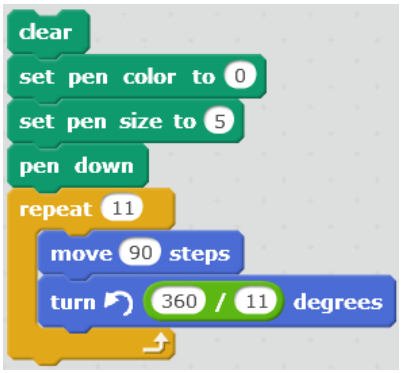
Để thực hiện phép chia 360 / <số cạnh> em hãy dùng lện (toán tử) phép chia  từ nhóm lệnh **Tính toán (Operators)** màu xanh lá cây. Ví dụ nếu vẽ đa giác đều 9 cạnh chúng ta nhập các giá trị 360 và 9 vào toán tử này như sau:  .

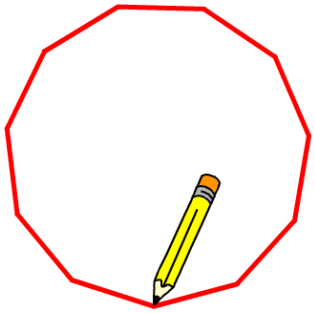
Với hình 9 cạnh, độ dài cạnh 100, em phải điều chỉnh vị trí ban đầu của bút chì sao cho hình nằm trong khung màn hình.





Với hình 11 cạnh, bắt buộc phải làm ngắn lại độ dài cạnh đa giác.

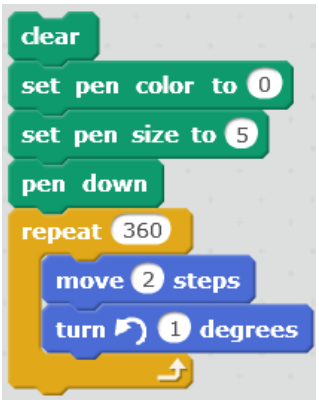


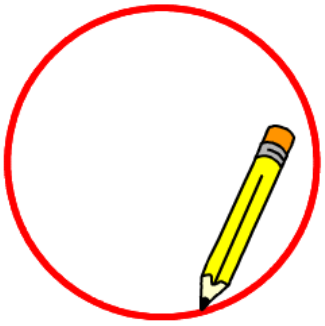


Dựa trên qui luật này, em hãy viết chương trình vẽ các hình đa giác đều có 7, 8, 10 cạnh.

3. Vẽ hình tròn

Để vẽ được hình tròn chúng ta hình dung như 1 đa giác đều n cạnh nhưng với n khá lớn. Em hãy thực hiện chương trình vẽ hình tròn theo cách vẽ 1 đa giác đều với số cạnh lớn.





Trong chương trình này, em dùng thuật toán đối với vẽ đa giác đều với số cạnh = 360, độ dài cạnh = 2.

Chú ý:

1. Em có thể không cần thiết lập số cạnh lớn như vậy. Có thể chỉ cần vẽ với số cạnh nhỏ hơn, ví dụ 180, hoặc thậm chí 90, vẫn thu được hình tròn.

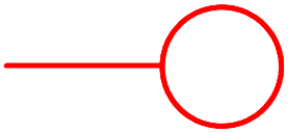
2. Muốn vẽ vòng tròn nhỏ em có thể chọn độ dành cạnh là các số bé hơn 1, ví dụ 0.5.

Luyện tập 1: hãy vẽ hình sau

Gợi ý: Trước khi thực hiện đoạn chương trình vẽ vòng tròn, cần vẽ 1 đoạn thẳng từ vị trí hiện thời của nhân vật đến vị trí bắt dầu đường tròn. Thực hiện công việc này bằng lệnh  thông thường. Ví dụ có thể sử dụng đoạn chương trình sau.

Lệnh này sẽ vẽ đoạn thẳng nối với đường tròn được vẽ trong lệnh lặp

Luyện tập 2: vẽ hình sau.



Gợi ý: tương tự trên, trước khi bắt đầu vẽ hình tròn, cho nhân vật quay 90 độ theo chiều kim đồng hồ. Ví dụ chương trình sau.

Lệnh này sẽ làm cho bút vẽ quay xuống dưới để vẽ được vòng tròn như trong hình vẽ yêu cầu.

Luyện tập 3: vẽ các hình sau.

Cách thực hiện tương tự trên.

Hình thứ nhất

Hình thứ 2

4. Vẽ hình phức tạp với vòng lặp lồng nhau

Chúng ta đã được làm quen và hiểu ý nghĩa của các lệnh lặp (lệnh **repeat, forever**). Các lệnh này cho phép thể hiện ngắn gọn, dễ hiểu hơn khi chương trình có các lệnh hoặc đoạn lệnh lặp lại. Trong Scratch có 2 lệnh lặp: lặp với số lần cố định **repeat** và lặp vô hạn lần **forever**.

Trong chương trình các vòng lặp có thể được lồng vào nhau, tức là vòng lặp này nằm trong 1 vòng lặp khác. Có rất nhiều ví dụ thực tế mô tả các vòng lặp lồng nhau như vậy.

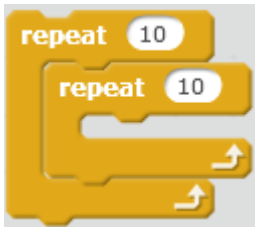
Ví dụ: khi xét các công việc hàng ngày em phải làm trong 1 tuần lễ (tứ thứ 2 đến thứ 7), có công việc đi học. Khi đó mô tả các công việc này trong 1 tuần, chúng ta có 1 vòng lặp đầu tiên:

```
Lặp lại 6 ngày trong tuần
     Sáng dậy sớm, ăn sáng
     Đi học
```

Bây giờ nếu xét trong cả 1 học kỳ bao gồm 17 tuần, mỗi tuần đều lặp lại lịch học trên, chúng ta sẽ có mô hình vòng lặp lồng nhau:

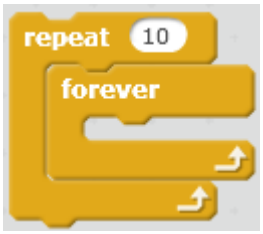
```
Lặp lại 17 tuần của học kỳ
     Lặp lại 6 ngày trong tuần
          Sáng dậy sớm, ăn sáng
          Đi học
```

Các vòng lặp có thể lồng nhau theo nhiều mức và kiểu khác nhau. Ví dụ trường hợp vòng lặp 1 mức ta có các trường hợp sau:



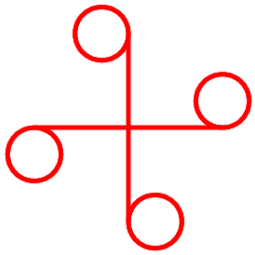
2 vòng lặp repeat lồng nhau

Vòng lặp repeat nằm trong vòng lặp forever



Vòng lặp vô hạn forever nằm trong vòng lặp repeat

Bây giờ em hãy thiết lập chương trình thể hiện hình vẽ dưới đây.

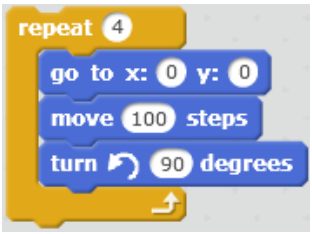


Phân tích bài toán này chúng ta thấy:

Vòng lặp ngoài cùng: thực hiện 4 lần việc vẽ 1 đoạn thẳng xuất phát từ tâm điểm của hình.

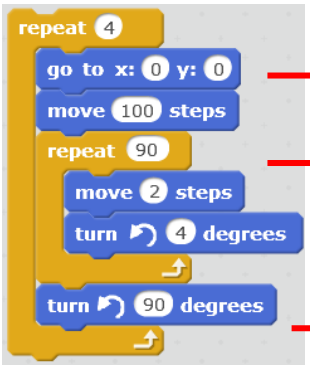
Vòng lặp bên trong: vẽ hình tròn.

Ở vòng lặp bên ngoài chúng ta có đoạn lệnh:



Ở vòng lặp trong là đoạn lệnh vẽ đường tròn mà chúng ta đã biết:

Ghép 2 vòng lặp này lồng vào nhau chúng ta thu được.



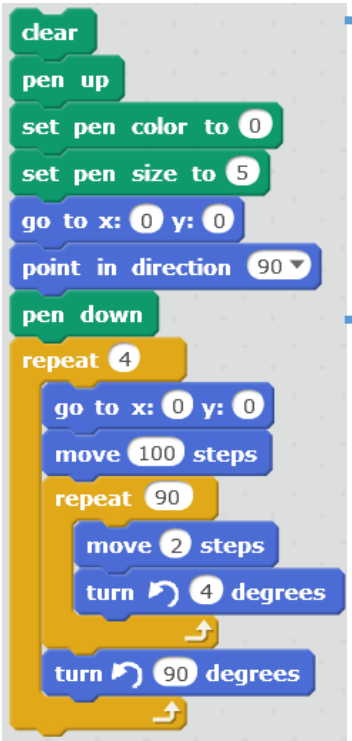
Vòng lặp ngoài, thực hiện 4 lần việc vẽ 1 đoạn thẳng từ vị trí tâm (0,0) độ dài 100.

Vòng lặp trong vẽ đường tròn với 90 lần lặp.

Sau khi vẽ vòng tròn, quay hướng bút vẽ 90 độ sang trái để thực hiện vòng lặp ngoài tiếp theo.

Chú ý trước khi thực hiện vòng lặp đầu tiên em cần đưa bút vẽ về vị trí ban đầu là tâm của hình:

Kết quả chương trình như sau:

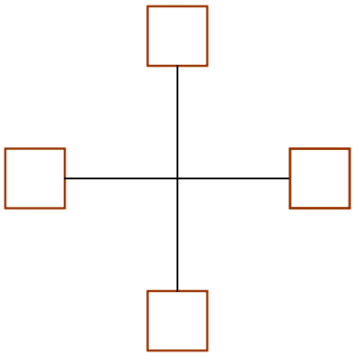


Thiết lập chế độ vẽ, chuyển đầu bút về tâm của sân khấu.

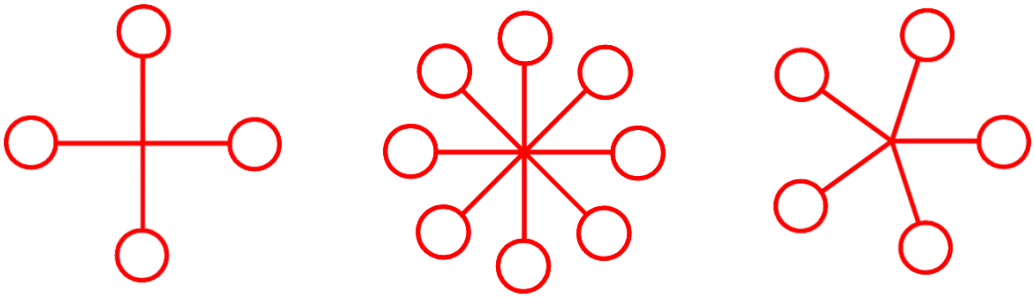
Đoạn chương trình vẽ chính bao gồm 2 vòng lặp lồng nhau.

Câu hỏi và bài tập

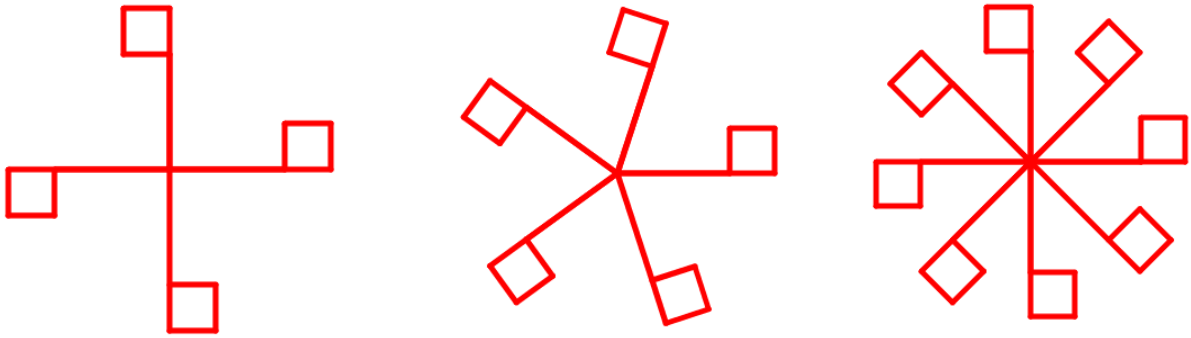
1. Viết chương trình vẽ hình sau:



2. Em hãy viết chương trình vẽ các hình sau:

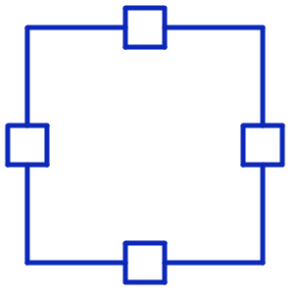


3. Viết chương trình vẽ các hình sau:



4. Viết lại các chương trình trên nhưng thể hiện mỗi nhánh 1 màu khác nhau.

5. Viết chương trình vẽ hình sau.



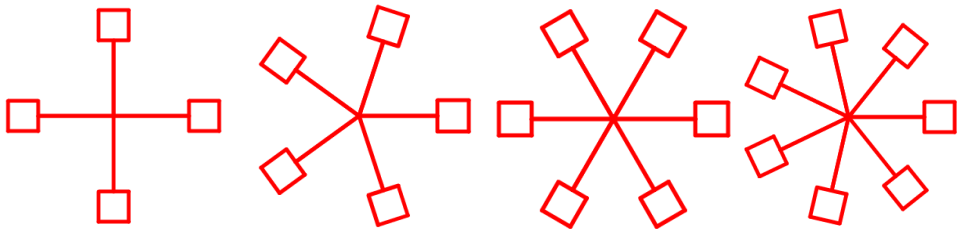
6. Viết chương trình mở rộng cho bài trên với yêu cầu tạo thêm 2 biến nhớ, **canh_dai** dùng để lưu độ dài cạnh hình vuông lớn và **canh_ngan** lưu độ dài cạnh hình vuông nhỏ. Ví dụ thiết lập **canh_dai** = 200, **canh_ngan** = 60.

7. Viết chương trình tạo ra 3 nhân vật là hình các loại bút chì khác nhau, mỗi bút chì vẽ 1 vòng tròn có bán kính khác nhau trên màn hình.

8. Viết chương trình vẽ 2 vòng tròn đồng tâm trên màn hình.

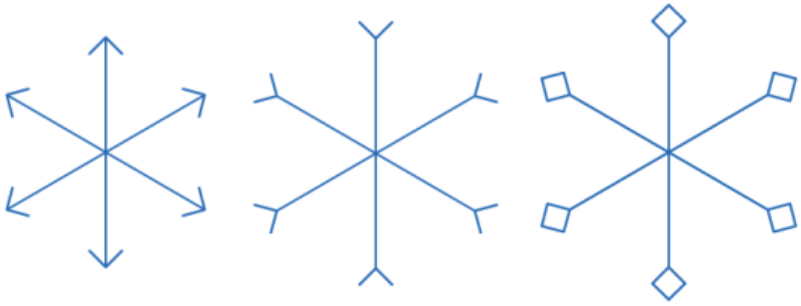
9. Viết chương trình vẽ 2 hình vuông đồng tâm trên màn hình.

10. Viết chương trình vẽ các hình sau:

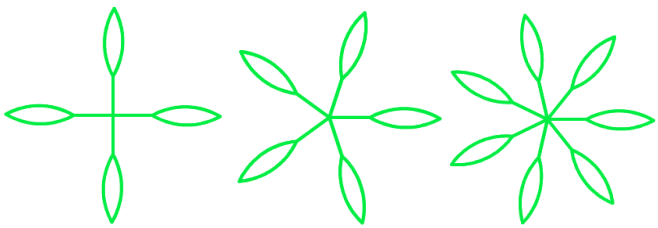


11. Viết chương trình vẽ các hình sau:

12. Viết chương trình vẽ các hình sau:

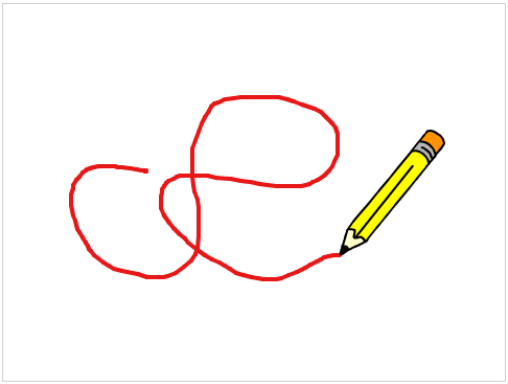


13. Viết chương trình vẽ bông hoa như sau:



Mở rộng

1. Thiết kế chương trình **Bút vẽ tự do** như sau:



Trên màn hình khi di chuyển chuột, bút vẽ luôn đi theo con trỏ chuột.

- Bấm phím Space: bắt đầu chế độ vẽ.
- Nháy chuột: kết thúc chế độ vẽ.
- Bấm phím x: chuyển màu vẽ là Xanh.
- Bấm phím d: chuyển màu vẽ là Đỏ.

2. Viết chương trình vẽ lá cờ quốc kỳ Việt Nam.

3. Viết chương trình vẽ vòng tròn biết trước tọa độ tâm và độ dài bán kính R.

Bài 8. Âm thanh 2

Mục đích

Học xong bài này, bạn sẽ biết:

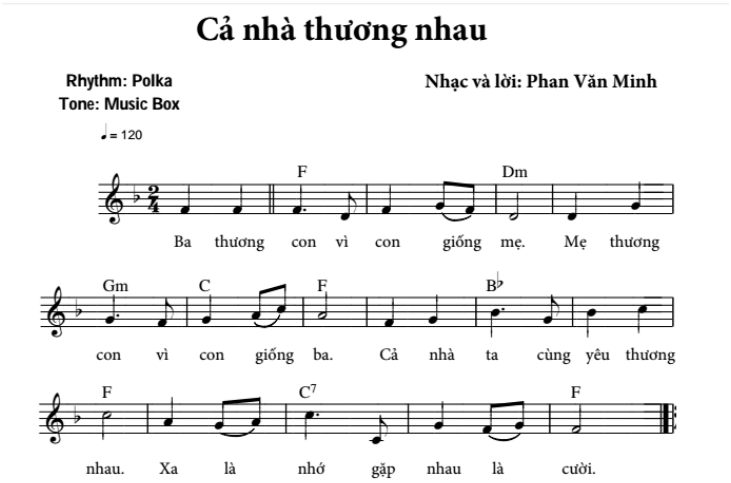
- Kết nối âm thanh với nhân vật. Lập trình cho nhân vật nói và thể hiện lời nói bằng chữ và âm thanh.

- Điều khiển các lệnh tạo âm thanh, đánh trống và chơi nhạc trên nền sân khấu.

- Điều khiển nhân vật nhảy múa theo nhạc nền.

Bắt đầu

Em hãy quan sát bản nhạc sau. Em có hiểu cách ghi các nốt nhạc, cao độ, trường độ, nhịp, phách của bản nhạc này không?



Trong bài học này chúng ta sẽ cùng nhau tìm hiểu các lệnh để có thể đánh được đúng các nốt của bản nhạc này.

Nội dung bài học

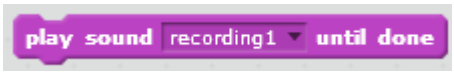
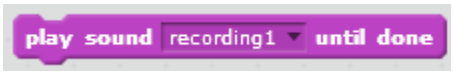
1. Kết nối âm thanh với nhân vật, nền sân khấu. Ứng dụng kể chuyện

Như chúng ta đã biết, mỗi nhân vật trong Scratch có thể có nhiều trang phục và âm thanh đi kèm, số lượng các âm thanh hay trang phục này là không hạn chế.

Tương tự, sân khấu cũng có thể có nhiều hình ảnh nền sân khấu (backdrop) và âm thanh kèm theo với số lượng không hạn chế.

Các lệnh làm việc với trang phục, âm thanh, nền sân khấu trong Scratch bao gồm:

Ý nghĩa lệnh	Nhân vật	Sân khấu
Chuyển trang phục / nền sân khấu tương ứng.	switch costume to costume1	switch backdrop to backdrop1

Ý nghĩa lệnh	Nhân vật	Sân khấu
Chuyển sang trang phục / nền sân khấu tiếp theo trong danh sách.		
Bật âm thanh tương ứng và thực hiện ngay các lệnh tiếp theo		
Bật âm thanh tương ứng và chờ xong mới thực hiện các lệnh tiếp theo		

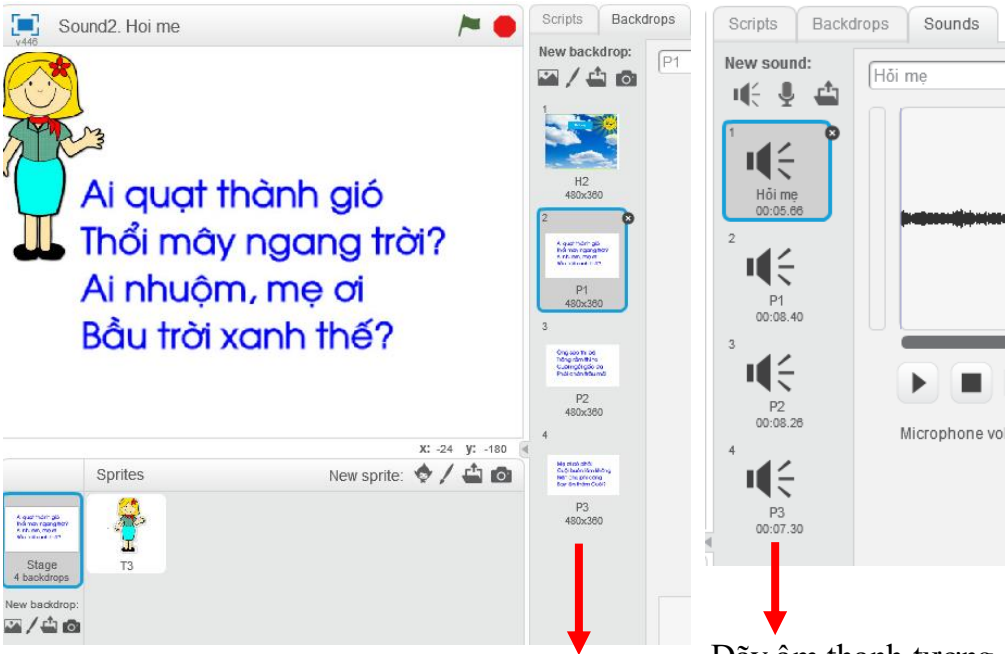
Chú ý quan trọng: Dãy các trang phục, nền sân khấu hay âm thanh còn có thể xác định bằng số thứ tự hoặc tên của chúng. Ví dụ nếu chúng ta sử dụng các biến nhớ  để lưu số thứ tự hoặc  để lưu tên của các trang phục, nền sân khấu hay âm thanh này thì các lệnh trên có thể sử dụng các biến nhớ trên để khai thác.

Chuyển nền sân khấu sang nền có số thứ tự trong danh sách là **stt**.
Chuyển nền sân khấu sang nền có tên trong danh sách là **name**.
Chuyển trang phục của nhân vật sang trang phục có số thứ tự trong danh sách là **stt**.
Chuyển trang phục của nhân vật sang nền có tên trong danh sách là **name**.
Bật âm thanh có số thứ tự trong danh sách là **stt**.

Bật âm thanh có tên trong danh sách là **name**.

Thiết lập ứng dụng kể chuyện

Chúng ta sẽ thiết lập một ứng dụng đơn giản có kết nối âm thanh với hình ảnh. Giả sử muốn thực hiện một bài kể chuyện bằng giọng nói theo tranh, ví dụ, bài thơ "hỏi mẹ". Em hãy thiết kế mô hình chương trình bao gồm 4 hình ảnh nền sân khấu và 4 tệp âm thanh kể chuyện tương ứng. Chú ý mỗi tệp âm thanh sẽ là giọng kể mô tả hình ảnh tương ứng.



Dãy hình nền sân khấu chính của câu chuyện muốn đưa vào ứng dụng này.

Dãy âm thanh tương ứng với dãy hình ảnh nền sâu khấu.

Đoạn chương trình chính được xây dựng trên của sổ lệnh của sân khấu. Cần tạo thêm biến nhớ **stt** để lưu số thứ tự của các tệp âm thanh. Chương trình bao gồm 1 vòng lặp (số vòng lặp = số lượng hình ảnh nền). Tại mỗi bước, cần thực hiện lệnh

dùng để bật nghe âm thanh cho đến khi xong thì tự động tăng **stt** lên 1 và chuyển sang nền sâu khấu tiếp theo.

Thiết lập giá trị ban đầu cho stt = 1

Vòng lặp chính của chương trình. Số lần lặp = số lượng hình nền = số lượng âm thanh tương ứng.

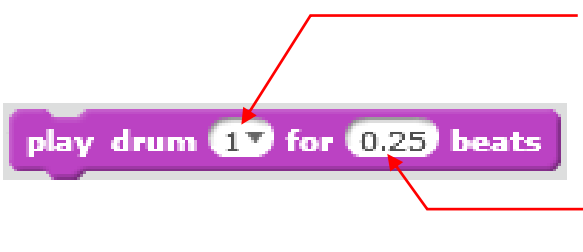
2. Trống và nhịp trống

Chắc các em đã được làm quen với các bộ trống trong dàn nhạc và vai trò của trống giữ nhịp trong các biểu diễn âm nhạc. Trong hoạt động này chúng ta sẽ làm quen với các lệnh trống của Scratch.

Bộ các công cụ trống hiện đại.

Lệnh gõ trống **play drum for beat**.

Lệnh này có tác dụng đánh 1 công cụ gõ (công cụ ghi trong tham số đầu tiên) và kéo dài trong khoảng thời gian tính theo nhịp (beat) trống, thời gian này ghi trong tham số thứ 2 của lệnh.



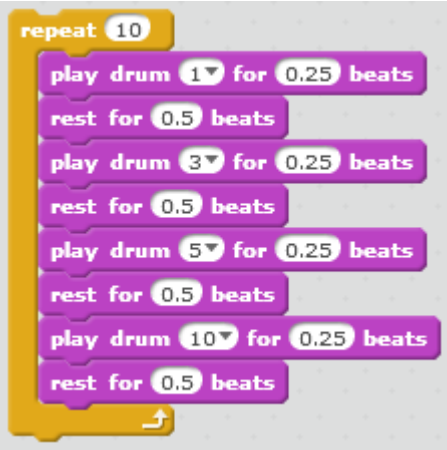
Ví dụ: lệnh 

sẽ đánh 1 lần trống nền (trống nền ) và đợi 1/2 nhịp trống tiếp theo. Ở đây nhịp trống được đo bởi giá trị **tempo** = số nhịp trống trong 1 phút. Ví dụ nếu temp = 60 thì nhịp trống sẽ là theo từng giây, mỗi giây 1 lần đánh trống.

Lệnh thiết lập nhịp trống là **set tempo to**.

Chú ý: Nhịp trống (beat) còn có ý nghĩa trong việc thể hiện trường độ nốt nhạc sẽ được học trong phần sau.

Em hãy soạn và chạy thử đoạn chương trình trống sau đây.

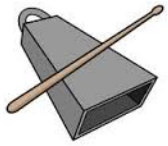


Chương trình bao gồm 10 vòng lặp, mỗi vòng lặp sẽ bao gồm 2 nhịp trống.

Nhịp 1: sẽ lần lượt đánh trống chính và thanh ngang 1/4 nhịp, xen giữa nghỉ 1/4 nhịp.

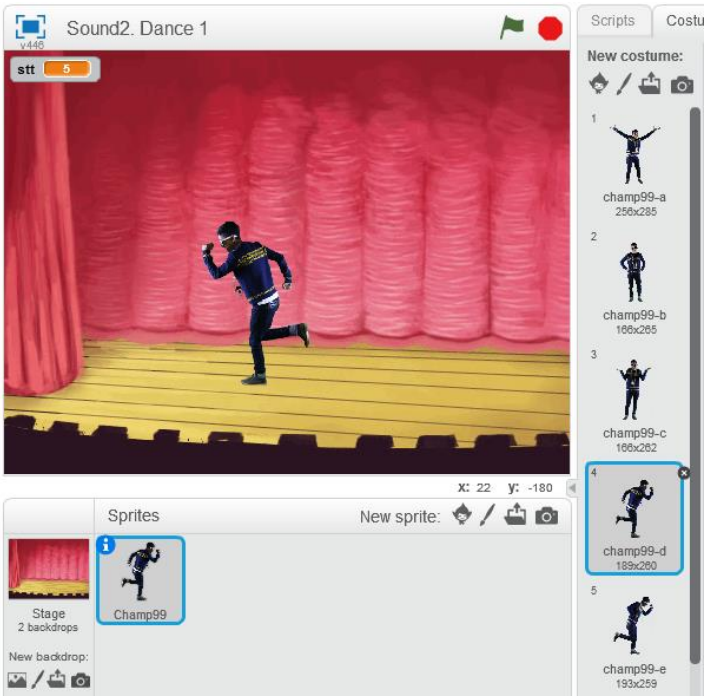
Nhịp 2: sẽ lần lượt đánh mặt mũ trên và trống gỗ 1/4 nhịp, xen giữa nghỉ 1/4 nhịp.

Bảng sau cho chúng ta nhận dạng thêm 1 số hình ảnh công cụ trong giàn trống.

Snare Drum		Tambourine		Bongo	
Bass Drum		Hand Clap		Conga	
Side Stick		Claves		Cabasa	
Crash Cymbal		Wood Block		Guiro	
Open Hi-Hat		Cowbell		Vibraslap	
Closed Hi-Hat		Triangle		Open Cuica	

3. Em bé nhảy và múa theo nhịp trống.

Em hãy thiết lập 1 chương trình đơn giản như thể hiện trong hình sau.



Chương trình sử dụng các hình ảnh trang phục khác nhau của nhân vật để mô tả hoạt động nhảy múa của nhân vật.

4. Đánh nốt nhạc

Trong hoạt động này em sẽ được học cách Scratch điều khiển các công cụ chơi nhạc. Các em chú ý đến các yếu tố sau:

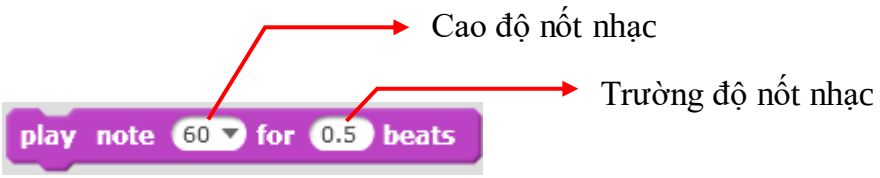
1. Công cụ chơi nhạc.

2. Độ cao nốt nhạc.

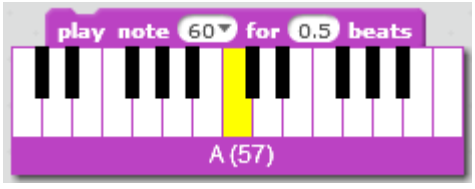
3. Trường độ nốt nhạc.

Độ cao của nốt nhạc

Lệnh phát nốt nhạc chính trong Scratch là **play note for beats**. Lệnh này có tác dụng phát 1 nốt nhạc đúng cao độ và trường độ được ghi ngay trong tham số của lệnh. Trường độ được tính theo nhịp trống (beat).



Khi lựa chọn cao độ nốt nhạc, em sẽ thấy hiện hình ảnh mô tả phím đàn Piano như hình dưới đây. Mỗi nốt nhạc có 1 ký tự chữ cái thể hiện và giá trị cao độ của nót nhạc.

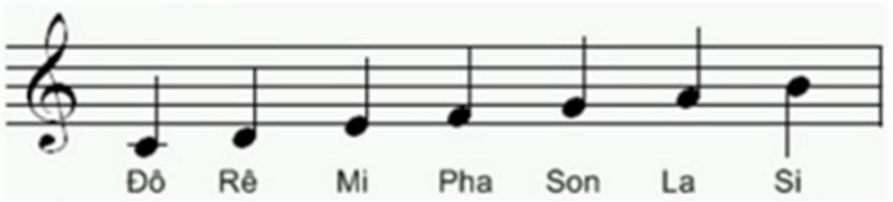


Khi nhấn chuột vào vị trí muốn nhập cao độ nột nhạc sẽ xuất hiện hình ảnh bên thể hiện phím đàn cho người dùng dễ quan sát và nhập. Bên cạnh giá trị số còn có ký hiệu các nốt nhạc

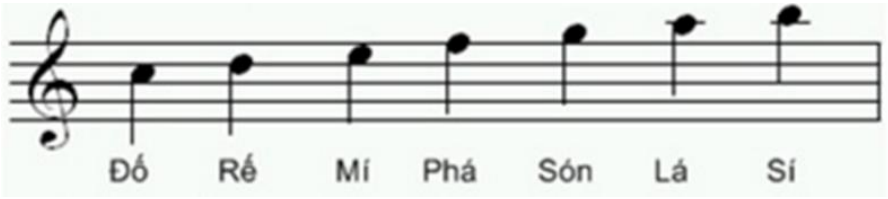
Bảng ký hiệu các nốt nhạc:

Đô	Rê	Mi	Pha	Son	La	Si	Đô
C	D	E	F	G	A	B	C

Ví dụ các giá trị nốt chính của bản nhạc được mô tả trong hình sau:



Note	48	50	52	53	55	57	59



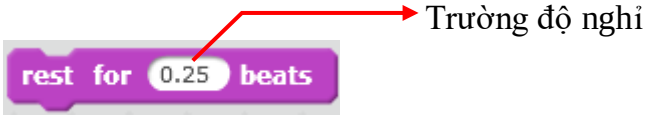
Note	60	62	64	65	67	69	71

Trường độ của nốt nhạc

Trường độ chỉ ra độ dài theo thời gian của nốt nhạc. Trong Scratch, đơn vị đo trường độ là **beat** (nhịp trống) mà chúng ta đã biết. Giá trị trường độ được thể hiện trong bảng sau:

Ký hiệu nốt nhạc							
Trường độ (beat)	**4**	**2**	**1**	**0.5**	**0.25**	**0.125**	**0.0625**

Với lệnh nghỉ (khoảng lặng của bản nhạc) chúng ta sử dụng lệnh sau:

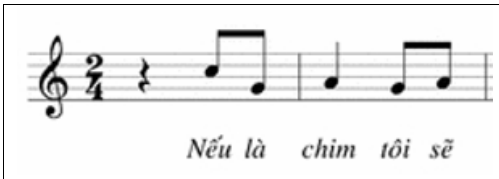
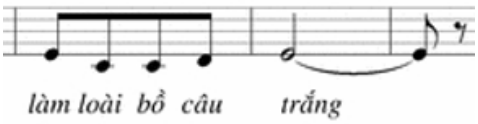
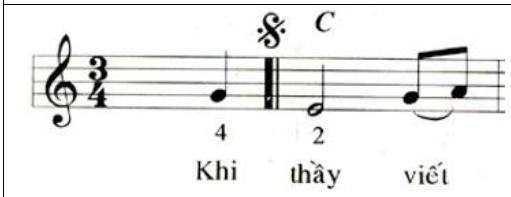
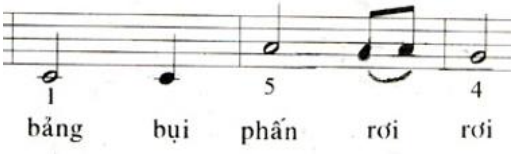


Bảng các ký hiêuj khoảng lặng trong khuôn nhạc như sau:

Ký hiệu nghỉ trên nốt nhạc					
beat	**4**	**2**	**1**	**0.5**	**0.25**

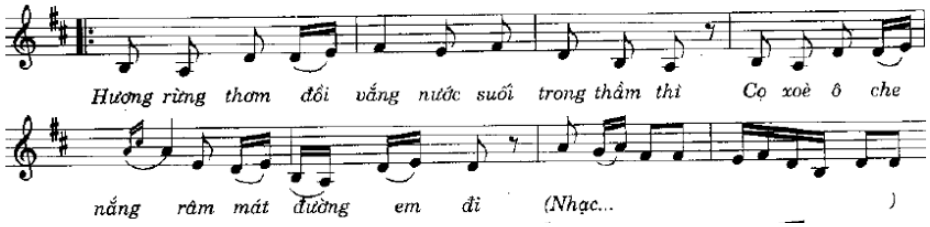
Bây giờ nhìn vào 1 bản nhạc đơn giản, chúng ta có thể viết đoạn chương trình mô phỏng bản nhạc đó.

Em quan sát bản nhạc và thực hiện các lệnh Scratch tương ứng trong các ô trống của bảng sau:

Bây giờ chúng ta sẽ cùng thiết lập các đoạn nhạc dài hơn, dựa trên các bài hát quen thuộc mà em đã biết.

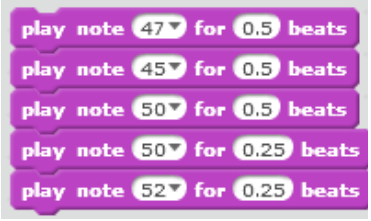
Ví dụ 1: một đoạn nhạc bài hát **Đi học** (nhạc Bùi Đình Thảo, thơ Minh Chính).



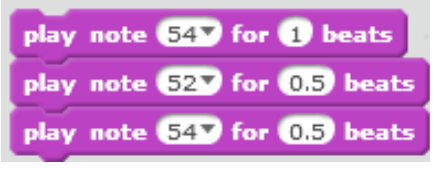
Đoạn nhạc này có 6 khuông nhịp.

Em hãy điền tiếp các lệnh Scratch cho các khuông nhịp 3, 4, 5.

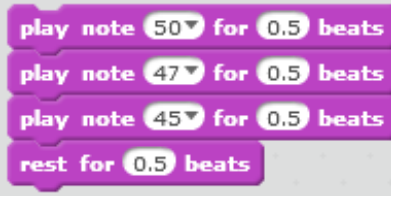
Nhịp 1



Nhịp 2

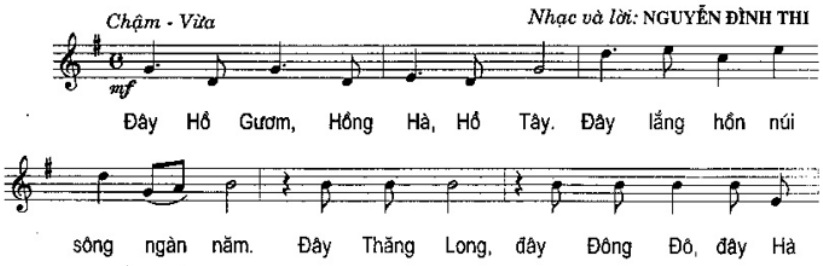


Nhịp 2



Nhịp 4 **Nhịp 5** **Nhịp 6**

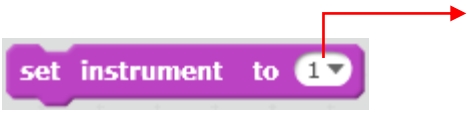
Ví dụ 2: một đoạn nhạc bài **Người Hà Nội** (nhạc: Nguyễn Đình Thi).



Đoạn đầu của bài hát được thiết lập như hình trên. Em hãy viết tiếp đoạn sau của bản nhạc này.

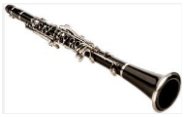
Lựa chọn công cụ chơi nhạc.

Lệnh thiết lập công cụ chơi nhạc trong Scratch là **set instrument to**.



Nhạc cụ lựa chọn tại đây.

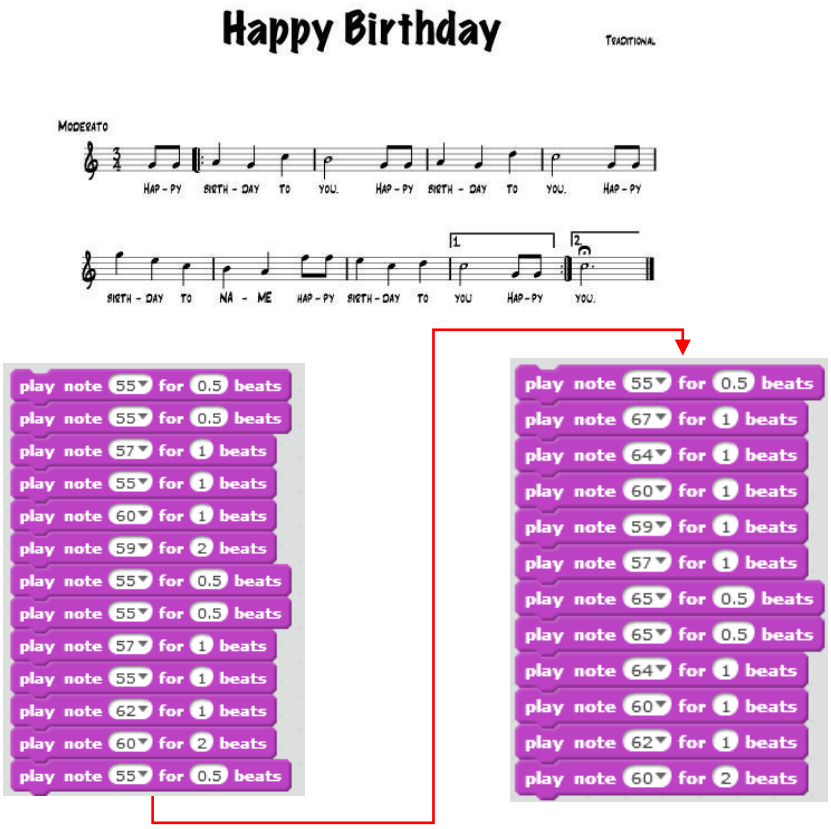
Bảng sau mô tả bằng hình các công cụ chơi nhạc hỗ trợ trong Scratch.

Piamo		Bass		Saxophone	
Electric piano		Pizzicato		Flute	
Organ		Cello		Wooden Flute	
Guitar		Trombone		Bassoon	
Electric guitar		Clarinet		Choir	

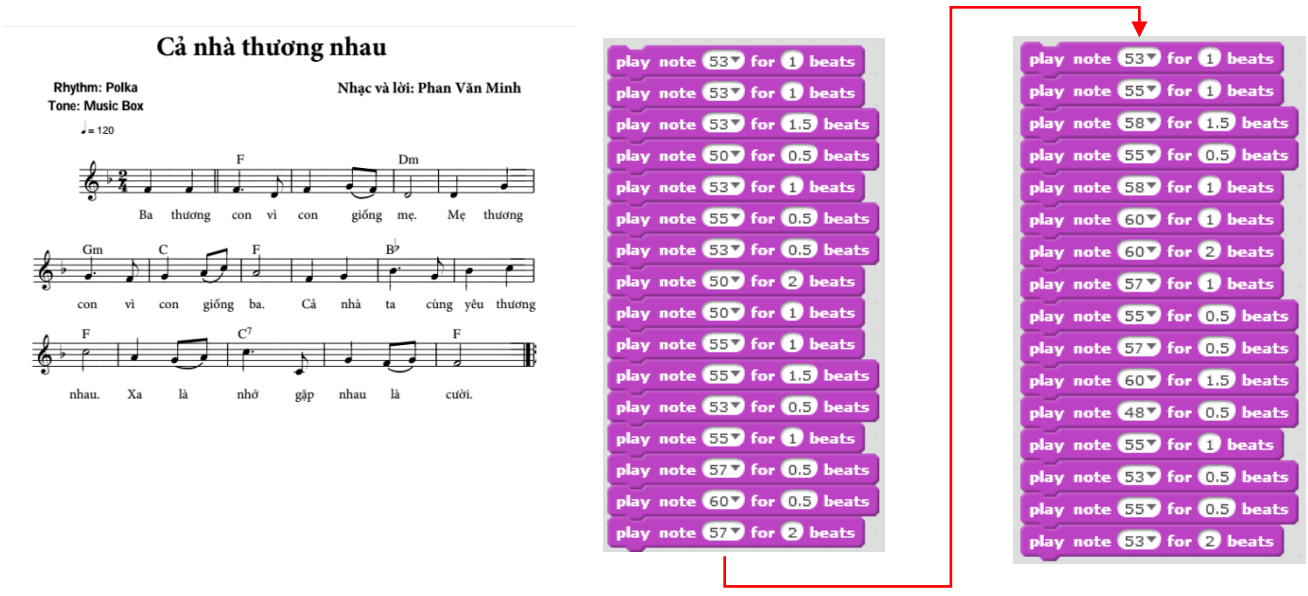
5. Một số bài hát, bản nhạc hoàn chỉnh

Em hãy cùng thực hiện tạo chương trình chơi các bản nhạc hoàn chỉnh sau.

Bài 1. Chúc sinh nhật. Happy birthday.

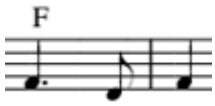
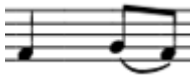
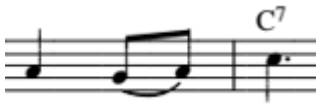


Bài 2. Cả nhà thương nhau. Nhạc và lời: Phan Văn Minh.



Câu hỏi và bài tập

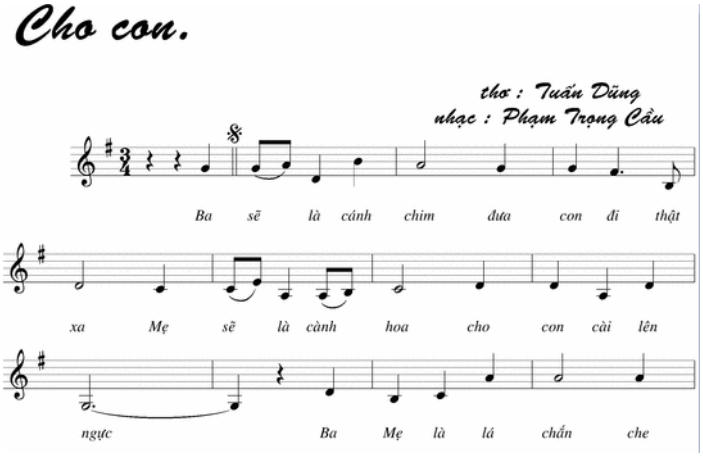
1. Điền vào cột 2 của bảng sau các lệnh Scratch tương ứng với các ký hiệu nốt nhạc tương ứng từ cột 1.

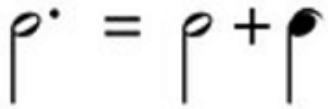
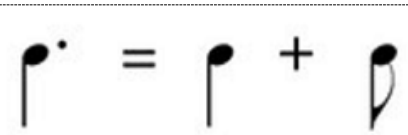
2. Sau khi viết 1 bản nhạc bằng dòng lệnh của Scratch, nếu muốn bản nhạc này chơi nhanh lên hay chậm đi thì dùng lệnh nào?

3. Viết 1 chương trình nhỏ điều khiển 2 hip-hop trong hình sau vừa đi, vừa nhảy trong tiếng trống trên sân khấu.

4. Em hãy viết đoạn chương trình thể hiện bài hát Cho con từ đầu cho đến "cho con cài trên ngực".



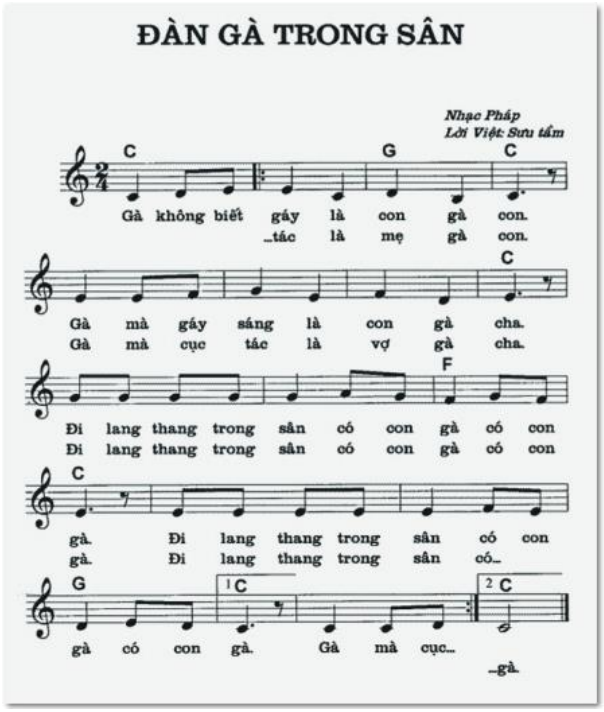
5. Em hãy viết đoạn chương trình Scratch tại cột bên trái để diễn tả ý nghĩa nốt nhạc trong cột trái.

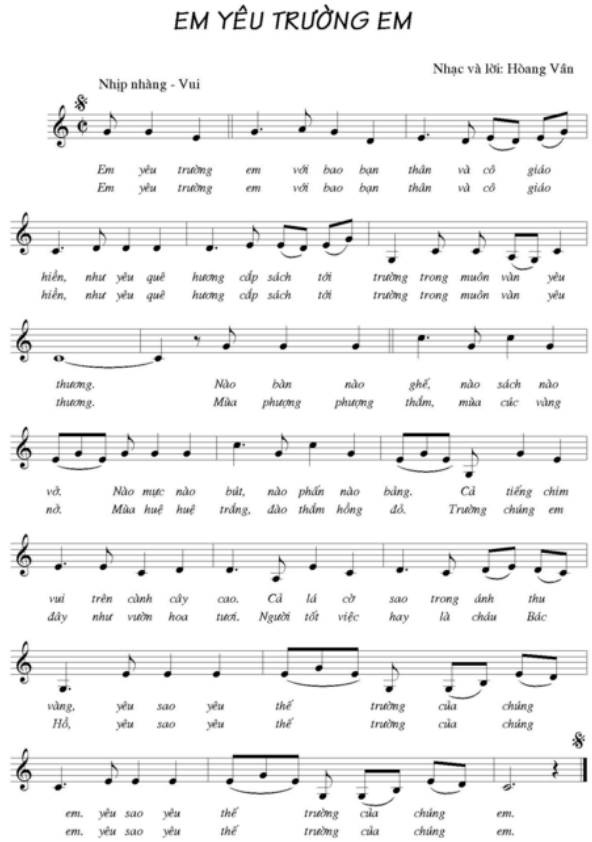
6. Em hãy viết đoạn chương trình Scratch tại cột bên trái để diễn tả ý nghĩa các nốt nhạc nghỉ trong cột trái.

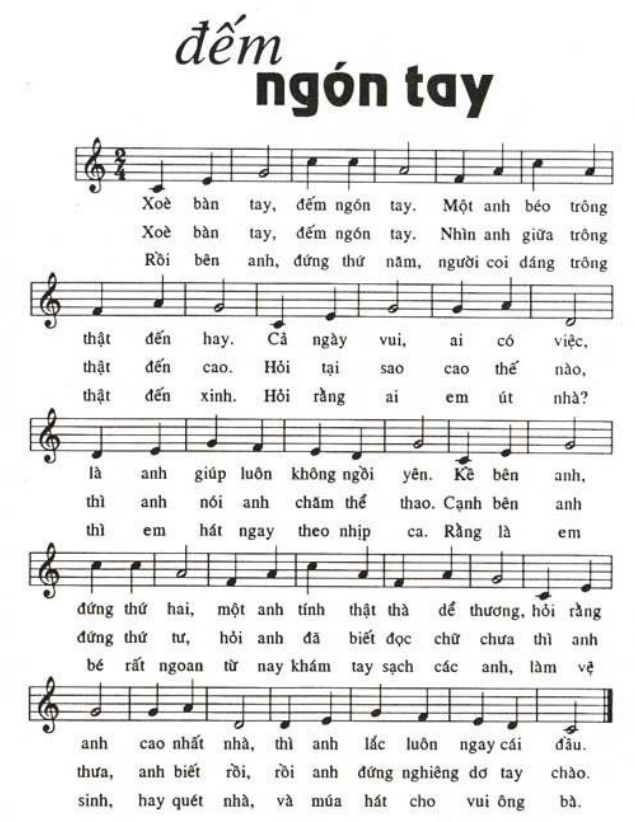
7. Em hãy tạo đoạn chương trình mô tả bài hát hoàn chỉnh, bài **Đàn gà trong sân**.



8. Em hãy tạo đoạn chương trình mô tả bài hát hoàn chỉnh, bài **Em yêu trường em**.

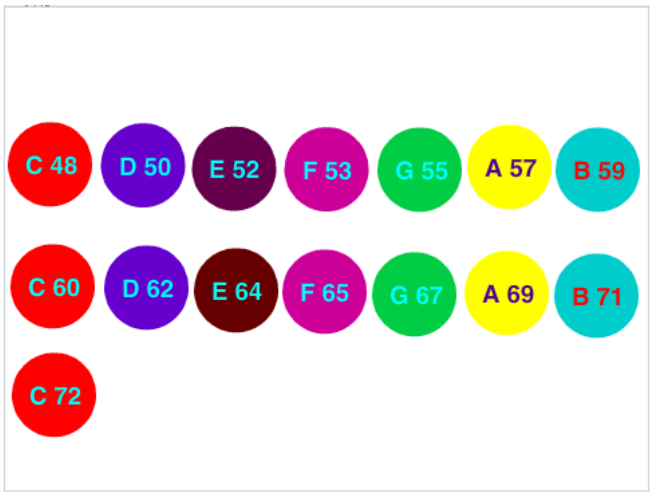


9. Em hãy tạo khuôn nhạc cho một bài hát hoàn chỉnh: **Đếm ngón tay**.



Mở rộng

Thiết lập chương trình đơn giản sau.

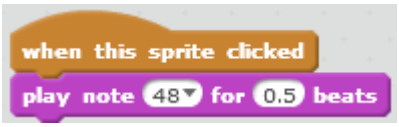


Trên màn hình là 15 nút tròn dùng để mô tả các nốt nhạc. Trên mỗi nút có chữ cái ký hiệu tên nốt và giá trị tương ứng của lệnh thể hiện nốt nhạc này.

Nháy chuột lên các nút tròn này sẽ phát âm thanh chính là nốt nhạc tương ứng.

Gợi ý: thiết lập 15 nhân vật có hình các nút tròn trên. Với mỗi nhân vật - nút tròn sử dụng lệnh bắt sự kiện nháy chuột sau.

Đoạn chương trình sau dành cho nút đỏ đầu tiên (C, 48).



CHƯƠNG 3: TÌM HIỂU SÂU HƠN SCRATCH

Chương 3 có chức năng đi sâu hơn vào những kỹ năng lập trình đặc trưng và chuyên biệt của Scratch.

- Học sinh sẽ bắt đầu làm quen với khái niệm biến nhớ, cách khởi tạo và vai trò của biến nhớ trong chương trình.

- 2 công nghệ cơ bản và sâu sắc nhất của Scratch được trình bày trong chương này là khái niệm thông điệp **truyền thông** và giới thiệu các lệnh **cảm biến**. Các kỹ năng này là cơ sở của việc tạo quan hệ giữa các nhân vật, giữa nhân vật và sân khấu, là các công cụ lõi để thiết kế các chương trình, hoạt cảnh animation, phần mềm hoàn chỉnh trên nền Scratch.

Như vậy chương 3 sẽ kết thúc phần học **Cơ bản** của Scratch và bắt đầu đi sâu vào phần **Nâng cao** và **Chuyên sâu** của môi trường lập trình này. Học hết chương này, học sinh sẽ có thể thiết lập các chương trình, phần mềm tương đối phức tạp, hoàn chỉnh có ý nghĩa thực tế.

Chương này sẽ bao gồm 3 bài học như sau:

Bài 9. Hội thoại

Bài 10. Hội thoại và truyền thông

Bài 11. Cảm biến

Bài 9. Hội thoại

Mục đích

Học xong bài này bạn có thể:

- Thực hiện giao tiếp hội thoại người - máy đơn giản.
- Hiểu được khái niệm biến nhớ, sử dụng biến nhớ trong cách giải quyết vấn đề.

Bắt đầu

1. Em hiểu thế nào là hội thoại người - máy, cho 1 vài ví dụ.

2. Em thực hiện đoạn chương trình trình diễn hội thoại đơn giản sau giữa 2 bạn Hoa và Lan.

```
Hoa: Chào Lan.
Lan: Chào bạn Hoa, lâu lắm mới gặp.
Hoa: Bạn bây giờ học trường nào?
Lan: Tôi học trường THCS Bế Văn Đàn. Bạn khỏe không?
Hoa: Cám ơn bạn, tôi vẫn khỏe.
```

Ví dụ có thể viết chương trình sau mô tả đoạn hội thoại trên.

Hoa

Lan

Theo em, chương trình trên có phải là hội thoại người - máy hay không? Vì sao?

3. Đoạn văn sau mô tả 1 hội thoại người - máy tính.

```
Máy tính: Bạn hãy nhập 1 số tự nhiên.
Người: thực hiện động tác nhập 1 số từ bàn phím, ví dụ
nhập số 12.
Máy tính: Bạn hãy nhập tiếp 1 số tự nhiên khác.
Người: thực hiện động tác nhập 1 số từ bàn phím, ví dụ
nhập số 18.
Máy tính (sau 1 giây suy nghĩ): Bạn đã nhập 2 số tự
nhiên 12, 18. Ước số chung lớn nhất 2 số này là 6.
```

Em có nhận xét gì về cuộc hội thoại trên.

Nội dung bài học

1. Bắt đầu 1 chương trình hội thoại đơn giản

Em hãy thiết lập chương trình để thực hiện hoạt động hội thoại sau.

Nhân vật chính là 1 thầy giáo, học sinh là người sử dụng máy tính.

```
Thầy giáo: Chào học sinh, em tên là gì?
Học sinh nhập từ bàn phím: Hòa Anh.
Thầy giáo: Chào Hòa Anh, rất vui được làm quen với em.
Sau 2 giây.
Thầy giáo: Bây giờ em hãy nhập từ bàn phím 1 số tự nhiên.
Học sinh nhập từ bàn phím: 15
Thầy giáo: Tốt lắm, em đã nhập số 15.
```

Trong ví dụ trên, chúng ta được làm quen với 1 dạng hội thoại thực sự giữa máy tính và người sử dụng. Máy tính đưa ra yêu cầu; học sinh trả lời thông qua bàn phím; máy tính nhận kết quả và đưa ra thông điệp tiếp theo tùy thuộc vào kết quả đã nhận được.

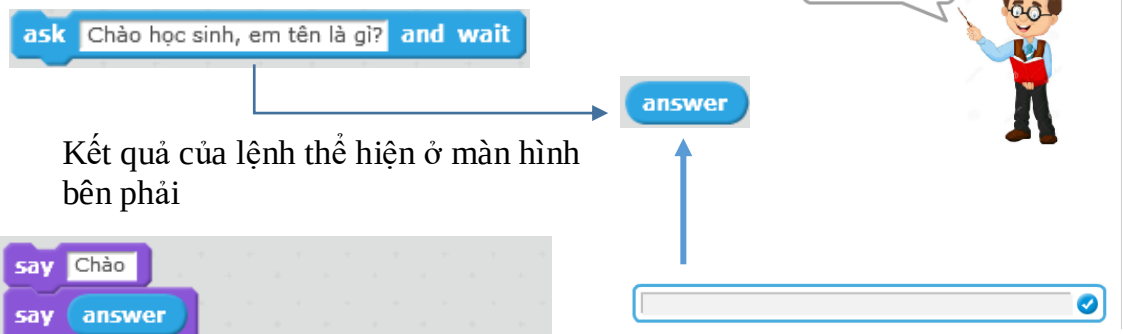
Để mô tả quá trình hội thoại trên chúng ta sử dụng lệnh sau từ nhóm lệnh Cảm biến (Sensing).

Lệnh 

(**ask and wait**) sẽ cho phép nhân vật hiện câu hỏi trên màn hình và chờ người dùng nhập câu trả lời từ bàn phím vào 1 dòng nhập liệu. Câu trả lời này sẽ được lưu trong "biểu thức" hay "lệnh" 

. Giá trị 

này sau đó có thể được sử dụng lại vào các công việc khác. Chúng ta cùng xét ví dụ cụ thể của lệnh này.



Kết quả của lệnh thể hiện ở màn hình bên phải

Vị trí nhập trả lời của lệnh từ bàn phím, kết quả lưu vào "biến nhớ" **answer**.

Giá trị **answer** được sử dụng để giáo viên tiếp tục hội thoại với học sinh dưới dạng đầy đủ như hình trên hoặc dạng rút gọn như hình dưới.

Trong lệnh trên chúng ta đã sử dụng lệnh 

trong nhóm lệnh **Tính toán** dùng để kết nối 2 giá trị hoặc 2 cụm từ.

Chương trình đầy đủ của hoạt động hội thoại này như sau:

2. Biến nhớ là gì?

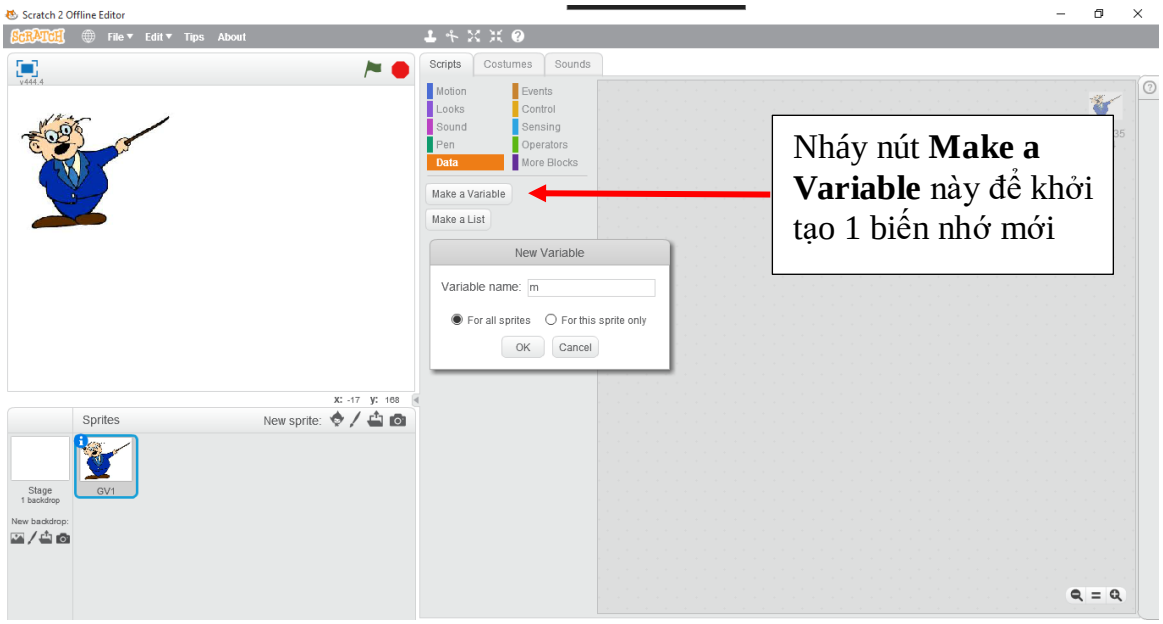
Trong hoạt động 1, chúng ta đã làm quen với lệnh hay "biến nhớ" **answer**, đây là 1 kỹ thuật được sử dụng thường xuyên trong Tin học, trong các môi trường lập trình. Cụm từ **answer** luôn được hiểu với ý nghĩa là 1 vùng trong bộ nhớ máy tính để lưu trữ giá trị mà người sử dụng nhập từ bàn phím thông qua lệnh **ask and wait**. Hiểu đơn giản biến nhớ được tạo ra để lưu trữ dữ liệu. Khi được tạo ra rồi thì chúng ta có thể thực hiện các phép tính với biến nhớ tương tự như với dữ liệu lưu trong biến nhớ này.

Em hãy thực hiện các chương trình đơn giản sau, qua đó hiểu được ý nghĩa của biến nhớ trong Scratch.

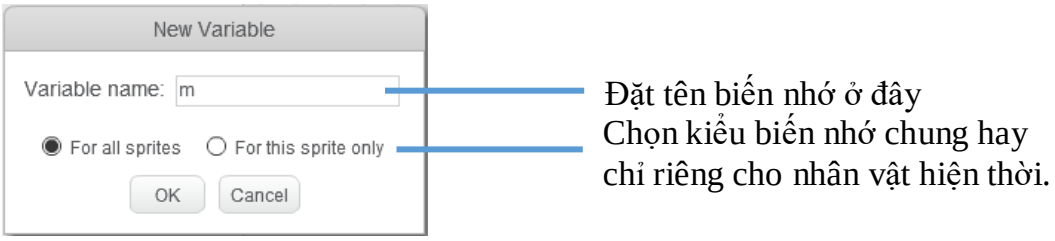
Chương trình 1.

Thầy giáo yêu cầu học sinh nhập từ bàn phím 2 số tự nhiên m, n, sau đó thông báo 2 số này và tổng của chúng.

Bước 1. Tạo 1 chương trình mới, bổ sung thêm nhân vật Thầy giáo, khởi tạo 2 biến nhớ mới đặt tên là m, n.



Để tạo 1 biến nhớ mới, em vào nhóm lệnh Dữ liệu (Data) và nháy nút Make a Variable. Cửa sổ tạo biến nhớ có dạng sau, em nhập m tại ô **Variable name**, chọn **For all sprites** và nháy **OK**.



Tương tự em tạo thêm biến nhớ có tên n. Hình ảnh sau trên khung mẫu lệnh của nhóm Dữ liệu (Data) sẽ xuất hiện các lệnh làm việc với biến nhớ.

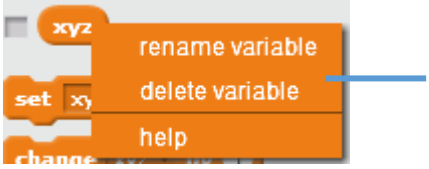
Có hai lệnh làm việc quan trọng với biến nhớ là:

2 lệnh chính thay đổi giá trị của biến nhớ.

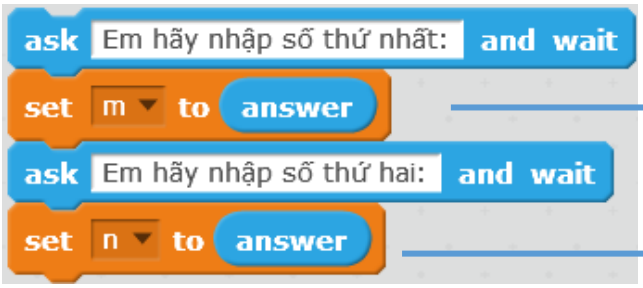
Lệnh thiết lập, gán 1 giá trị cụ thể cho biến nhớ.

Lệnh thay đổi giá trị của biến nhớ (tăng lên hoặc giảm đi) theo 1 giá trị cụ thể.

Chú ý: Nháy chuột phải lên 1 biến nhớ sẽ xuất hiện 1 bảng chọn nhỏ để thực hiện thêm các lệnh bổ sung cho biến nhớ này.



Bước 2. Thực hiện hội thoại giữa thầy giáo và học sinh (người dùng máy tính).



Gán giá trị của **answer** vào biến nhớ **m**.

Gán giá trị của **answer** vào biến nhớ **n**.

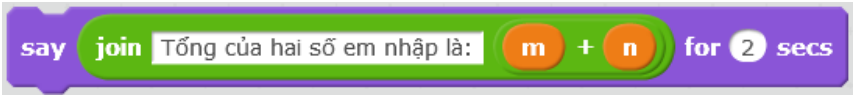
Bước 3. Thông báo tổng 2 số m, n.

Để thể hiện thông báo này chúng ta cần hiển thị 1 thông tin chữ và giá trị tổng m + n. Dùng lệnh / biểu thức 

(từ nhóm lệnh Tính toán) để thực hiện yêu cầu này. Lệnh hay toán tử 

dùng để tính giá trị m + n.

Kết quả của thông báo này như sau:



Chương trình 2.

Thay đổi chương trình trên, yêu cầu học sinh nhập 1 số tự nhiên với yêu cầu số này phải > 100. Giáo viên sẽ liên tục thông báo cho học sinh cần nhập đúng và yêu cầu nhập lại nếu học sinh nhập không đúng số theo yêu cầu.

Để thực hiện chương trình này chúng ta sẽ sử dụng một lệnh lặp mới, lệnh lặp có điều kiện repeat until để kiểm tra học sinh có nhập đúng yêu cầu hay không. Lệnh lặp này không chỉ ra trước số lần lặp, lệnh sẽ thực hiện việc lặp cho đến khi 1 điều kiện logic được thực hiện.

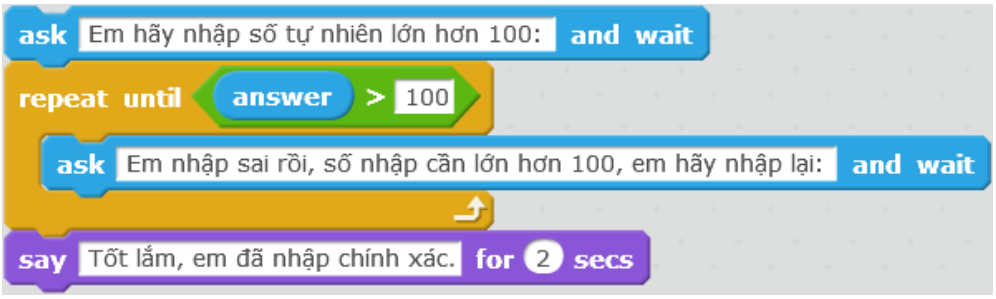
Biểu thức logic là điều kiện của lệnh lặp: nhóm lệnh này sẽ được lặp lại liên tục cho đến khi biểu thức này có giá trị đúng.

Với yêu cầu cụ thể của chương trình 2, điều kiện cần kiểm tra là giá trị số mà học sinh nhập cần > 100. Nếu điều kiện này đúng thì việc lặp sẽ dừng lại. Chúng ta sẽ sử dụng lệnh, biểu thức logic so sánh 

từ nhóm lệnh tính toán, cụ thể biểu thức cần đưa vào vị trí kiểm tra của lệnh lặp là 

.

Kết quả chương trình 2 được xây dựng như sau:



3. Lệnh, biểu thức, hàm có giá trị dùng như biến nhớ

Trong Scratch có rất nhiều lệnh có ý nghĩa của 1 biểu thứcm hàm số và được dùng tương tự như biến nhớ **answer** mà em đã biết.

Em cần chú ý đến hình dạng các lệnh của Scratch để phân biệt.

	Lệnh có 2 đầu thẳng góc	Đây là các lệnh bình thường, không có giá trị trả lại như 1 biểu thức.
	Lệnh có 2 đầu nhọn	Đây là các lệnh trả lại giá trị logic, tức là Đúng hoặc Sai (True or False).
	Lệnh có 2 đầu tròn	Đây là các lệnh trả lại giá trị số hoặc chữ.

Trong các loại lệnh trên, 2 loại sau có thể sử dụng như 1 biến nhớ, có thể đưa vào trong các điều kiện, biểu thức tính toán của các lệnh khác.

Các biểu thức, lệnh hay biến nhớ có giá trị này có thể chèn vào các vị trí trống trong các lệnh khác nhau của Scratch. Hình sau mô tả qui định của các giá trị có thể chèn.

Tại các vị trí **tròn** chỉ cho phép nhập hoặc chèn giá trị số, nguyên hoặc thập

Tại các vị trí **vuông** cho phép nhập hoặc chèn giá trị số, chữ hoặc logic.

Sau đây là 1 vài lệnh có ý nghĩa biểu thức và có thể sử dụng như các biến nhớ trong Scratch.

	Nhóm **Chuyển động (Motion)**	Trả lại giá trị là tọa độ X của nhân vật tại thời điểm hiện thời.
	Nhóm **Chuyển động (Motion)**	Trả lại giá trị là tọa độ Y của nhân vật tại thời điểm hiện thời.
	Nhóm **Cảm biến (Sensing)**	Trả lại trạng thái logic Đúng / Sai, đúng nếu nhân vật va chạm với 1 nhân vật khác hoặc cạnh sân khấu.
	Nhóm **Cảm biến (Sensing)**	Trả lại trạng thái logic Đúng / Sai, đúng nếu nhân vật tiếp xúc với màu sắc được chọn trong lệnh.
	Nhóm **Cảm biến (Sensing)**	Trả lại trạng thái logic Đúng / Sai, đúng nếu người dùng nhấn phím tương ứng.

	Nhóm **Tính toán (Operators)**	Trả lại giá trị số ngẫu nhiên nằm giữa 2 giá trị số được ghi trên thanh lệnh.
	Nhóm **Cảm biến (Sensing)**	Trả lại giá trị thời gian hiện thời (giây, phút, giờ, năm, ….). Các thông số có thể lựa chọn. - year: năm (2016) - month: tháng (1→12) - date: ngày của tháng (1→31) - day of week: ngày của tuần (1→7) - hour: giờ hiện thời (0→23) - minute: phút hiện thời (0→59) - second: giây hiện thời (0→59)
	Nhóm **Tính toán (Operators)**	Trả lại giá trị dãy ký tự là ghép nối của 2 biểu thức trong ô vuông của lệnh.

4. Một số ví dụ sử dụng biến nhớ và hàm số

4.1. Để điều khiển nhân vật chính xuất hiện ngẫu nhiên trên màn hình, em có thể sử dụng đoạn lệnh sau:

4.2. Chúng ta muốn thể hiện thời gian chạy online trên màn hình, ví dụ trong hình sau.

thì cần thực hiện lệnh sau:

4.3. Điều kiện kiểm tra xem nhân vật đã chạm đáy màn hình chưa.

4.4. Điều kiện kiểm tra xem nhân vật đã chạm đỉnh trên màn hình hay chưa.

5. Tạo nút cho phép nhập điều chỉnh biến nhớ trên màn hình

Hoạt động này cho phép thiết lập giao diện cho phép người sử dụng nhập hoặc điều chỉnh dữ liệu ngay trên màn hình trong quá trình chạy chương trình mà không cần nhập bằng lệnh **ask and wait**.

Chúng ta cùng thực hiện bài toán sau.

`Chương trình cho phép người dùng nhập điều chỉnh 2 giá trị số m, n ngay trên màn hình và sau khi người dùng nháy chuột lên nhân vật chính thì chương trình thông báo tổng của hai số này.`

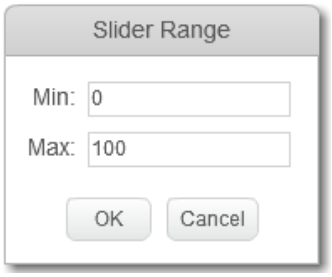
Để cho biến nhớ luôn hiển thị giá trị trên màn hình em cần nháy chọn vào ô hiển thị bên cạnh biểu tượng biến nhớ 

. Làm tương tự như vậy với biến nhớ n. Các biến nhớ hiện trên màn hình sẽ có dạng như sau:

Bây giờ chúng ta hãy thực hiện thao tác cho phép người dùng có thể thay đổi trực tiếp giá trị của các biến nhớ này trên màn hình. Với mỗi biến nhớ thực hiện 2 thao tác sau:

- Nháy chuột phải lên vị trí biến nhớ trên sân khấu và nháy chọn lệnh **slider**.

- Nháy chuột phải lên vị trí biến nhớ 1 lần nữa và chọn lệnh **set slider min and max** để thiết lập vùng dữ liệu muốn điều chỉnh trực tiếp giá trị của biến nhớ. Xuất hiện hộp hội thoại nhỏ sau, cần nhập 2 giá trị **Min** và **Max**.



Bây giờ em có thể thiết lập các lệnh để thực hiện theo yêu cầu của chương trình.

6. Hội thoại với sân khấu

Sân khấu không có lệnh **say**, nhưng vẫn có thể thực hiện được lệnh **ask and wait**.

Xét ví dụ sau:

Hình bên cho ta thấy cách sân khấu hội thoại với người sử dụng.

Trong ví dụ trên Tên của người dùng sau khi nhập vào biến **answer** sẽ được lưu trong biến nhớ **Name**.

Chú ý:

Sân khấu cũng có quyền tạo các biến nhớ tương tự như nhân vật. Điểm khác biệt là sân khấu chỉ được phép khởi tạo các biến nhớ dùng chung.

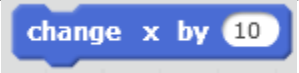
7. Điều khiển nhân vật chạy dọc màn hình theo 1 hướng

Bài toán **Mèo chạy**

Bài toán: mô tả nhân vật chạy ngang trên màn hình từ trái sang phải, sau khi khuất vào cánh gà sân khấu bên phải, nhân vật lại xuất hiện tại cánh gà sân khấu bên trái.

Để thực hiện yêu cầu này chúng ta không thể dùng lệnh 

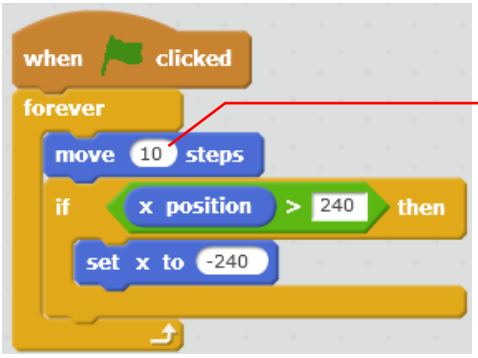
mà phải dùng cách khác. Một trong các cách có thể sử dụng là điều khiển dùng lệnh kiểm tra khi nào tọa độ X của nhân vật vượt ra ngoài sân khấu bên phải thì cần đặt lại giá trị tọa độ này của nhân vật. Chúng ta sẽ dùng 2 lệnh sau:

	Thiết lập lại giá trị tọa độ X của nhân vật.
	Thau đổi giá trị tọa độ X của nhân vật theo 1 số cho trước.
	Hàm trả lại tọa độ X chính xác của nhân vật tại thời điểm nhận giá trị của hàm.

Chương trình sẽ được thực hiện theo thuật toán đơn giản như sau:

```
Trong thời gian chạy
Kiểm tra nếu tọa độ X > 240 thì
    Đặt lại giá trị tọa độ X = -240
```

Chương trình trên Scratch như sau:



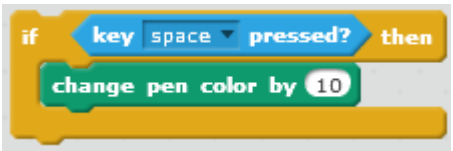
Chú ý quan trọng: Giá trị số bước chạy trong lệnh **move** có ý nghĩa mô phỏng tốc độ chạy của nhân vật. Giá trị này càng lớn, nhân vật chạy càng nhanh.

Tương tự có thể mô tả chương trình cho nhân vật chuyển động theo hướng thẳng đứng từ dưới lên hoặc ngược lại.

Câu hỏi và bài tập

1. Các lệnh sau đây thực hiện những công việc gì?

A.



B.

C.

2. Em hãy viết các lệnh hoặc đoạn chương trình mô tả các yêu cầu sau.

A. Mô tả thời gian hiện thời bằng tiếng Việt, ví dụ:

Bây giờ là 15 giờ 20 phút 14 giây.

B. Cho nhân vật chuyển động ngẫu nhiên trên sân khấu, nếu gặp cạnh sân khấu thì quay lại.

C. Đếm ngược thời gian: biến nhớ Time được gán ban đầu giá trị 10 và sẽ hiển thị đếm ngược trên màn hình về 0.

3. Viết chương trình với 2 nhân vật (giả sử là Hùng và Vân) nói chuyện với nhau như sau:

Hùng: Chào bạn Vân, bạn đi đâu đấy?

Vân: Chào Hùng, tôi đi chợ. Còn bạn, bạn đi đâu vậy?

Hùng: Tôi đi học nhóm. Chào bạn nhé.

Vân: Chào bạn, tạm biệt.

Chú ý: khi người này nói xong người kia mới nói tiếp.

4. Viết chương trình mô tả hoạt động của câu chuyện sau.

Nhân vật: Thầy giáo và học sinh tên Hùng. Thầy chưa biết tên học sinh.

Thầy giáo: Chào học sinh, em tên gì?

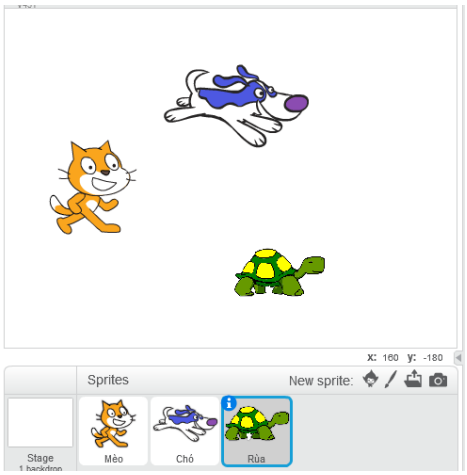
Học sinh: Dạ, em tên Hùng. (Trả lời qua lệnh ask)

Thầy giáo: Chào Hùng. Bây giờ thầy yêu cầu em làm bài toán sau nhé. Số 20 có bao nhiêu ước số nguyên tố?

Học sinh: 5

Thầy giáo: em làm sai rồi.

5. Viết chương trình mô tả 3 con vật: con Mèo, con Chó, con Rùa cùng chạy từ trái sang phải trên màn hình nhưng 3 con chạy với vận tốc khác nhau.



Gợi ý: xem lại ví dụ Mèo chạy và kiểm tra tham số nào xác định tốc độ chạy của nhân vật.

6. Mở rộng chương trình trên (bài 5) bằng cách bổ sung thêm nền sân khấu là một khu đất rộng để có hiệu ứng chạy thực sự của các con vật.

Chim bay 1. sb2

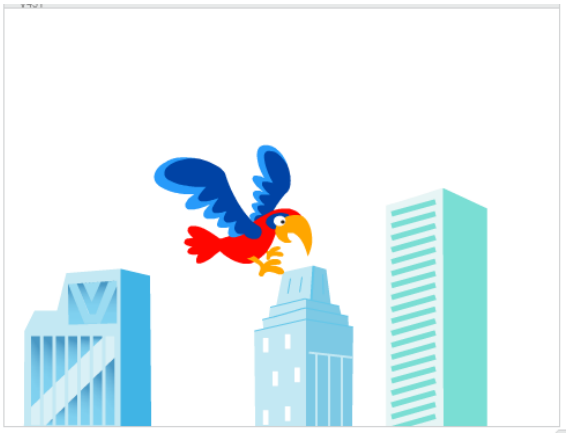
7. Bài toán mô phỏng **chim bay**.

Thiết kế chương trình với các yêu cầu, điều kiện sau:

- Nhân vật: chú vẹt (parrot) và các tòa nhà.

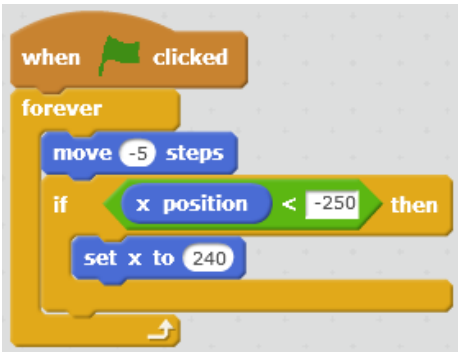
Chú ý: trong tài nguyên của Scratch có sẵn hình ảnh tòa nhà này. Muốn tạo thêm nhiều tòa nhà nữa sử dụng lệnh duplicate (nháy chuột phải lên nhân vật trong khu vực điều khiển, chọn duplicate), sau đó dùng các trang phục khác nhau của tòa nhà.

- Để điều khiển chim bay chúng ta sẽ điều khiển cho các tòa nhà chuyển động từ phải qua trái (với cùng 1 vận tốc).



Nhân vật chính của chương trình là con vẹt và các tòa nhà.

Ví dụ đoạn chương trình điều khiển 1 tòa nhà:



Em hãy hoàn thiện chương trình và ghi lại với tên **Chim bay 1.sb2**.

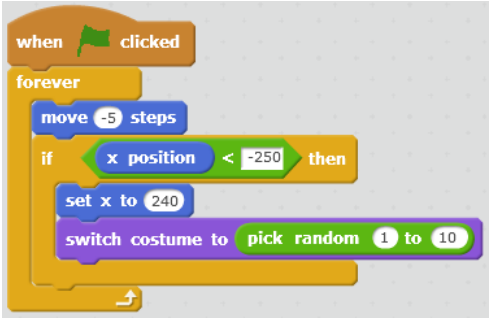
8. Mở rộng chương trình chim bay trên theo hướng sau:

a) Cho con vẹt thay đổi liên tục trang phục để tạo hình ảnh vỗ cánh bay.

b) Cho các tòa nhà thay đổi hình dạng mỗi khi ra khỏi sân khấu, để tạo hình ảnh đa dạng của thành phố.

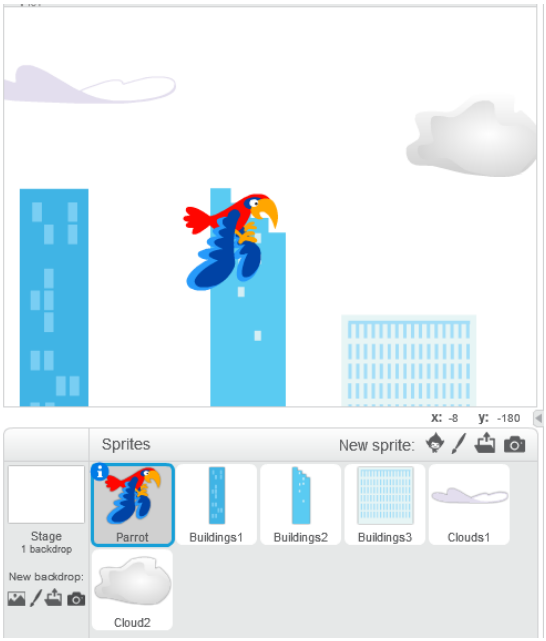
Với thay đổi a) chúng ta bổ sung đoạn chương trình sau cho chim vẹt:

Với thay đổi loại b), đoạn chương trình sẽ thay đổi như sau:



Em hãy thực hiện các thay đổi này, và ghi lại tên chương trình **Chim bay 2.sb2**.

9. Trong chương trình chim bay trên, bổ sung các đám mây, cho các nhân vật mây này chuyển động từ phải sang trái nhưng với vận tốc khác nhau để cảm nhận đươc chim thực sự bay trên bầu trời.

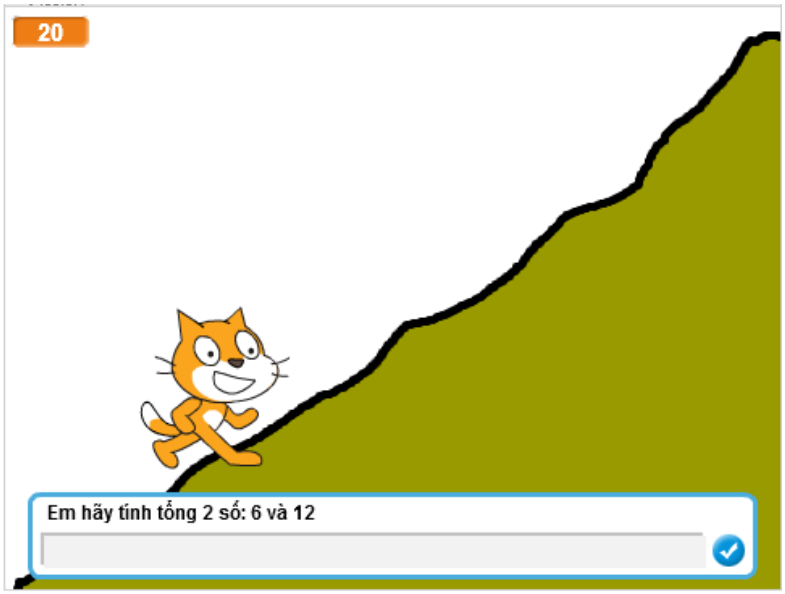


Em hãy thực hiện các thay đổi này, và ghi lại tên chương trình **Chim bay 3.sb2**.

10. Em hãy mở rộng chương trình chim bay hơn nữa, bổ sung thêm tiếng chim vẹt kêu, tiếng gió, mây để làm cho chương trình thêm sinh động, hấp dẫn.

11. Viết chương trình **Mèo leo núi** như sau:

Meo leo nui.sb2



- Bắt đầu chương trình Mèo ở chân núi.

- Chương trình sẽ liên tục đưa ra câu hỏi là một phép tính yêu cầu người chơi trả lời. Nếu trả lời đúng sẽ được thưởng 1 điểm, nếu sai bị trừ 1 điểm.

- Cứ mỗi lần người chơi đạt được các mốc điểm 5, 10, 15, … mèo sẽ đi được 30 bước lên phía đỉnh núi.

- Khi Mèo lên được đỉnh núi ở góc phải trên, Mèo sẽ kêu lên "Ura thắng rồi" và chương trình kết thúc.

Mở rộng

1. Hãy thiết kế 1 trò chơi đơn giản **Mèo và Bóng** sau:

Trên màn hình có con Mèo và quả Bóng.

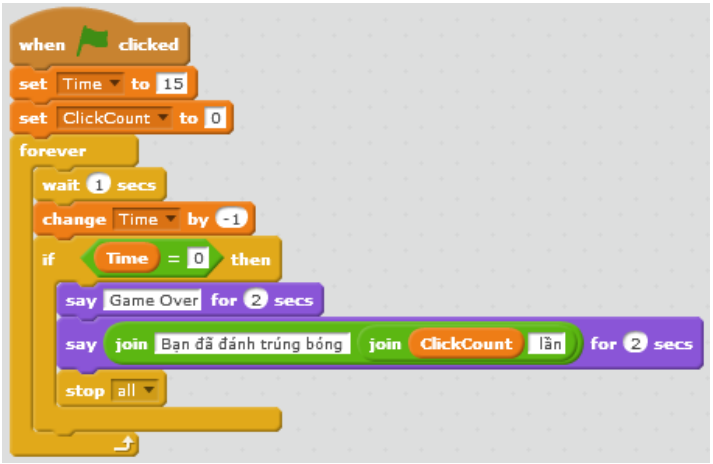
Thời gian cho mỗi cuộc chơi là 15 giây. Trên màn hình biến nhớ Time sẽ đếm ngược từ 15 về 0.

Khi bắt đầu chương trình, quả bóng sẽ dịch chuyển nhanh trên sân khấu một cách ngẫu nhiên. Nhiệm vụ của người chơi là nháy chuột nhanh lên quả bóng, càng nhiều lần càng tốt.

Khi thời gian lùi về 0, trò chơi dừng lại. Mèo con thông báo "Game Over" và sau đó thông báo số lần bạn nháy chuột chính xác lên quả bóng.

Thiết lập 2 nhân vật Mèo và Bóng. Tạo 2 biến nhớ la **Time** - thời gian của mỗi lần chơi và **ClickCount** dùng để đếm số lần người chơi nháy đúng lên quả bóng.

Chương trình cho Mèo.



Chương trình cho Bóng.

2. Bổ sung thêm âm thanh vào trò chơi trên cho chương trình thêm sinh động khi chơi.

Bài 10. Hội thoại và truyền thông

Mục đích

Sau khi học xong bài này, bạn sẽ làm được:

- Thực hiện bài toán hội thoại có điều khiển thông qua thông điệp.

- Giải quyết vấn đề thông qua thông điệp truyền thông giữa các nhân vật.

Bắt đầu

1. Em đã bao giờ tham dự 1 cuộc họp ở lớp mà các bạn đều đồng thanh nói, không ai chịu nghe ai cả chưa? Hãy kể 1 vài ví dụ thực tế như vậy.

2. Em hãy mô tả theo hiểu biết của em về hoạt động của mô hình thư điện tử (email) trên thực tế.

3. Trên lớp học, cô giáo ra 1 bài tập yêu cầu cả lớp làm. Sau 5 phút, cô nói: bạn nào xung phong lên bảng làm. Một số cánh tay giơ lên. Cô gọi 1 bạn lên bảng làm cho cả lớp xem. Em có nhận xét gì về mô hình hội thoại trong ví dụ trên.

Nội dung bài học

1. Bài toán hội thoại có điều khiển giữa 3 nhân vật

Chúng ta cùng nhau xây dựng chương trình với bài toán sau:

Có 3 bạn: Lan, Bình, Việt trên sân khấu,

Lan

Bình

Việt

Đầu tiên Lan chào: "Xin chào bạn Bình, bạn đi đâu đấy"?.

Bình nghe hiểu và trả lời: "Tôi đi chơi với mẹ".

Sau đó Lan nói: "Việt đấy à, chào bạn".

Việt nghe thấy và trả lời: "Chào Lan, bạn đi đâu đấy?

Qui trình thực hiện bài toán hội thoại - truyền thông này như sau.

- Lan thiết lập 2 thông điệp "Hỏi Bình" và "Chào Việt".
- Lan nói "Xin chào bạn Bình, bạn đi đâu đấy?" và gửi đi thông điệp "Hỏi Bình".
- Bình nhận được thông điệp "Hỏi Bình" thì thực hiện 3 việc sau:
    + Thiết lập thông điệp "Bình xong rồi".
    + Nói: "Tôi đi chơi với mẹ".
    + Gửi đi thông điệp "Bình xong rồi".
- Lan nhận được thông điệp "Bình xong rồi" thì thực hiện tiếp 2 việc sau:
    + Lan nói: "Việt đấy à, chào bạn".
    + Gửi đi thông điệp "Chào Việt".
- Việt khi nhận được thông điệp "Chào Việt" thì nói: "Chào Lan, bạn đi đâu đấy".

Chương trình được xây dựng và thiết kế như sau.

Chương trình của Lan.

Chương trình của Bình.

Chương trình của Việt.

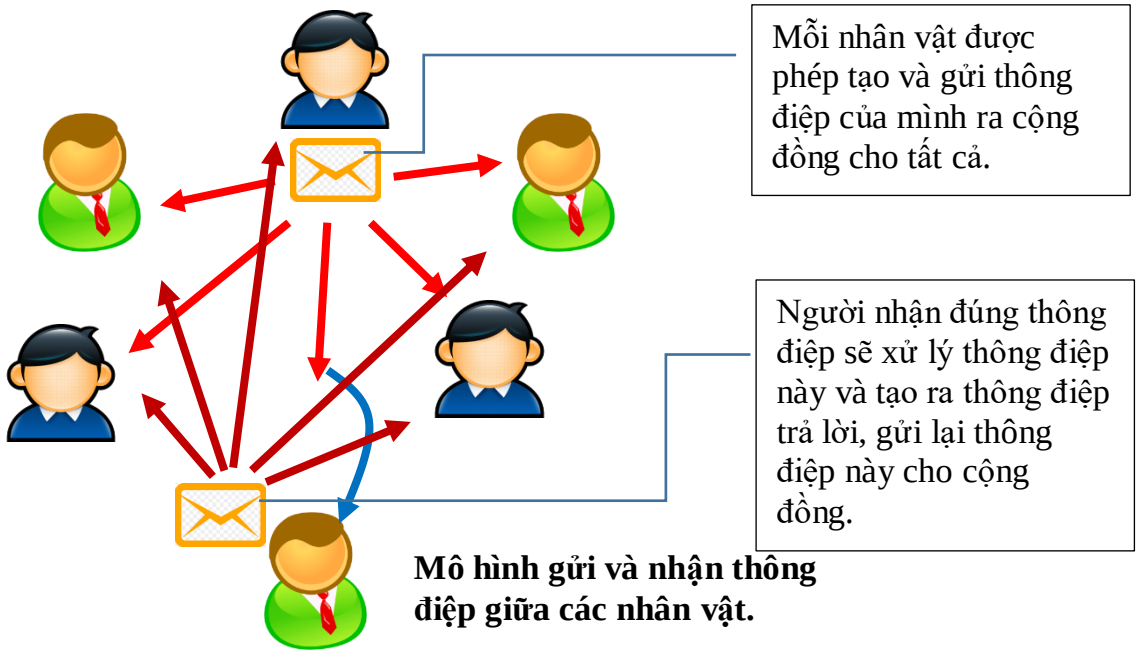
2. Gửi và nhận thông điệp

Đọc và ghi nhớ các lệnh truyền thông trong Scratch.

Để xử lý các bài toán hội thoại - truyền thông, trong Scratch có 2 lệnh quan trọng sau trong nhóm **Sự kiện**.

	Lệnh phát / truyền thông điệp, đồng thời có thể tạo ra 1 thông điệp mới. Mỗi nhân vật có thể tạo ra và phát không hạn chế các thông điệp. Mỗi thông điệp có 1 tên riêng và tồn tại trong suốt thời gian thực hiện chương trình.
	Sự kiện nhận thông điệp. Lệnh sự kiện này cho phép người nhận thông điệp xử lý chính xác
	Cửa sổ tạo 1 thông điệp mới. - Vào lệnh **broadcast**, chọn **New Message**. - Nhập tên của thông điệp tại vị trí **Message Name**. - Nháy **OK**.

Như vậy mô hình gửi và nhận thông điệp, xử lý khi nhận thông điệp là các công cụ chính xử lý bài toán truyền thông, quan hệ giữa các nhân vật trong mô hình Scratch.



Mô hình gửi và nhận thông điệp giữa các nhân vật.

Xét 1 vài ví dụ sau, em hãy viết suy nghĩ của mình cần đưa ra quan hệ, thông điệp gì giữa các nhân vật để giải quyết các bài toán sau.

Bài toán, vấn đề	**Quan hệ truyền thông giữa nhân vật**
Có 1 thầy giáo và nhiều học sinh trên sân khấu. Tại 1 thời điểm thầy chỉ nói được với 1 học sinh. Làm thế nào để học sinh nào biết được thầy đang nói và đang hỏi ai?	

Bài toán, vấn đề	Quan hệ truyền thông giữa nhân vật
Trên sân khấu có 3 nhân vật: 1 bút chì, 2 nút có màu xanh và đỏ. Khi người dùng nháy lên nút đỏ thì bút chì vẽ màu đỏ, khi người dùng nháy lên nút xanh, bút sẽ vẽ màu xanh. Mô tả mô hình truyền thông của sân khấu này.	
Trên sân khấu có 2 vận động viên, 1 quả bóng và 1 trọng tài. Khi trọng tài thổi còi, 2 vận động viên bắt đầu đá quả bóng sang nhau. Hãy mô tả mô hình truyền thông của sân khấu này.	

3. Sử dụng "nút lệnh" trong các bài toán giao tiếp

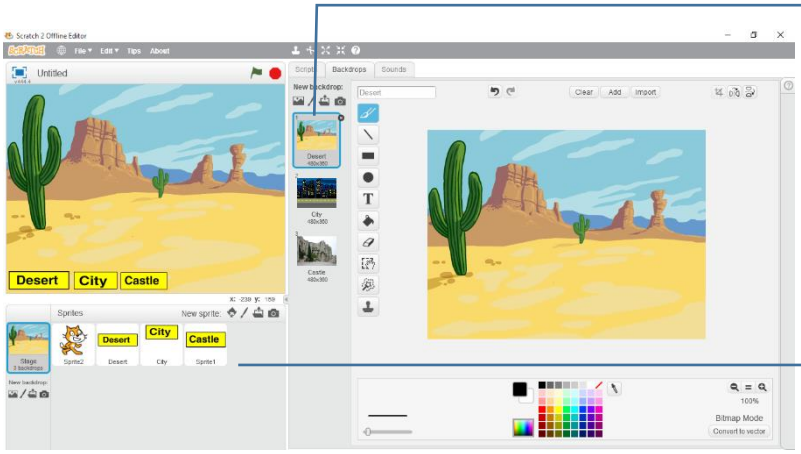
Chúng ta cùng nhau thực hiện một chương trình đơn giản có sử dụng các "nút lệnh" để hiểu thêm vai trò của thông điệp trong các bài toán có giao tiếp.

Thiết lập 1 chương trình bao gồm có 3 nút lệnh: Desert, City và Castle. Khi nháy lên các nút này thì nền sân khấu sẽ thay đổi hình ảnh tương ứng.

Để xây dựng chương trình này, em cần thực hiện các việc sau:

- Thiết lập 1 dự án Scratch mới, bổ sung 3 nền sân khấu và đặt tên Desert, City và Castle.

- Tạo mới 3 nhân vật có hình nút lệnh có tên Desert, City và Castle.



Thiết lập 3 hình nền sân khấu với các tên gọi desert (sa mạc), city (thành phố), castle (lâu đài).

Thiết lập 3 nhân vật dạng nút lệnh với tên gọi desert (sa mạc), city (thành phố), castle (lâu đài).

- Làm ẩn nhân vật chính (Mèo con) trên sân khấu.

- Với 3 nhân vật (nút lệnh) vừa khởi tạo, mỗi nhân vật thiết lập 1 thông điệp có tên trùng với tên của nhân vật.

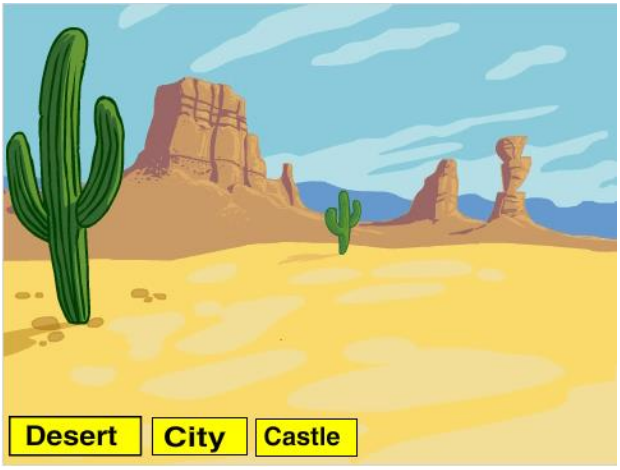
- Xây dựng 3 đoạn chương trình tương ứng với 3 nút lệnh này như sau:

Desert

City

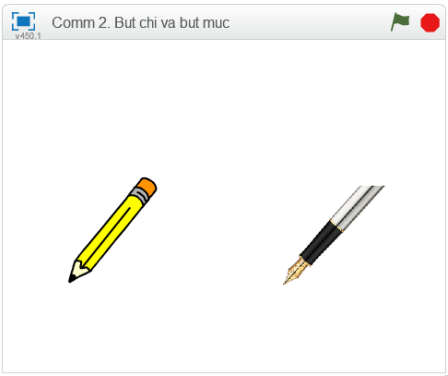
Castle

Bây giờ có thể thiết lập đoạn chương trình của nhân vật chính như sau:



4. Thông điệp của nền sân khấu

Sân khấu cũng có thể truyền thông điệp như mọi nhân vật trong môi trường Scratch. Em hãy thiết lập các chương trình theo thiết kế sau và phân tích ý nghĩa của việc truyền thông điệp từ sân khấu. Xét ví dụ sau:



Giả sử có 2 nhân vật là Bút chì và Bút mực.

Yêu cầu bút chì vẽ 1 hình vuông màu đỏ, bút mực vẽ 1 hình vuông màu xanh.

Trước khi vẽ cần làm sạch màn hình.

Chúng ta cùng phân tích, thực hiện bài toán này theo 2 cách.

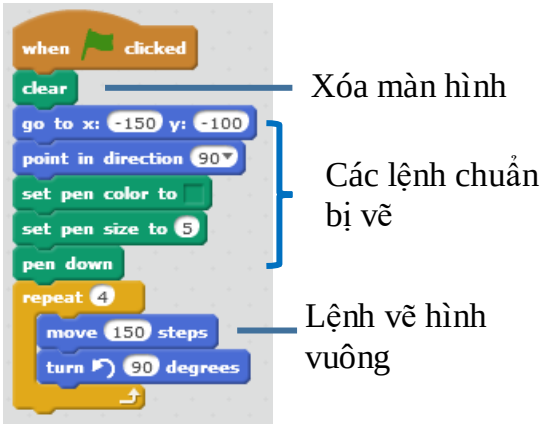
Chương trình 1: Không dùng thông điệp.

Nhân vật 1: Bút chì	**Nhân vật 2: Bút mực**
Xóa các hình đã có trên màn hình. Bắt đồng công việc vẽ của mình.	Xóa các hình đã có trên màn hình. Bắt đầu công việc vẽ của mình.

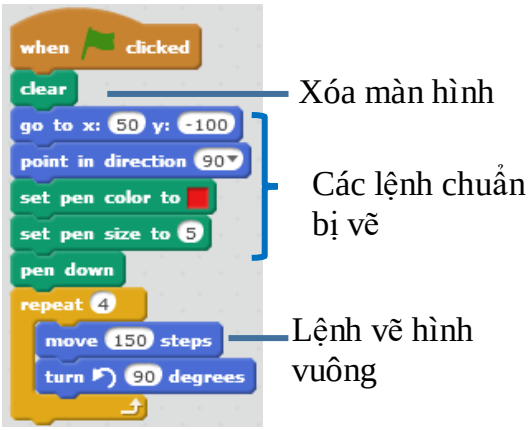
Đây là cách thực hiện chương trình theo tư duy dễ hiểu nhất. Mỗi nhân vật có 1 chương trình riêng, sau khi xóa màn hình, thực hiện các bước chuẩn bị công cụ vẽ, cuối cùng là thực hiện lệnh vẽ hình vuông.

Chương trình của 2 nhân vật Bút chì, Bút mực có thể như sau:

Bút chì



Bút mực



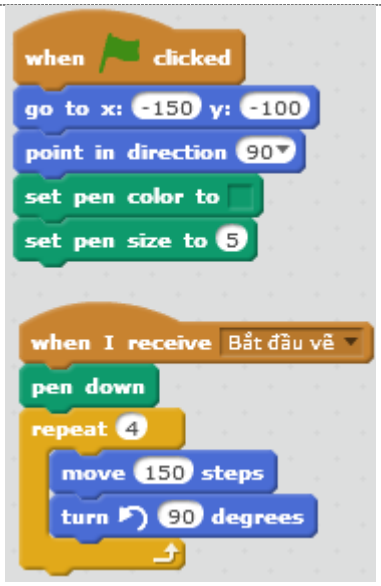
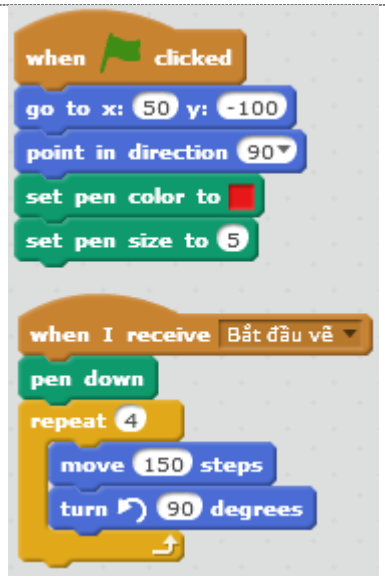
Em hãy thực hiện chương trình trên và đưa ra nhận xét của mình.

Câu hỏi: vì sau một trong 2 hình vuông tạo bởi 2 nhân vật trên lại không hoàn thiện? Có 1 hình vuông bị khuyết?

Lý do rất đơn giản: lệnh **clear** của 2 nhân vật không được thực hiện đồng thời vì chúng độc lập, không có quan hệ với nhau.

Bây giờ chúng ta cùng thực hiện phương án 2 của chương trình. Phương án này sẽ dùng kỹ thuật truyền thông điệp, nhưng thông điệp được gửi từ nền sân khấu đồng thời cho 2 nhân vật.

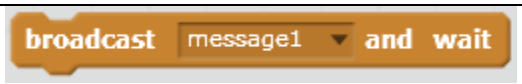
Chương trình 2: dùng thông điệp từ sân khấu.

Sân khấu	**Nhân vật 1: Bút chì**	**Nhân vật 2: Bút mực**
Xóa các hình đã có trên màn hình. Ra thông điệp: **bắt đầu vẽ**	Nếu nhận được thông điệp **bắt đầu vẽ** thì tiến hành công việc của mình.	Nếu nhận được thông điệp **bắt đầu vẽ** thì tiến hành công việc của mình.
		

Nhận xét: Lần này chương trình chạy hoàn hảo, chính xác.

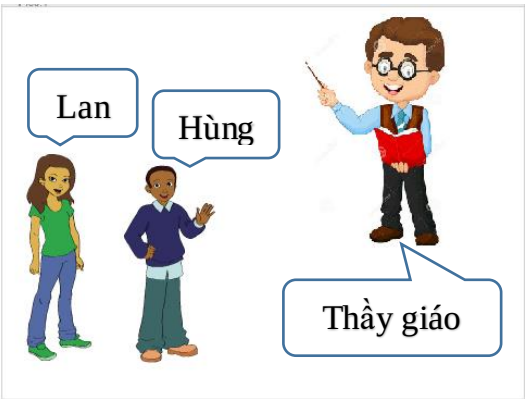
5. Phân biệt 2 lệnh truyền thông điệp

Trong Scratch có 2 lệnh **broadcast** (truyền thông điệp). Ý nghĩa của chúng được ghi trong bảng sau.

 Gửi thông điệp xong thì tiếp tục thực hiện các lệnh sau lệnh này (không dừng lại).	 Gửi thông điệp xong thì dừng lại, nếu có ít nhất 1 nhân vật khác nhận thông điệp và thực hiện xong nhóm lệnh ***when I receive <message>*** thì mới thực hiện lệnh tiếp theo.

Xét ví dụ để hiểu sự khác nhau của 2 lệnh trên.

Giả sử có 3 người Lan, Hùng và Thầy giáo như hình sau.



Lan và Hùng đang đi trên đường thì gặp thầy giáo, cả 2 cùng chào thầy.

Lan chào xong thì tiếp tục đi.

Hùng chào xong đứng lại đợi thầy chào lại thì mới đi tiếp.

Chúng ta hãy lập trình cho 3 nhân vật theo yêu cầu trên.

Lan

Hùng

Thầy giáo

Em sẽ thấy gì?

Lan: sau khi gửi thông điệp vẫn đi tiếp mặc dù Hùng đã nhận thông điệp này và đã trả lời.

Hùng: sau khi gửi thông điệp thì dừng lại, thầy giáo nhận thông điệp, trả lời xong sau 3 giây thì Hùng đi tiếp.

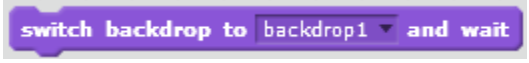
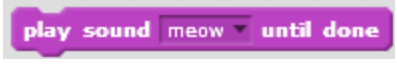
Thầy giáo: chỉ nhận thông điệp của Hùng và trả lời trong 3 giây.

6. Lập trình theo sự kiện hay theo thông điệp truyền thông

Bên cạnh kỹ thuật thông điệp, chúng ta cũng có thể thiết kế quan hệ giữa các nhân vật thông qua sự kiện. Ví dụ các sự kiện sau hỗ trợ trong Scratch.

Sự kiện	Lệnh hỗ trợ
Gõ một phím	
Nháy chuột lên một nhân vật	
Nháy chuột lên nền sân khấu	
Thay đổi nền sân khấu	
Thay đổi giá trị của một biến nhớ	
Khởi tạo 1 nhân thân (clone) của nhân vật. Khái niệm Clone sẽ được học kỹ trong các phần sau của chương trình.	

Một số lệnh khác trong Scratch có liên quan đến các sự kiện thời gian.

Lệnh	Ý nghĩa
 (lệnh dành cho nền sân khấu)	Lệnh chuyển nền sân khấu và đợi cho đến khi 1 nhân vật khác thực hiện xong nhóm lệnh bên dưới lệnh  .
	Nhân vật / sân khấu chơi bản âm thanh cho đến khi kết thúc bản nhạc.
	Nhân vật / sân khấu yêu cầu nhập dữ liệu và chờ cho đến khi nhập xong thì thực hiện lệnh tiếp theo. Dữ liệu người dùng nhập được lưu trong biến nhớ **answer**.

Chú ý: tất cả các cách trên đều có thể dùng như một thủ thuật lập trình khi cần kết nối các sự kiện và nhân vật. Tuy nhiên cách hiệu quả nhất vẫn là sử dụng thông điệp truyền thông qua lệnh **broadcast**.

Trong 1 số trường hợp sử dụng các lệnh có liên quan đến sự kiện cũng rất hợp lý, chính xác mà không cần dùng thông điệp.

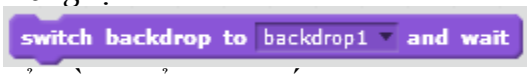
Một số ví dụ sử dụng sự kiện thay thế cho thông điệp

1. Giả sử 1 chương trình có 2 nền sân khấu và 1 nhân vật làm ca sĩ. Ca sĩ nãy cần hát 2 bài hát. Ca sĩ này có 2 trang phục. Yêu cầu của chương trình như sau:

Khi bắt đầu chương trình Ca sĩ hát bài đầu tiên. Khi hát xong thì tự động chuyển nền sân khấu, ca sĩ chuyển đổi trang phục và tự đông hát bài thứ 2 (yêu cầu không dùng lệnh gửi thông điệp).

Ví dụ có thể lập trình điều khiển nền sân khấu và nhân vật như sau:

San khau va ca si -1.sb2

Sân khấu	**Nhân vật Ca sĩ.**
	
Dùng lệnh  để điều khiển sân khấu tự động chuyển nền khi nhân vật thực hiện xong 1 nhóm lệnh mà không cần dùng thông điệp.	Đây là các chương trình điều khiển nhân vật theo các sự kiện khi sân khấu thay đổi nền. Sử dụng lệnh **play sound <…> until done** để kéo dài thời gian thực hiện lệnh theo âm thanh đang mở.

2. Chương trình có 2 nền sân khấu, 2 nhân vật. Khi bắt đầu chương trình nhân vật số 1 sẽ hát 1 bài, khi hát xong thì tự động chuyển nền sân khấu và nhân vật thứ 2 xuất hiện và hát bài hát của mình.

Em hãy thực hiện chương trình trên như một bài tập.

7. Vai trò truyền thông của thông điệp

Trong hoạt động trên em đã thấy rõ vai trò của thông điệp quan trọng như thế nào trong việc điều khiển kết nối giữa các nhân vật trên sân khấu và giải quyết các bài toán nói chung trong môi trường Scratch.

Thông điệp có tính năng truyền thông tin tuyệt đối chính xác.

Thông điệp được truyền qua các Message có tên chính xác, khi nhân vật nhận thông điệp này sẽ xử lý thông qua lệnh **when I receive <message>**, do vậy sẽ tuyệt đối chính xác.

Truyền thông điệp rất nhanh chóng, gần như tức thời.

Thông điệp được gửi bằng lệnh **broadcast** gần như tức thời, không có hiệu ứng chậm trễ về thời gian.

Việc lập trình sử dụng thông điệp truyền thông rất sáng sủa, rõ ràng và đơn giản.

Em hãy thử kiểm tra bằng cách thực hiện chương trình trong mục 3 bằng 2 cách. Cách 1 sử dụng thông điệp, cách 2 sử dụng biến nhớ để điều khiển.

Câu hỏi và bài tập

1. Một nhân vật có thể tạo đồng thời nhiều nhiều thông điệp và gửi đồng thời tất cả các thông điệp đó được hay không?

2. Một nhân vật có thể nhận và xử lý đồng thời 2 thông điệp từ 2 địa chỉ khác nhau được hay không? Cho ví dụ.

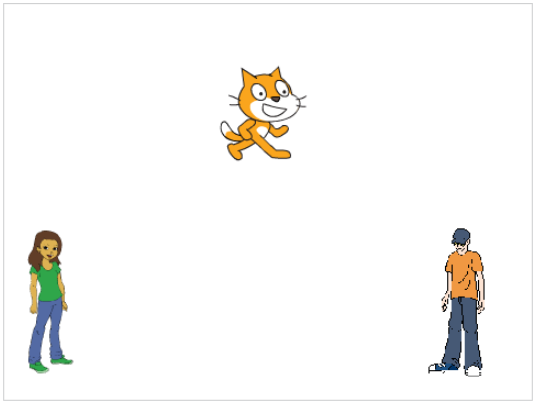
3. Hai hoặc nhiều nhân vật có thể gửi cùng 1 thông điệp được hay không? Cho ví dụ.

4. Viết chương trình cho bài yêu cầu sau.

Trên màn hình có 3 nhân vật: Mèo con, Lan và Hùng đang đứng ở vị trí như trong hình. Cả Lan và Hùng đều muốn Mèo chạy về phía mình.

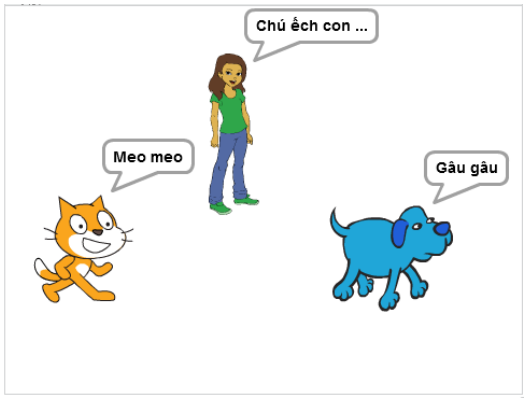
Khi nháy chuột lên Lan, Mèo sẽ quay về hướng Lan và chạy về phía Lan.

Khi nháy chuột lên Hùng, Mèo sẽ quay về hướng Hùng và chạy về phía Hùng.



Vị trí ban đầu của Mèo con, Lan và Hùng.

5. Viết chương trình thực hiện yêu cầu sau:



Có 3 nhân vật: Lan; Mèo con và Chó Cún.

- Nếu nháy chuột lên Lan thì Lan sẽ hát bài **Chú ếch con**.

- Nếu nháy chuột lên Mèo thì Mèo sẽ kêu **meo meo**.

- Nếu nháy chuột lên Chó thì Chó sẽ kêu **gâu gâu**.

6. Sửa đổi yêu cầu bài tập trên như sau:

- Khi nháy lên Mèo, nếu lúc đó Lan vẫn đang hát thì Lan sẽ dừng hát và Mèo sẽ chỉ kêu meo meo sau khi Lan đã dừng.

- Khi nháy lên Chó, nếu lúc đó Lan vẫn đang hát thì Lan sẽ dừng hát và Chó sẽ chỉ kêu gâu gâu sau khi Lan đã dừng.

7. Một chương trình kiểu "trình diễn" (tương tự PowerPoint presentation) có thể được thiết kế như sau bằng chương trình Scratch.

- Xây dựng và thiết kế nội dung của bài biểu diễn bằng 1 dãy các nền sân khấu, đánh số từ 2 đến n. Nền sân khấu gốc đánh số 1.

- Trên trang chính (slide 1) có 1 nút có tên **Bắt đầu**.

- Chương trình chạy như sau: Khi người dùng nháy lên nút **Bắt đầu**, nút này sẽ ẩn đi, chương trình sẽ trình diễn bằng cách thay đổi lần lượt nền sân khấu trên màn hình, mỗi nền hiện 5 giây. Khi trình diễn xong nút **Bắt đầu** xuất hiện trở lại.

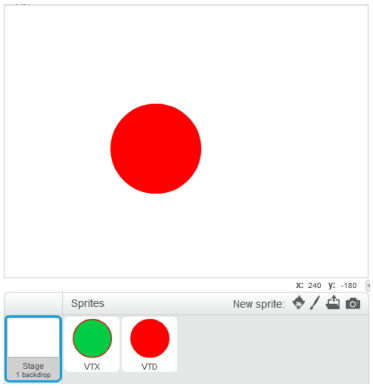
Em hãy thiết kế 1 chương trình dạng trình diễn như trên.

8. Có thể viết 1 chương trình như sau được hay không?

Trên màn hình ban đầu xuất hiện một vòng tròn màu đỏ, khi nháy chuột lên vòng tròn này thì nó biến thành màu xanh, nháy lần nữa lại biến thành màu đỏ.

Gợi ý:

Tạo ra 2 nhân vật là hình tròn bằng nhau, 1 hình màu đỏ, 1 hình màu xanh. Sử dụng thông điệp trao đổi để điều khiển sự xuất hiện của hai nhân vật này.



Ví dụ có thể điều khiển 2 nhân vật này như sau:

Vòng tròn đỏ	Vòng tròn xanh
	

Hãy thiết lập chương trình theo gợi ý trên và giải thích hoạt động của chương trình.

9. Viết chương trình thực hiện yêu cầu của hoạt động 6 của bài học:

Chương trình có 2 nền sân khấu, 2 nhân vật. Khi bắt đầu chương trình nhân vật số 1 sẽ hát 1 bài, khi hát xong thì tự động chuyển nền sân khấu và nhân vật thứ 2 xuất hiện và hát bài hát của mình. Yêu cầu không sử dụng lệnh **broadcast** trong chương trình.

10. Viết 1 chương trình hoàn chỉnh mô phỏng 1 buổi ca nhạc theo các yêu cầu sau:

- Mở màn chương trình sẽ xuất hiện MC trên nền sân khấu còn đóng. MC sẽ giới thiệu bài hát đầu tiên.

- Sau đó sân khấu mở ra cảnh 1, ca sĩ số 1 xuất hiện và hát 1 bài hát.

- Khi hát xong bài đầu tiên, MC xuất hiện trở lại, sân khấu đóng lại. MC giới thiệu tiết mục thứ 2.

- Sau đó sân khấu lại mở ra cảnh 2, ca sĩ số 2 xuất hiện và hát bài hát số 2.

- Khi hát xong, sân khấu đóng lại, MC xuất hiện thông báo kết thúc buổi ca nhạc và cám ơn khán giả.

Mở rộng

1. Mở rộng bài tập lớn của bài Vẽ hình 2, em hãy thiết lập chương trình Vẽ tự do sau.

Yêu cầu:

- Giao diện bao gồm 1 cây bút chì và 2 nút Xanh, Đỏ góc dưới màn hình.

- Khi di chuyển chuột trên màn hình, bút vẽ luôn đi theo vị trí con trỏ.

- Nhấn phím Space sẽ bật / tắt chế độ vẽ.

- Nháy chuột lên nút màu xanh hoặc đỏ sẽ có tác dụng đổi màu vẽ thành xanh hoặc đỏ.

2. Thiết lập chương trình đơn giản sau.

Yêu cầu:

- Giao diện phần mềm bao gồm: 1 Thầy giáo, 2 nút lệnh "Tính tổng" và "Tính hiệu". 2 giá trị số m, n có thể nhập, điều chính ngay trên màn hình.

- Khi người dùng nháy lên các nút lệnh "Tính tổng" và "Tính hiệu", thầy giáo sẽ thông báo kết quả của phép tính tương ứng ngay trên màn hình.

Ví dụ kết quả của phép **Tính hiệu** sẽ như sau:

Bài 11. Cảm biến

Mục đích

Học xong bài này, bạn sẽ hiểu và thực hiện được:

- Khái niệm và mô hình cảm biến trong Scratch.

- Hiểu được các lệnh cảm biến đơn giản: tọa độ, khoảng cách, màu sắc.

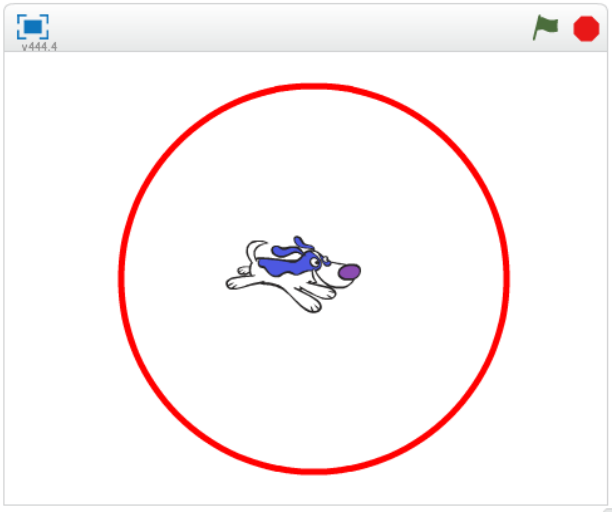
- Cảm biến va chạm giữa các nhân vật.

- Mô hình biến nhớ chung và riêng.

Bắt đầu

1. Em có nghe, đọc sách báo nói về 6 giác quan, em hãy nhớ lại đó là các giác quan nào?

2. Giả sử em muốn viết 1 chương trình cho 1 con vật (nhân vật) của em chạy tự do trong 1 vòng tròn màu đỏ, không cho nhân vật chạy ra ngoài. Em sẽ phải làm gì?



3. Trong các câu sau, câu nào có liên quan đến sự kiện, câu nào liên quan đến cảm biến?

- Sáng chủ nhật em đến trường tập hát.

- Đi trên đường em bị lóa mắt vì mặt trời chiếu thẳng vào mắt.

- Bạn An rất thích màu đỏ.

- Sáng em thức giấc đúng 6h khi nghe chuông báo thức.

- Sáng nào bố mẹ em cũng dậy sớm vào lúc 6h để tập thể dục.

- Khi nhận được thư mẹ, bố em lập tức viết thư trả lời.

- Khi nhấn phím Enter, dữ liệu được nhập sẽ đưa lên mạng Internet.

- Em bé rất sợ tiếng ồn, mỗi khi nghe tiếng động lớn thì khóc.

Qua các ví dụ trên em hãy chỉ ra sự giống nhau và khác nhau giữa 2 khái niệm **Sự kiện** và **Cảm biến**. Hãy điền suy nghĩ của em vào bảng sau:

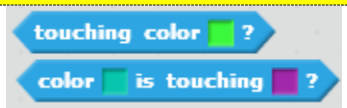
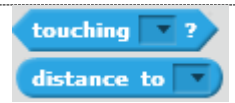
Vấn đề / câu hỏi	Sự kiện	Cảm biến
Hành động xuất phát từ bên ngoài hay từ bản thân nhân vật		
Sự việc có phụ thuộc vào bản thân nhân vật hay không?		
Sự việc có được lên kế hoạch trước hay không?		
Sự việc có được thực hiện theo thời gian định trước hay không?		
Sự việc có các tham số liên quan đến các giác quan con người hay không?		

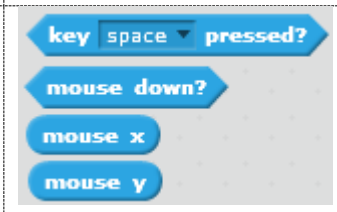
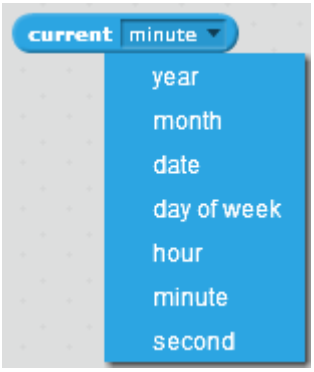
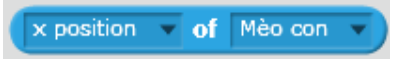
Nội dung bài học

1. Các câu lệnh cảm biến trong Scratch

Các lệnh cảm biến nằm trong nhóm lệnh Sensing (Cảm biến). Các lệnh này thường điều khiển, xử lý thông qua các thông số mang tính cảm nhận, định tính và không phụ thuộc vào các tác động bên ngoài (sự kiện). Chính vì vậy chúng ta gọi chúng là các lệnh cảm biến. Để so sánh cảm biến và sự kiện chúng ta thường nghĩ đến cảm nhận của các giác quan của con người và các sự kiện độc lập từ bên ngoài.

Trong Scratch, các lệnh cảm biến thường là các biểu thức, hàm số. Các hàm số này sẽ trả lại giá trị của các thông số mang tính "cảm biến", chỉ phụ thuộc vào nội tại nhân vật. Bảng sau cho ta một tổng quan về các hàm hay dùng nhất trong nhóm lệnh Cảm biến.

Loại cảm biến	Lệnh cảm biến	Ý nghĩa
Cảm biến màu sắc.		- Trả lại đúng hay sai nếu nhân vật va chạm với màu sắc này. - Trả lại đúng hay sai nếu nhân vật màu <1> va chạm với màu sắc <2>
Cảm biến khoảng cách, va chạm.		- Trả lại đúng nếu nhân vật va chạm với một trong các đối tượng sau: con trỏ chuột; cạnh sân khấu; nhân vật khác.

Loại cảm biến	Lệnh cảm biến	Ý nghĩa
		- Trả lại khoảng cách từ nhân vật đến 1 đối tượng khác: con trỏ chuột; nhân vật khác.
Cảm biến chuột và bàn phím.		- Trả lại đúng nếu phím <> được bấm, nếu không trả lại sai. - Trả lại đúng nếu người dùng nháy chuột. - Trả lại tọa độ X của vị trí con trỏ chuột (dùng với lệnh trên). - Trả lại tọa độ Y của vị trí con trỏ chuột (dùng với lệnh trên).
Cảm biến thời gian.	   	Giá trị thời gian hệ thống hiện thời. Các tham số bao gồm:  - Số thứ tự ngày hiện thời tính từ năm 2000. - Biến nhớ dùng để đếm thời gian bằng giây. Giá trị này sẽ được tính lại từ đầu sau lệnh **reset timer**.
Thông tin thuộc tính của nhân vật và sân khấu.		- Trả lại các giá trị, thuộc tính, biến nhớ riêng của một nhân vật.

Loại cảm biến	Lệnh cảm biến	Ý nghĩa
		
Lệnh hỗ trợ hội thoại nhập dữ liệu trực tiếp từ bàn phím.		Lệnh hỏi và nhận trả lời của người dùng được ghi vào biến nhớ **answer**.

2. Cảm biến màu sắc

Cảm biến màu sắc là những càm nhận, nhận biết liên quan đến màu sắc. Em hãy cùng thực hiện các hoạt động sau để hiểu các lệnh cảm biến liên quan đến màu sắc trong Scratch.

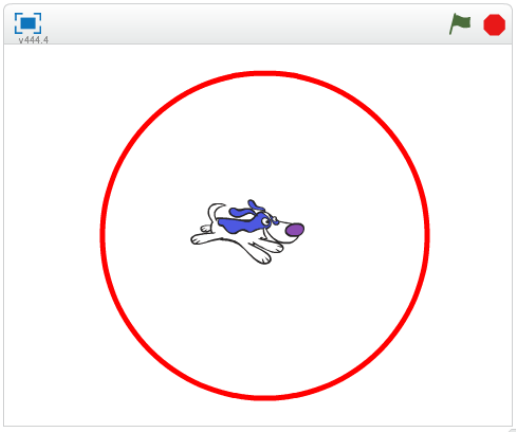
Nhân vật chính của chúng ta là 1 con chó nhỏ hiện đang nằm giữa sân khấu.

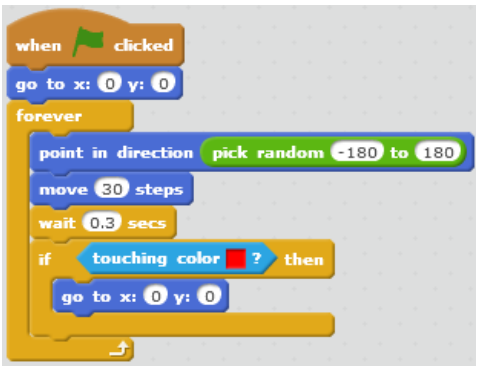
Yêu cầu thực hiện chương trình.

Cho con chó nhỏ chạy theo các hướng ngẫu nhiên, mỗi lần chạy 10 bước và nghỉ 0.3 giây. Nếu chó chạm vạch đỏ thì lập tức bị đặt vào vị trí tâm của vòng tròn, và chương trình lại tiếp tục.

Em hãy sử dụng lệnh cảm biến  để điều khiển nhận biết khi chó chạm vạch đỏ. Để chọn màu cảm biến chính xác em thực hiện: sau khi kéo thả lệnh trên ra cửa sổ lệnh, em nháy chuột lên ô màu bên phải lệnh, sau đó nháy chuột lên vạch đỏ trên sân khấu để chọn màu cảm biến.

Chương trình được mô tả trong hình dưới đây.





3. Cảm biến khoảng cách, va chạm

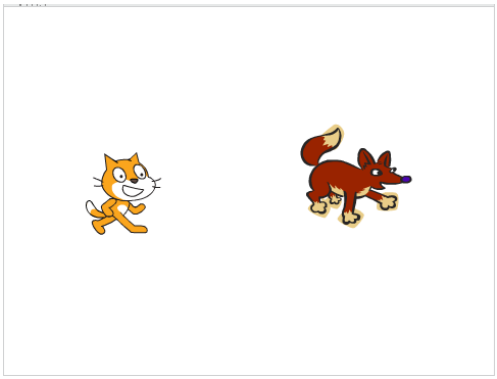
Trong môi trường Scratch có 2 lệnh liên quan đến cảm biến khoảng cách, va chạm.

Lệnh này sẽ trả lại giá trị đúng nếu nhân vật hiện thời va chạm với 1 nhân vật khác / hoặc va chạm với con trỏ chuột / hoặc va chạm với cạnh sân khấu. Ngược lại nó trả về giá trị sai.

Lệnh này sẽ tính và trả về giá trị khoảng cách từ nhân vật hiện thời đến 1 nhân vật khác / hoặc đến vị trí con trỏ chuột.

Chúng ta cùng tìm hiểu cảm biến khoảng cách, va chạm thông qua ví dụ sau.

Trò chơi Chó đuổi Mèo.

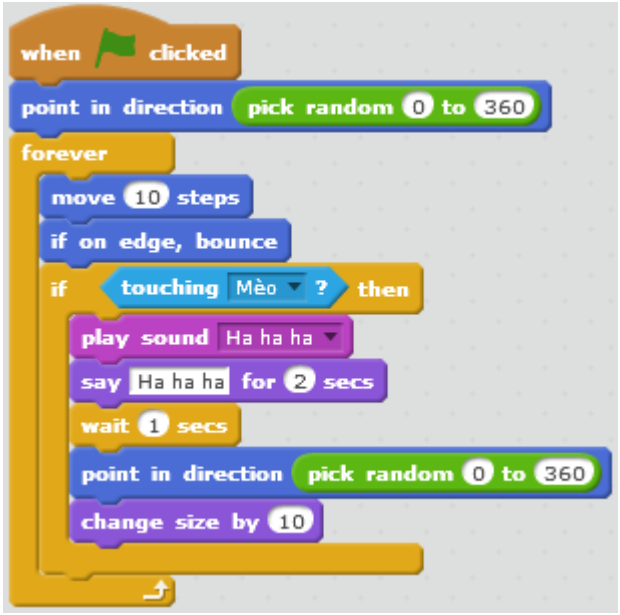


Trên sân khấu có 2 nhân vật chính là Chó và Mèo.

Mèo được điều khiển chuyển động bằng bàn phím. Người dùng sử dụng các phím up, down, left, right để dịch chuyển mèo theo các hướng lên, xuống, trái, phải tương ứng. Cần điều khiển để Mèo tránh gặp Chó.

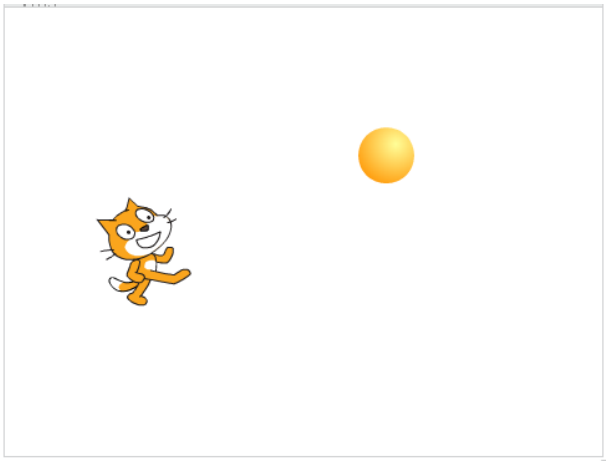
Chó chuyển động ngẫu nhiên trên màn hình. Nếu Chó va chạm với Mèo, Chó sẽ kêu "Ha ha ha" và tự to lên 10%.

Chương trình riêng cho Mèo và Chó được thể hiện trong hình sau. Có thể tạo thêm âm thanh tiếng kêu mừng rỡ của Chó khi bắt được Mèo.



4. Cảm biến chuột và bàn phím

Em hãy thiết kế trò chơi đơn giản Mèo đuổi Bóng.



Yêu cầu của trò chơi như sau:

Mèo sẽ luôn hướng đến Bóng và chạy về phía Bóng. Nếu bắt được bóng thì Mèo sẽ kêu lên tiếng "Bắt được bóng rồi" (có thể thêm âm thanh để trò chơi thêm hấp dẫn) và kết thúc.

Người chơi sẽ dùng chuột để điều khiển quả bóng. Nháy chuột lên màn hình sẽ lập tức làm cho Bóng dịch chuyển nhanh đến vị trí chuột.

Người dùng cũng có thể click và rê chuột trên màn hình để kéo thả Bóng chạy trốn khỏi Mèo.

Em cần sử dụng lệnh cảm biến 

và lệnh chuyển động 

để điều khiển Mèo luôn hướng về Bóng và chạy về phía Bóng.

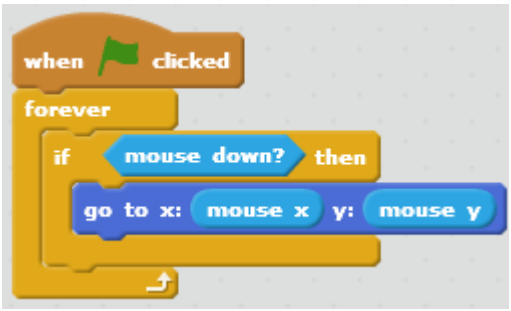
Với Bóng, em cần sử dụng các lệnh cảm biến chuột là 

để xác định thời điểm người dùng nháy chuột, dùng các hàm 

, 

để tính tọa độ chính xác của chuột tại thời điểm nháy chuột.

Chương trình dành cho Bóng (trái) và Mèo (phải) như hình dưới đây.



5. Cảm biến thời gian

Các hàm cảm biến thời gian trong Scratch bao gồm biến 

tự động đếm thời gian theo giây, và hàm 

trả lại các giá trị của thời gian hệ thống. Lệnh 

có tác dụng đặt lại từ đầu cho biến đếm 

.

Áp dụng: thiết kế chương trình mô tả đồng hồ kim chạy trên màn hình.

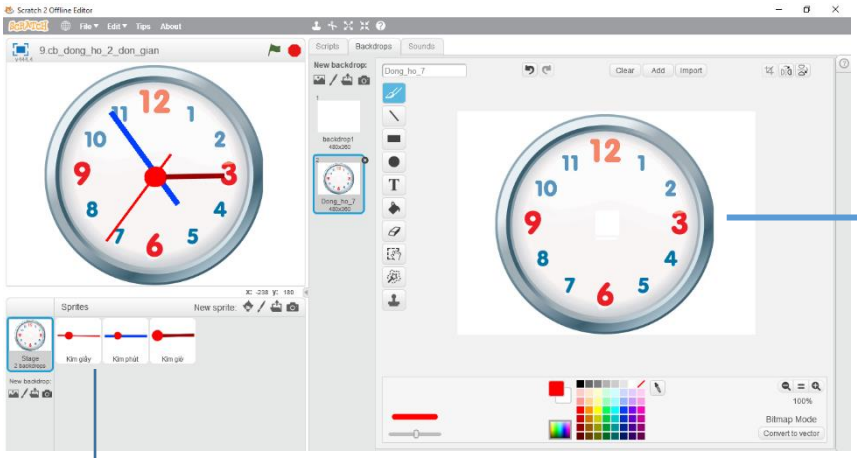
Thiết lập dự án chương trình như sau:

- Sân khấu nền là mặt đồng hồ kim (analog clock)

- Có 3 nhân vật là: kim giây, kim phút, kim giờ.

Chú ý thiết lập tâm của các kim này tại vị trí tâm hình tròn.

- Đưa tất cả các kim về vị trí xuất phát: tâm của các kim mày trùng với tâm của sân khấu.



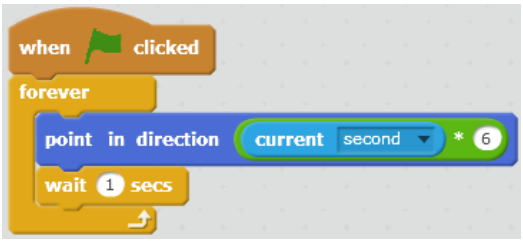
Vẽ hình nền sân khấu là mặt tròn của đồng hồ kim.

Thiết lập 3 nhân vật là kim giây, kim phút, kim giờ.

Để viết chương trình điều khiển, em sẽ viết cho từng kim đồng hồ. Vấn đề là tính góc quay của từng kim đồng hồ chính xác theo thời gian. Sử dụng hàm

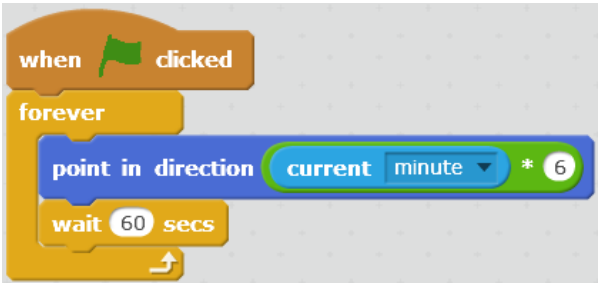
để tính giá trị của giây, phút, giờ theo thời gian hệ thống.

Chương trình cho kim giây.



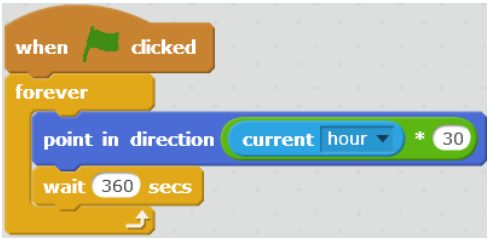
Thời gian theo giây có giá trị từ 0 đến 59, tương ứng với giá trị góc quay là 0 đến 360. Do đó mỗi giây, góc cần quay kim giây là <giá trị giây> * (360/60) =

Chương trình cho kim phút.



Thời gian theo phút có giá trị từ 0 đến 59, tương ứng với giá trị góc quay là 0 đến 360. Do đó mỗi phút, góc cần quay kim phút là <giá trị phút> * (360/60) =

Chương trình cho kim giờ.



Thời gian theo giờ có giá trị từ 0 đến 23, tương ứng với giá trị góc quay là 0 đến 360x2 = 720. Do đó mỗi giờ, góc cần quay kim giờ là <giá trị giờ> * (720/24) =

Câu hỏi: em có thấy chương trình chạy chính xác không?

6. Cảm biến âm thanh

Trong nhóm lệnh cảm biến của Scratch, biến **loudness** 

dùng để đo âm lượng âm thanh được đọc vào từ micro máy tính. Giá trị 

sẽ nằm trong khoảng từ 1 đến 100. Do vậy đây chính là **sensor**, máy đo cảm biến âm thanh của Scratch.

Cảm biến âm thanh được sử dụng rất nhiều trong các ứng dụng, và các phần mềm, trò chơi. Thay vì dùng bàn phím hay chuột để điều khiển nhân vật, chúng ta sẽ dùng loa, âm thanh để điều khiển. Xét 1 vài ví dụ.

1) Mô tả chuyển động trên màn hình với vận tốc phụ thuộc vào âm thanh.

sensor âm thanh làm tăng vận tốc chuyển động nhân vật.

Với đoạn chương trình trên, nếu người chơi (em và các bạn) vỗ tay cổ vũ càng nhiều cho nhân vật của chúng ta thì nhân vật sẽ càng chạy nhanh hơn trên màn hình.

2) Thể hiện kích thước nhân vật phụ thuộc vào âm thanh.

sensor âm thanh làm tăng kích thước nhân vật.

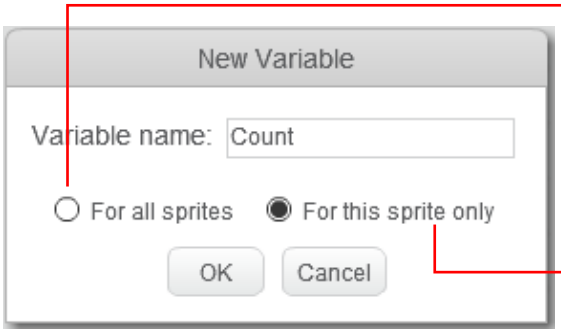
Đoạn chương trình trên nằm trong 1 chương trình luyện nháy chuột nhanh lên nhân vật. Để nhìn rõ hơn chúng ta phải hò hét, vỗ tay cổ vũ thì nhân vật sẽ tự động phóng to lên, ngược lại nếu không có ai cổ vũ nhân vật sẽ có kích thước rất nhỏ bé.

Các bài luyện tập sẽ có trong phần Câu hỏi và bài tập sau bài học này.

7. Kiểu biến nhớ: dùng chung và riêng

Biến nhớ chung và riêng.

Khi khởi tạo 1 biến nhớ trong Scratch, chúng ta cần lựa chọn 1 trong 2 kiểu, biến nhớ dùng chung và biến nhớ dùng riêng.



Biến nhớ dùng chung: tất cả mọi nhân vật đều có quyền nhập, thay đổi giá trị của biến nhớ này. Biến nhớ chung còn có tên gọi **Global Variable**.

Biến nhớ dùng riêng: chỉ nhân vật là chủ của biến này có quyền nhập, thay đổi giá trị, các nhân vật khác chỉ có quyền xem, khai thác giá trị. Biến nhớ riêng còn có tên gọi **Local Variable**.

Chú ý: Nếu chúng ta khởi tạo 1 biến nhớ riêng cho 1 nhân vật thì biến nhớ này chỉ xuất hiện trong bảng lệnh của nhân vật này mà không hiện trong bảng lệnh của các nhân vật khác. Tuy nhiên biến nhớ này sẽ xuất hiện như 1 thuộc tính của nhân vật này. Các nhân vật khác muốn truy cập giá trị này cần thực hiện thông qua lệnh, hàm cảm biến truy xuất thuộc tính của nhân vật.

Trong ví dụ trên, biến Count là riêng được khởi tạo cho nhân vật Lan thì các nhân vật khác sẽ truy cập biến nhớ này thông qua lệnh 

.

Câu hỏi và bài tập

1. Viết chương trình sau và giải thích hoạt động.

2. Mở rộng trò chơi Chó đuổi Mèo, bổ sung thêm biến đếm cho phép Chó va chạm vào Mèo 3 lần thì kết thúc chương trình.

3. Mở rộng trò chơi Mèo bắt Bóng, bổ sung thêm biến đếm thời gian từ khi bắt đầu cho đến khi Mèo bắt được bóng, thể hiện thời gian này trên màn hình.

4. Viết chương trình ngắn thực hiện công việc sau:

Trên màn hình hiện giá trị biến đếm (ví dụ đặt tên Count). Giá trị ban đầu Count = 0. Mỗi lần người dùng nháy chuột lên sân khấu thì biến Count tăng lên 1 đơn vị. Khi đủ 100 lần nháy chuột thì chương trình dừng lại.

5. Khi chạy chương trình sau em sẽ thấy điều gì?

6. Viết chương trình với yêu cầu màn hình luôn hiện thông tin ngày tháng năm của thời gian hiện tại, tương tự như hình dưới đây.

7. Mở rộng yêu cầu bài tập 6, màn hình luôn thể hiện chính xác giây, phút, giờ, ngày, tháng, năm của thời gian hệ thống máy tính.

8. Hai chương trình sau có tác dụng giống nhau hay không?

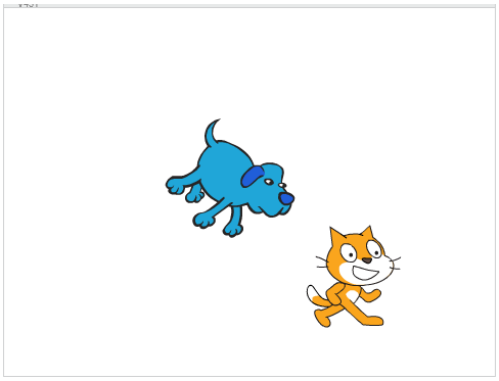
Hãy giải thích sự giống nhau và khác nhau giữa chúng.

9. Thiết lập các biến nhớ để tính tự động cập nhật các thông số sau liên quan đến nhân vật:

- Khoảng cách từ nhân vật đến các cạnh của sân khấu.

- Khoảng cách từ nhân vật đến tâm của sân khấu.

10. Viết chương trình đơn giản, vui nhộn sau:



Hai nhân vật chính của chương trình: **Chó cún** mà **Mèo con**.

Trên màn hình có 2 nhân vật: Chó cún và Mèo con. Chó cún đứng giữa sân khấu.

Mèo con sẽ chạy khắp sân khấu theo các hướng ngẫu nhiên, nếu gặp cạnh thì quay lại. Chó cún luôn nhìn về phía Mèo con, nếu thấy Mèo con va phải cạnh sân khấu thì kêu lên gâu gâu.

11. Viết chương trình mô phỏng trò chơi Thi chạy Marathon như sau:

Thi chay Marathon.sb2

Trên màn hình ban đầu có 1 trọng tài và 2 bạn học sinh như trên hình. Khi bắt đầu chơi, trọng tài nói chuẩn bị, và sau đó hô "Bắt đầu". Hai bạn sẽ chạy từ trái sang phải (với vận tốc ngẫu nhiên). Đến cạnh phải thì dừng lại. Phía trên có đồng hồ đo thời gian của 2 người chơi. Sau khi kết thúc, trọng tài sẽ tuyên bố tên người thắng cuộc.

Thi chay Marathon 2.sb2

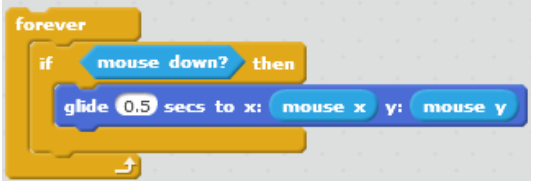
12. Mở rộng bài tập trên với thay đổi yêu cầu như sau:

Mỗi người chơi cần chạy 2 vòng sân khấu, chạy từ trái sang phải, gặp cạnh và quay lại chạy từ phải sang trái thì kết thúc đường chạy.

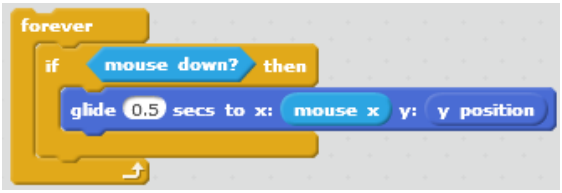
Các yêu cầu khác của trò chơi vẫn giữ nguyên.

13. Trong các chương trình có thể dùng chuột để điều khiển chuyển động của nhân vật. Hãy chỉ ra ý nghĩa của các đoạn chương trình điều khiển sau.

Chương trình 1:



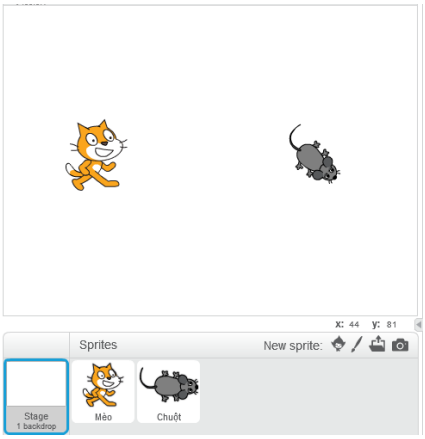
Chương trình 2:



Chương trình 3:

14. Viết chương trình **Mèo bắt Chuột** như sau:

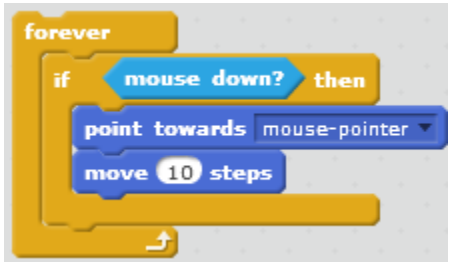
Meo bat chuot.sb2



- Mèo sẽ luôn hướng đến Chuột và chạy về phía Chuột để bắt Chuột.
- Chuột được điều khiển bởi chuột máy tính (xem chương trình bên dưới).
- Nếu Mèo bắt được Chuột thì chương trình kết thúc.

Mèo

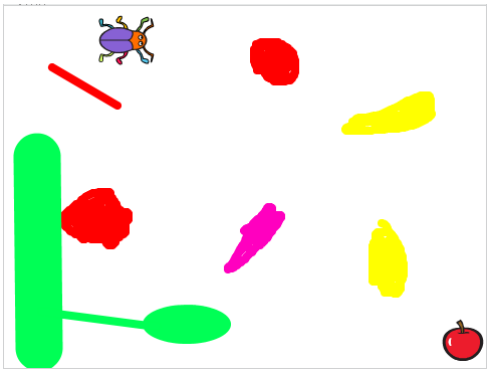
Chuột



Hãy giải thích hoạt động của Mèo và Chuột trong chương trình trên.

Mở rộng

1. Trò chơi **Ăn táo**. Viết chương trình điều khiển con Cánh cam bằng bàn phím, chuyển động đến vị trí Quả táo ở góc phải dưới sân khấu, dọc đường đi không được va chạm với bất cứ vết màu nào trên màn hình. Nếu chạm vào các vết màu này thì bị thua.



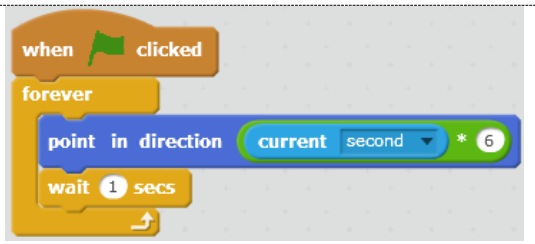
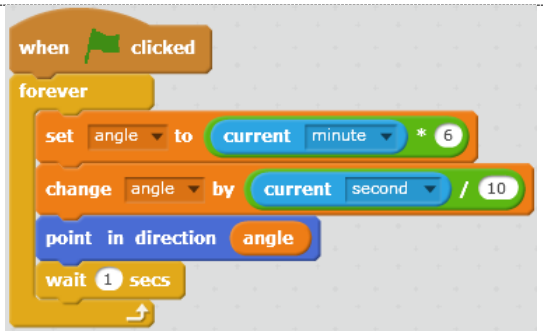
2. Mở rộng chương trình thể hiện đồng hồ kim chạy chính xác trên màn hình.

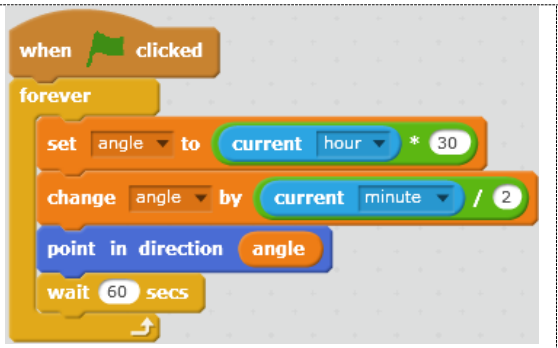
Gợi ý:

Cần điều khiển kim phút chạy theo từng giây và kim giờ theo từng phút.

Muốn vậy đối với kim phút và kim giờ cần thiết lập thêm biến nhớ (riêng) angle cho mỗi nhân vật này và tính toán chính xác góc cần quay theo thời gian.

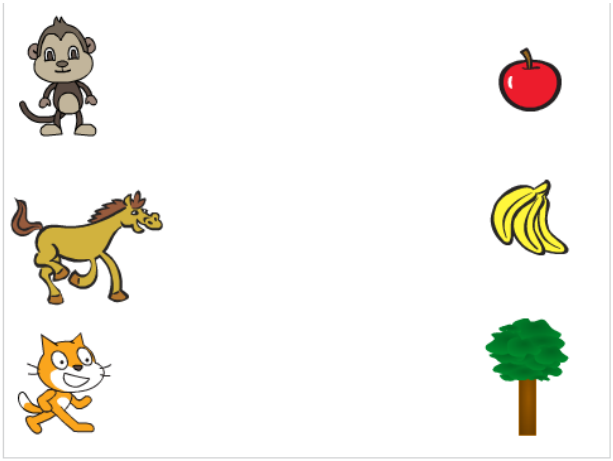
Bảng sau mô tả chi tiết các điều khiển các kim giờ, phút, giây chính xác.

Điều khiển kim giây theo từng giây.		Kim giây không cần thay đổi điều khiển, cập nhật theo từng giây. Thời gian theo giây có giá trị từ 0 đến 59, tương ứng với giá trị góc quay là 0 đến 360. Do đó mỗi giây, góc cần quay kim giây là <giá trị giây> * (360/60) = <giây>*6.
Điều khiển kim phút theo từng giây.		Nâng cấp điều khiển kim phút theo từng giây. Thời gian theo phút có giá trị từ 0 đến 59, tương ứng với giá trị góc quay là 0 đến 360. Do đó mỗi phút, góc cần quay kim phút là <giá trị phút> * (360/60) = <phút>*6.

Điều khiển kim giờ theo từng phút.		Nâng cấp điều khiển kim giờ theo từng phút. Thời gian theo giờ có giá trị từ 0 đến 23, tương ứng với giá trị góc quay là 0 đến 360x2 = 720. Do đó mỗi giờ, góc cần quay kim giờ là <giá trị giờ> * (720/24) = <giờ>*30.

3. Thiết kế trò chơi **Con gì ăn cái gì?**

Con gi an cai gi.sb2



Trên màn hình bên trái là các con Mèo, Khỉ, Ngựa. Bên phải là Táo, Chuối và Cây. Mỗi con chỉ ăn 1 thứ mà thôi: Khỉ ăn Chuối, Mèo ăn Táo, Ngựa ăn Cây.

Chương trình chơi như sau:

Người dùng lần lượt phải dùng chuột kéo thả các con vật ở bên trái sang đúng vị trí thức ăn của nó ở bên phải.

- Nếu kéo thả không đúng (chưa tới hoặc nhầm) thì con vật sẽ kêu "Sai rồi" và chạy về chỗ cũ.

- Nếu kéo thả đúng vị trí thì con vật sẽ dính vào thức ăn của mình và kêu "Ngon quá".

- Trò chơi kết thúc khi người chơi kéo thả đúng cả 3 con Mèo, Khỉ, Ngựa.

4. Thiết kế trò chơi đơn giản **Cổ vũ chạy đua** như sau.

Co vu chay dua.sb2

Chương trình hoạt động như sau:

- Hình trên là trạng thái ban đầu của cuộc thi chạy khi đã nháy nút lá cờ để chuẩn bị sẵn sàng cho cuộc chạy đua.

- Phía trên là các dữ liệu: bên trái là thời gian tính từ lúc bắt đầu chạy đến khi kết thúc. Bên phải là chỉ số vòng đua, nhân vật sẽ phải chạy quanh sân khấu 10 vòng. Đến vòng thứ 10 sẽ xuất hiện cột đích.

- Nháy nút Begin để bắt đầu cuộc đua. Nhân vật sẽ chạy từ trái sang phải và phải qua 10 lần sân khấu. Khi vượt qua cạnh bên phải, sân khấu sẽ chuyển sang vòng sau và người chạy xuất hiện lại từ bên trái.

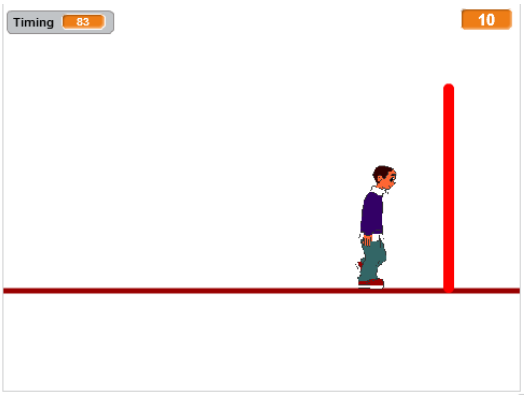
- Tốc độ chạy của nhân vật sẽ chậm. Muốn nhân vật chạy nhanh hơn thì phải cổ vũ bằng âm thanh, âm thanh càng lớn nhân vật chạy càng nhanh.

- Khi đến vòng 10 sẽ xuất hiện cột đích màu đỏ. Gặp đích, nhân vật sẽ kêu lên "Ura" và cuộc chạy kết thúc.

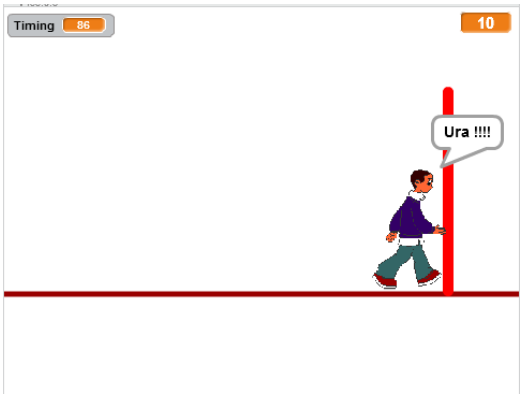
Các hình ảnh sau mô tả các trạng thái của nhân vật khi chạy.

H1. Lúc bắt đầu chương trình. Nháy lên nút **Begin** để bắt đầu chạy. Đồng hồ thời gian chỉ số 1 và bên phải chỉ vòng chạy 1.

H2. Trong quá trình chạy từ vòng 2 đến vòng 9. Đồng hồ thời gian luôn chạy. Người chơi chạy từ trái sang phải màn hình, rồi lại quay lại chạy từ bên trái.



H3. Ở vòng 10 sẽ xuất hiện cột màu đó chính là đích.



H4. Khi người chơi chạm cột đỏ sẽ đứng lại, kêu Ura và chương trình kết thúc.

CHƯƠNG 4: SCRATCH NÂNG CAO

Chương 4 có thể coi là phần trung tâm, dài nhất và quan trọng nhất của cuốn sách. Phần này sẽ cung cấp một số kiến thức lõi về xử lý dữ liệu, thuật toán và giải quyết vấn đề. Phần kiến thức này được xếp vào mức nâng cao vì nó chỉ phù hợp với học sinh từ cấp THCS trở lên. Chương này sẽ bao gồm 9 bài học và được tách theo các nhóm nội dung riêng biệt: các thuật toán và bài toán xử lý số, xử lý xâu ký tự, làm việc với mảng số, thủ tục và cuối cùng là clone. Cũng trong chương này sẽ lần đầu tiên đưa khái niệm Sơ đồ hoạt động chương trình như 1 sơ đồ khối thiết kế hệ thống cho 1 chương trình hoàn chỉnh.

- Các thuật toán với số như các phép chia, chia có dư với số nguyên, khai triển số nguyên tố, chuyển đổi số thập phân sang nhị phân và ngược lại. Thuật toán tính ƯSCLN và BSCNN của 2 số tự nhiên.

- Các bài toán và thuật toán với xâu ký tự: tìm xâu con, hoán vị ngẫu nhiên xâu ký tự, tách và ghép xâu, tìm xâu đối xứng.

- Các bài toán và thuật toán với dãy số: tìm phần tử và giá trị Max, Min của dãy, hoán vị ngẫu nhiên dãy số, các thuật toán sắp xếp thứ tự dãy, tìm dãy con theo các tiêu chí khác nhau.

- Khái niệm thủ tục và thủ tục có tham số. Biến nhớ tổng thể và địa phương. Bước đầu học sinh được làm quen với thủ tục đệ qui.

- Giới thiệu mô hình và kỹ thuật lập trình Clone. Clone là mô hình mô phỏng lập trình đối tượng rất đặc sắc của Scratch. Sử dụng thành thạo kỹ thuật này, học sinh sau này sẽ dễ dàng tiếp cận với các ngôn ngữ lập trình bậc cao.

Danh sách các bài học trong chương này:

Bài 12. Xử lý số 1.

Bài 13. Xử ký số 2.

Bài 14. Xử lý xâu ký tự 1.

Bài 15. Xử lý xâu ký tự 2.

Bài 16. Làm việc với List 1.

Bài 17. Làm việc với List 2.

Bài 18. Thủ tục 1.

Bài 19. Thủ tục 2.

Bài 20. Clone 1. Phân thân của nhân vật.

Bài 20. Clone 2. Thuộc tính của Clone.

Clone (phân thân) là một tính chất công nghệ đặc biệt của Scratch. **Clone** được dùng trong rất nhiều bài toán lập trình Scratch. Là một trong những kiến thức khó nhất và sâu sắc nhất của Scratch, **clone** là hình ảnh mô tả rõ nhất những khái niệm lập trình hướng đối tượng mà bình thường rất khó truyền đạt cho học sinh. Khi các

em hiểu rõ được vai trò và ý nghĩa của **clone** thì sau này khi học sang các ngôn ngữ lập trình bậc cao khác, những khái niệm như thừa kế, đa hình sẽ trở nên rất dễ hiểu.

Bài 12. Xử lý số 1

Mục đích

- Biến nhớ và vai trò của biến nhớ.
- Phân biệt biến nhớ và biểu thức.
- Các phép tính, tính toán đơn giản với giá trị số và logic.
- Thực hiện một vài bài bài toán xử lý số đơn giản.

Bắt đầu

1. Chúng ta nhớ lại 2 đoạn chương trình vẽ hình vuông và ngũ giác đều đã học trong bài học **Vẽ hình 2**.

Vẽ hình vuông

Vẽ hình ngũ giác đều

Em có suy nghĩ gì về việc có thể tổng quát hóa đoạn chương trình này để vẽ 1 hình đa giác đều n cạnh bất kỳ?

2. Các biểu thức số học, ví dụ:

$$\frac{7}{8}+\frac{9}{10}$$

$$7+\frac{5}{6}-\frac{23}{50}$$

$$\sqrt{2}+\frac{1}{2}45-\cos 15$$

sẽ được tính toán như thế nào trong Scratch?

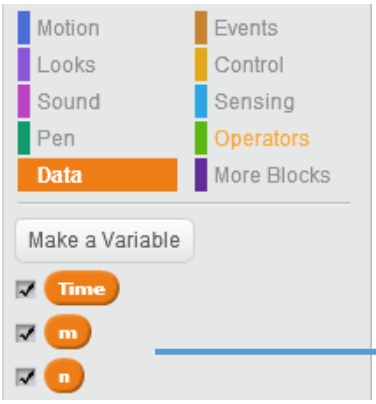
Nội dung bài học

1. Biến nhớ và hàm số

Chúng ta đã làm quen với biến nhớ và các lệnh trả lại giá trị (hay còn gọi là hàm số) trong Scratch. Hoạt động này là một củng cố kiến thức đã biết.

Định nghĩa biến nhớ:

Biến nhớ là 1 vùng trong bộ nhớ, được đặt tên bằng một dãy chữ và số, dùng để lưu trữ các giá trị (số, chữ hoặc logic). Biến nhớ có thể do hệ thống tạo sẵn hoặc do người dùng tự tạo ra để giúp giải quyết các bài toán lập trình.



Các biến nhớ do người lập trình tạo ra sẽ thể hiện trong khung lệnh **Dữ liệu (Data)**.

Qui định đặt tên biến nhớ: Scratch không đặt các điều kiện chặt cho cách đặt tên biến nhớ. Tên biến nhớ phân biệt chữ hoa, chữ thường và có thể chứa dấu cách. Tuy nhiên với các ngôn ngữ lập trình bậc cao thì tên biến nhớ thường được qui định khá chặt chẽ. Các em cần nhớ 1 số qui tắc chung này để áp dụng cho các ngôn ngữ lập trình bậc cao trong tương lai.

Tên biến nhớ phải là dãy chữ hoặc số, không có dấu cách, không nên chứa các ký tự đặc biệt như : ; , \ / + - [ ] { } ( ). Nên đặt tên biến nhớ là 1 cụm từ có ý nghĩa để dễ nhớ. Không nên đặt tên biến nhớ với quá nhiều chữ hoa

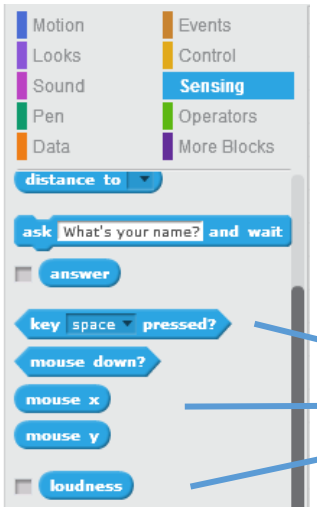
Ví dụ tên biến nhớ như sau là hợp lệ: x1, abc, solution, time_out, click_count, con_meo, chim_bay, thoi_gian, ….

Các tên biến nhớ như sau thì không nên và không hợp lệ: 12+23, sinx, ANHSang, C://thu_muc, ….

Chú ý: các biến nhớ dùng chung cần có tên khác nhau từng đôi một trong chương trình. Biến nhớ riêng nhân vật có thể đặt tên trùng nhau.

Định nghĩa hàm số:

Hàm số do người dùng hoặc hệ thống tạo ra, được đặt tên bằng một dãy chữ và số, dùng để tính toán một giá trị nào đó, được lưu trữ trong bộ nhớ và được sử dụng tương tự như các biến nhớ trong các câu lệnh.



Các hàm số được tạo sẵn trong hệ thống, mang ý nghĩa khác nhau và đều trả lại các giá trị có ý nghĩa trong quá trình giải quyết vấn đề.

Hàm số có thể có hoặc không có các tham số đầu vào. Ví dụ.

- Hàm 

trả lại tọa độ X của vị trí con chuột hiện thời. Hàm này không có tham số.

- Hàm 

có 1 tham số là tên phím tương ứng trên bàn phím. Hàm sẽ trả lại giá trị Đúng nếu phím này được nhấn, và trả lại Sai nếu phím này không được nhấn.

- Hàm tính tổng 

có 2 tham số đầu vào là 2 số. Hàm sẽ trả lại giá trị là tổng của hai số này. Tham số đầu vào có thể lấy giá trị cụ thể hoặc lấy từ các biến nhớ hoặc hàm số khác. Ví dụ các cách truyền tham số cho hàm tính tổng này:


, 

, 

.

Biến nhớ có rất nhiều ý nghĩa trên thực tế. Chúng ta hãy cùng xét 1 ví dụ sau, lấy cảm hứng từ ví dụ vẽ các hình tứ giác, ngũ giác đều trong phần mở đầu của bài học này.

Sử dụng các biến nhớ **so_canh** để lưu số cạnh của đa giác muốn vẽ, biến nhớ **do_dai_canh** để lưu độ dài cạnh đa giá, khi đó đoạn chương trình vẽ 1 đa giác đều tổng quát như sau:

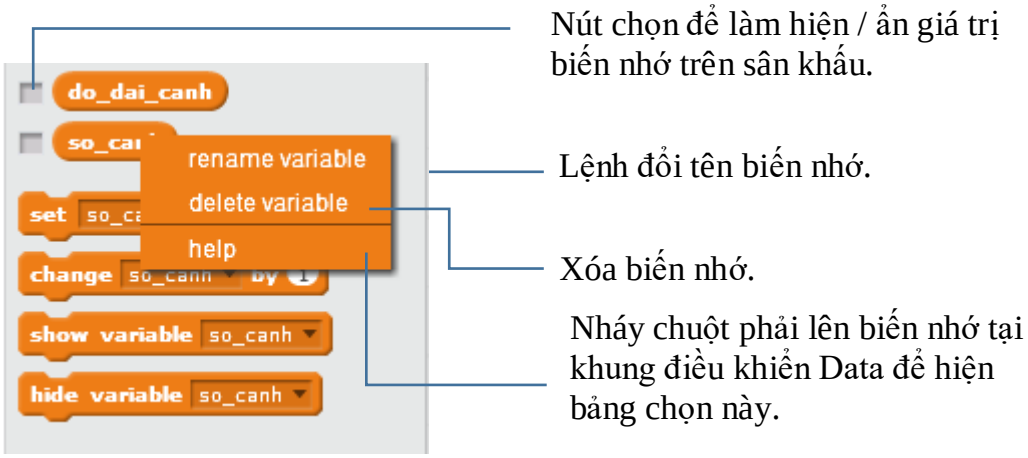
2. Các thao tác làm việc với biến nhớ

Đối với biến nhớ, 2 lệnh quan trọng nhất là lênh gán giá trị và lệnh thay đổi giá trị của biến nhớ. Các lệnh này được thực hiện trong Scratch như sau, so sánh với cách viết trong 1 số ngôn ngữ lập trình bậc cao khác.

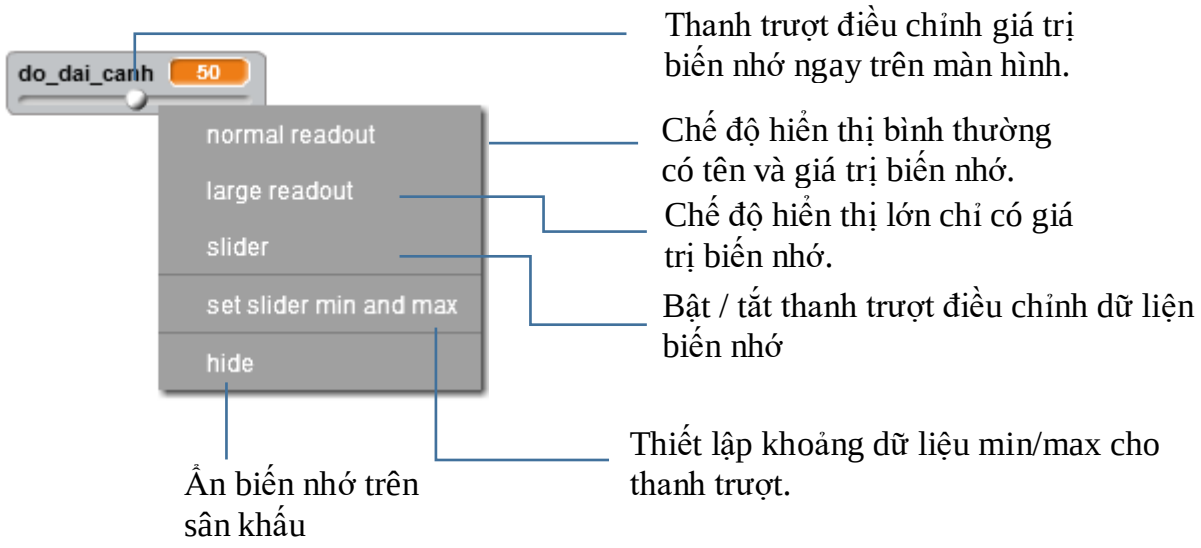
Lệnh	Scratch	Pascal	C, Java
Gán giá trị cho biến nhớ		`m:=10;`	`m = 10;`
Thay đổi giá trị của biến nhớ + 1		`m:=m+1;`	`m++;`
Thay đổi giá trị của biến nhớ - 1		`m:=m-1;`	`m--;`

Một số lệnh, thao tác khác với biến nhớ.

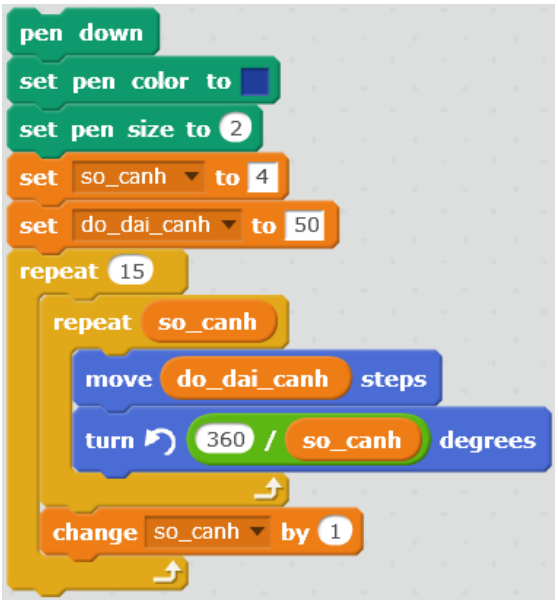
Hiện giá trị biến nhớ trên sân khấu, xóa biến nhớ, đổi tên biến nhớ.

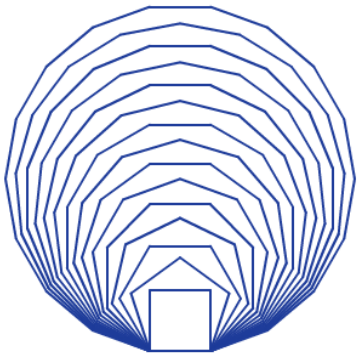


Khi biến nhớ đã được hiển thị trên sân khấu thì có thể thực hiện các thao tác sau bằng cách nháy chuột phải lên vị trí biến nhớ.



Đoạn chương trình sau mô tả bài toán vẽ liên tục các hình đa giác đều với số cạnh tăng dần từ 4 đến 18. Kết quả thể hiện trong hình phải.





3. Kiểu dữ liệu trong Scratch

Khi 1 biến nhớ được khởi tạo trong Scratch, chúng ta không cần khai báo biểu dữ liệu cho biến nhới này. Trong quá trình làm việc, tùy thuộc vào dữ liệu nhập, biến nhớ tự động nhận biết 1 trong 3 kiểu dữ liệu sau:

Kiểu dữ liệu trong Scratch.

Số (Number). Bao gồm số nguyên và thập phân.

Logic (Boolean). Kiểu dữ liệu chỉ có 2 giá trị Đúng (true) và Sai (false).

Xâu (String). Kiểu dữ liệu là dãy các chữ, kí tự.

Ví dụ nếu biến nhớ  được khởi tạo thì có thể gán cho n các giá trị sau:

	Gán giá trị số cho n
	Gán xâu ký tự cho n
	Gán giá trị logic cho n

Em hãy thực hiện các lệnh sau và điền kết quả vào cột phải trong bảng dưới đây, từ đó có nhận xét gì về các kiểu dữ liệu và cách thể hiện chúng trong Scratch.

Biểu thức, lệnh	**Kết quả**
	
	
	
	

4. Các phép tính đơn giản với số trong Scratch

Nhóm lệnh Operators (màu xanh lá cây) bao gồm các lệnh, phép tính làm việc với dữ liệu số, logic và xâu ký tự. Trong bài học này chúng ta làm quen với các phép tính đơn giản với số.

Phép tính cộng, trừ nhân, chia số thập phân

 (hàm) phép cộng 2 số

 (hàm) phép trừ 2 số

 (hàm) phép nhân 2 số

 (hàm) phép chia 2 số, kết quả thập phân.

Làm tròn số

 (hàm) làm tròn số của 1 số, lấy số nguyên gần nhất số này.

 (hàm) làm tròn (lên) của 1 số, lấy số nguyên gần nhất trên số này.

 (hàm) làm tròn (dưới) của 1 số, lấy số nguyên gần nhất dưới số này.

Phép chia số nguyên

 (hàm) lấy số dư của 2 số nguyên

 (hàm) lấy thương nguyên của 2 số nguyên

Phép tính lũy thừa

 (hàm) lấy lũy thừa của 2 số thập phân n^a

(hàm) lấy lũy thừa của 2 số nguyên n^a

Phép lấy căn thức, trị tuyệt đối

(hàm) lấy giá trị tuyệt đối của 1 số.

(hàm) lấy căn bậc 2 của 1 số dương a ($a^{1/2}$)

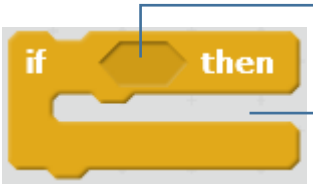
(hàm) lấy căn bậc n của 1 số a ($a^{1/n}$)

Ví dụ:

1/2 + 3/4	
$m^2 + n^2 - mn$	
| x | + | x + 2| - | x - 1|	
$x^2 - x + 1$	
$(a + b)^2$	

5. Kiểu dữ liệu logic

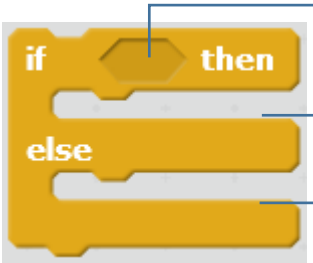
Em đã được làm quen với lệnh điều khiển có điều kiện **if .. then**. Biểu thức sau **if** là 1 điều kiện logic. Khi thực hiện, lệnh sẽ kiểm tra điều kiện này, nếu **đúng (true)** thì sẽ thực hiện đoạn lệnh sau từ khóa **then**, nếu **sai (false)** thì bỏ qua lệnh này.



<điều kiện> = biểu thức logic

đoạn chương trình sẽ thực hiện nếu **<điều kiện>** này **đúng (true)**.

Tương tự chúng ta có 1 lệnh tổng quát hơn: lệnh điều khiển đầy đủ có điều kiện **if .. then … else**. Biểu thức sau **if** là 1 điều kiện logic. Khi thực hiện, lệnh sẽ kiểm tra điều kiện này, nếu **đúng (true)** sẽ thực hiện đoạn lệnh sau từ khóa **then**, nếu **sai (false)** sẽ thực hiện đoạn lệnh sau từ khóa **else**.



<điều kiện> = biểu thức logic

đoạn chương trình sẽ thực hiện nếu **<điều kiện>** này **đúng (true)**.

đoạn chương trình sẽ thực hiện nếu **<điều kiện>** này **sai (false)**.

Các toán tử, phép tính với biểu thức và biến nhớ logic trong Scratch bao gồm.

Phép toán logic	Kết quả
	(hàm) toán tử logic **and (và)**. Trả lại true (đúng) khi và chỉ khi cả 2 biểu thức thành phần đều true (đúng).
	(hàm) toán tử logic **or (hoặc)**. Trả lại true (đúng) nếu 1 trong 2 biểu thức thành phần là true (đúng), các trường hợp khác sẽ trả lại false (sai).
	(hàm) toán tử **not (phủ định / không)**. Trả lại true (đúng) nếu biểu thức là false (sai) và ngược lại, trả lại false nếu biểu thức là true (đúng).

Bảng cụ thể các phép tính logic **and, or, not** như sau.

Toán từ and (và)

A	B	A and B
true	true	true
true	false	false
false	true	false
false	false	false

Toán từ or (hoặc)

A	B	A or B
true	true	true
true	false	true
false	true	true
false	false	false

Toán tử not (không)

A	not A
true	false
true	true

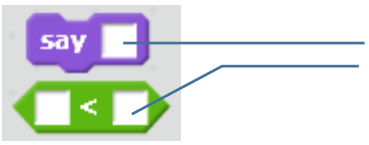
6. Biểu diễn biểu thức logic

Em hãy biểu diễn các điều kiện logic sau trong môi trường Scratch.

a > 10 và a < 100	
m >= 100	
0 < m < 20	
ngay = 10 và thang = 3 và nam = 2016	
m là số lẻ và 0 < m < 100	
m là số chẵn và m > 20	

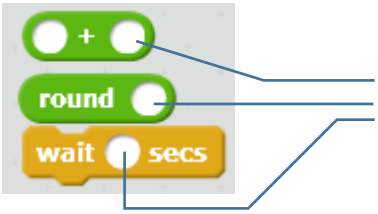
Chú ý đến các ô nhập trực tiếp dữ liệu trên các dòng lệnh. Có 2 loại ô chữ nhật và ô tròn.

Ô vuông, chữ nhật



Ô dữ liệu vuông, chữ nhật có thể nhập số, chữ hoặc biểu thức logic.

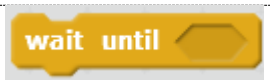
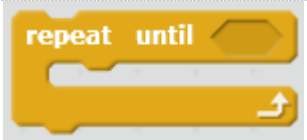
Ô tròn



Ô dữ liệu tròn chỉ thể nhập số hoặc biểu thức logic, không nhập được chữ.

7. Các lệnh điều khiển có sử dụng biểu thức logic

Chúng ta cùng xem lại tất cả các câu lệnh điều khiển (trong nhóm lệnh Điều khiển) có sử dụng các biểu thức logic, điều kiện.

Câu lệnh	Ý nghĩa
	Kiểm tra <điều kiện> trong ô của lệnh, nếu đúng thì thực hiện nhóm lệnh trong khung lệnh. Nếu <điều kiện> sai thì bỏ qua lệnh này.
	Kiểm tra <điều kiện> trong ô của lệnh, nếu đúng thì thực hiện nhóm lệnh đầu tiên trong khung lệnh, nếu sai thì thực hiện nhóm lệnh thứ hai trong khung lệnh.
	Tạm dừng chương trình và kiểm tra <điều kiện>. Nếu <điều kiện> đúng thì tiếp tục chạy lệnh tiếp theo sau lệnh này.
	Kiểm tra <điều kiện> trong ô của lệnh, nếu sai thì thực hiện nhóm lệnh trong khung lệnh, sau đó lại kiểm tra <điều kiện>, quá trình này lặp lại cho đến khi <điều kiện> là đúng thì chuyển sang lệnh tiếp theo sau lệnh này.

Chú ý: Lệnh **repeat until <điều kiện>** trong Scratch có ý nghĩa tương tự như các lệnh **while <điều kiện> do** trong các ngôn ngữ lập trình bậc cao khác như C++, Java, ….

Sau đây là 1 vài ví dụ minh họa cho các lệnh điều khiển sử dụng biểu thức logic trên.

a) Đoạn chương trình ngắn dùng để đếm thời gian. Việc đếm thời gian này rất có ý nghĩa và được ứng dụng nhiều trong các phần mềm trò chơi. Thay vì cần 1 biến nhớ dạng count down để đếm ngược thời gian, chúng ta sẽ dùng biến timer để đếm thời gian, dùng lệnh **wait until** để điều khiển công việc này.

Đặt biến timer = 0

Chờ đúng 20 giây

Thông báo kết thúc cuộc chơi

b) Chương trình yêu cầu người dùng liên tục thực hiện việc nhập tên từ bàn phím, quá trình này chỉ dừng khi người dùng bấm phím Enter (không nhập tên).

Gán giá trị ban đầu cho Name bất kỳ khác rỗng.

Ô này để rỗng, không nhập bất cứ giá trị nào.

Câu hỏi và bài tập

1. Viết hàm kiểm tra xem năm **year** có phải là năm nhuận không?

Tính chất toán học của năm nhuận: là năm chia hết cho 4 và không chia hết cho 100, hoặc chia hết cho 400.

2. Viết chương trình nhập hai giá trị month (tháng) và year (năm) từ bàn phím và kiểm tra xem giá trị nhập vào có hợp lệ hay không?

3. Biểu diễn các biểu thức sau dùng lệnh Scratch\

`(7/8) + (9/10)`

$ax^2 + bx + c$

`m*n - 2(m+n)`

`|x| - |2x - 1|`

$10^2 + (3+ 7/8)$

4. Biểu diễn các điều kiện logic sau bằng lệnh Scratch

`(m > 100) và (n < 100)`

`mn < 0`

$(m + n) > (m^2 + n^2)$

`(x >= 2) hoặc (y <= 10)`

`m không chia hết cho 4`

`m không bằng 0`

5. Viết chương trình hiển thị trên màn hình thời gian đầy đủ hệ thống, vi dụ:

`Ngày 16 tháng 5 năm 2016`

12 giờ 45 phút 24 giây

6. Viết chương trình nhập từ bàn phím 3 số: Ngày - Tháng - Năm và kiểm tra xem bộ 3 số này có hợp lệ hay không.

7. Em hãy dùng lệnh để viết các biểu thức số sau bằng lệnh của Scratch.

Biểu thức	**Lệnh Scratch**
Khoảng cách từ nhân vật đến cạnh trên của sân khấu.	
Khoảng cách từ nhân vật đến cạnh dưới của sân khấu.	
Khoảng cách từ nhân vật đến cạnh trái của sân khấu.	
Khoảng cách từ nhân vật đến cạnh phải của sân khấu.	
Khoảng cách từ nhân vật đến tâm của sân khấu.	
Khoảng cách giữa 2 nhân vật Chó và Mèo. Chú ý cách viết biểu thức sẽ khác nhau trong cửa sổ lệnh của Chó hay Mèo.	
Hôm nay là thứ 7 hoặc chủ nhật (là ngày nghỉ cuối tuần).	

8. Em hãy dùng lệnh để viết các biểu thức logic sau bằng lệnh của Scratch.

Biểu thức	**Lệnh Scratch**
Khoảng cách từ nhân vật hiện thời đến Chó cún > 200.	
Mèo con không va chạm với Chó cún.	
Chó cún kêu vượt quá ngưỡng 80%	
Khoảng cách đến tâm của sân khấu vượt quá giới hạn d.	
Hôm nay không là chủ nhật.	
Biểu thức logic kiểm tra xem a, b, c có tạo thành 3 cạnh của tam giác hay không.	
a + b chia hết cho c.	
a + b + c chia cho d dư 1.	

9. Em hãy dùng lệnh (hoặc nhóm lệnh) Scratch để mô tả các hành động sau:

Hành động	**Lệnh Scratch**
Chó luôn hướng mặt về phía Mèo.	
Chó kêu gâu gâu nếu Mèo va phải Chó.	

Hành động	Lệnh Scratch
Chuột nhìn thấy Mèo ở cự li gần < 50 sẽ quay đầu chạy mất.	
Nếu nháy chuột lên Mèo thì Mèo sẽ kêu meo meo và thay đổi trang phục mới.	
Mèo chạy ngẫu nhiên lăng xăng trên sân khấu, nếu va phải cạnh sân khấu thì kêu "ối đau quá" và đứng lại 1 giây trước khi quay lại chạy tiếp.	
Tạm dừng chương trình trong 10 giây.	
Đợi sau 20 giây nữa sẽ dừng toàn bộ chương trình.	

10. Viết các biểu thức toán học sau bằng lệnh hoặc nhóm lệnh Scratch.

Biểu thức	Lệnh Scratch
πR^2	
$\frac{1}{1^2}+\frac{1}{2^2}+\cdots+\frac{1}{N^2}$	
$m^2 + n^2$, với m, n lấy ngẫu nhiên trong khoảng từ 1 đến 100.	
$\sqrt{(a^2+b^2+c^2)}$	
$(a + b)^2 > 2ab$	
$\lvert 1 + \sin x\rvert + \lvert 1 + \sin(2x)\rvert + \ldots + \lvert 1 + \sin(100x)\rvert$	

11. Viết chương trình nhập từ bàn phím số tự nhiên n, sau đó tính và thông báo tổng 1 + 2 + …. + n.

12. Viết chương trình nhập từ bàn phím số tự nhiên n, sau đó tính và thông báo tổng $1^3 + 2^3 + \ldots + n^3$.

13. Viết chương trình nhập từ bàn phím 3 số dương a, b, c sau đó chương trình sẽ đưa ra thông báo:

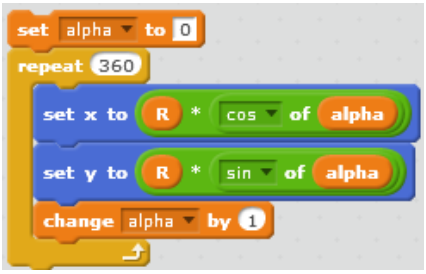
Có thể lập thành các cạnh của tam giác

hoặc

Không thể lập thành các cạnh của tam giác

14. Đoạn chương trình sau có chức năng gì?

15. Đoạn chương trình sau thực hiện công việc gì?

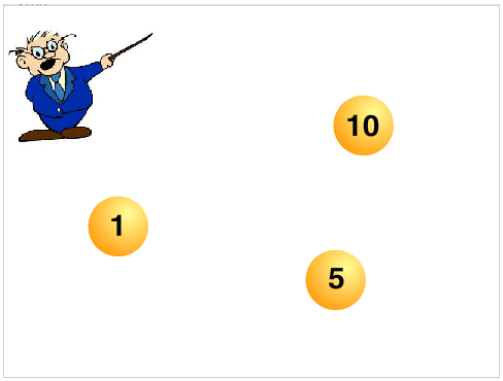


16.

Mở rộng

1. Thiết kế trò chơi đơn giản sau:

Game click bong.sb2



Trên sân khấu có 3 quả bóng với các số hiệu 1, 5, 10 chuyển động ngẫu nhiên, lúc ẩn lúc hiện trên màn hình.

Nhiệm vụ của em là nháy chuột chính xác lên các quả bóng, em sẽ được tích lũy điểm theo số được ghi trên quả bóng.

Thời gian cho mỗi lần chơi là 10 giây. Bạn nào đạt được điểm cao thì càng giỏi. Khi kết thúc 1 lần chơi, giáo viên sẽ thông báo điểm của em trên màn hình.

2. Viết chương trình trò chơi số đơn giản sau.

Game 3 canh tam giac.sb2

Trên màn hình chỉ có 1 nhân vật là giáo viên và 3 biến nhớ a, b, c hiện trên màn hình ở dạng cho phép người chơi được nhập và điều chỉnh. Giá trị các biến này chạy từ 1 đến 100.

Nhiệm vụ của người chơi là nhập các giá trị a, b, c trực tiếp trên màn hình và nháy lên giáo viên để kiểm tra xem nhập được đúng chưa, 3 số này có lập thành 3 cạnh của tam giác hay không.

Giáo viên sẽ thông báo dạng như sau:


Đúng rồi. Tạo được 1 tam giác !


Sai rồi. Không thể tạo được 1 tam giác !

Bài 13. Xử lý số 2

Mục đích

- Một vài thuật toán đơn giản với số nguyên, phân số.
- Bài toán tìm ước số, tìm số nguyên tố, tìm ước số chung của hai số.
- Bài toán rút gọn phân số, tính ước số chung lớn nhất, bội số chung nhỏ nhất.
- Thiết lập 1 vài trò chơi số đơn giản.

Bắt đầu

1. Em đã bao giờ đặt bút tính 100! chưa? Thử lấy bút và tính xem có khó không?
2. Em hãy nhớ lại một số khái niệm về số học để chuẩn bị cho bài học mới này.

- Thế nào là số nguyên tố, hợp số?
- Khai triển 1 số tự nhiên thành tích các thừa số nguyên tố.
- Khái niệm và cách tính ƯSCLN và BSCNN của 2 số tự nhiên.
- Thế nào là 1 phân số tối giản?

Nội dung bài học

1. Một số thuật toán số đơn giản

*Kiểm tra **n** có chia hết cho **m** hay không?*

Sử dụng hàm lấy số dư phép chia (mod): 

. Nếu giá trị này = 0 thì n chia hết cho m, ngược lại n không chia hết cho m.

*Kiểm tra **a** có phải là ước số thực sự của **m** hay không? (Ước thực sự là ước số khác 1 và chính số đó).*

Hàm kiểm tra như sau:

*Kiểm tra năm **year** có phải là năm nhuận hay không? (năm nhuận là năm chia hết cho 4 và không chia hết cho 100, hoặc chia hết cho 400).*

Hàm kiểm tra năm nhuận như sau:

Em hãy viết tiếp các hàm kiểm tra sau:

1. *Hàm kiểm tra hai phân số m/n và p/q có bằng nhau hay không?*
2. *Hàm kiểm tra số p có là bội số của cả hai số n, m hay không?*

3. *Hàm kiểm tra số p có là ước số đồng thời của cả hai số n, m hay không?*

4. *Hàm kiểm tra số p có phải là nghiệm của phương trình bậc nhất px + q = 0 hay không?*

2. Bài toán tìm các ước số thực sự của 1 số tự nhiên cho trước

Bài toán: yêu cầu nhập 1 số tự nhiên n từ bàn phím, chương trình sẽ thông báo tất cả các ước số thực sự của n trên màn hình.

Ví dụ nhập số 100 thì chương trình thông báo các giá trị 1, 2, 5, 10, 20, 25, 50 trên màn hình.

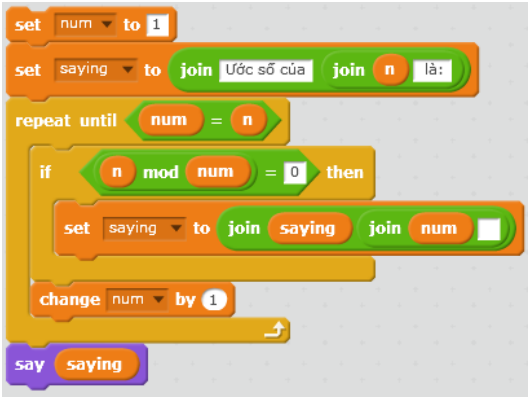
Ý tưởng cách giải (thuật toán) của bài toán này như sau:

Thiết lập 1 biến chạy num, cho num chạy từ 1 đến n-1 (chú ý không tính khi num = n), với mỗi bước lặp, kiểm tra xem num có phải là ước số của n hay không. Nếu đúng thì đưa num vào DS các số sẽ được in ra kết quả.

Mô tả thuật toán này một cách tường minh:

```
Gán num = 1
Lặp cho đến khi num = n
     Nếu n mod num = 0 thì
          thông báo num
     tăng n lên 1 đơn vị
```

Để viết chương trình trên Scratch, chúng ta thiết lập 2 biến nhớ **num** và **saying**. Biến num để chạy và kiểm tra ước số của n. Biến **saying** để lưu lại các ước cụ thể và hiển thị thông báo ra màn hình.



Câu hỏi: trong bài toán trên nếu thay đổi yêu cầu "ước số thực sự" bằng "ước số" thì chương trình sẽ phải thay đổi như thế nào?

3. Bài toán tìm ƯSCNN của hai số tự nhiên

Bài toán: yêu cầu nhập 2 số tự nhiên n, m từ bàn phím, chương trình sẽ thông báo kết quả là ước số chung lớn nhất của hai số n, m trên màn hình.

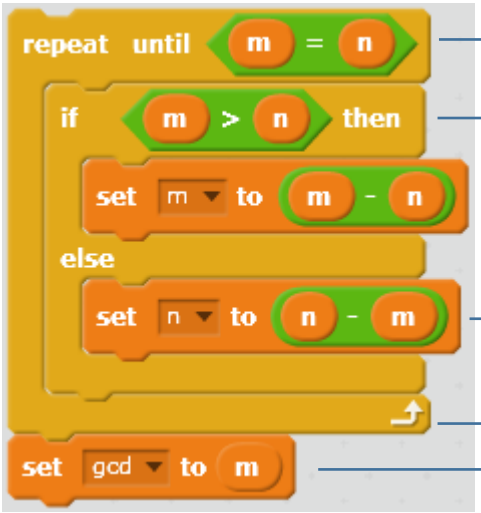
Chúng ta sử dụng biến nhớ **gcd** (greatest common divisor) để lưu trữ kết quả phép tính. Thuật toán đơn giản nhất là dùng 1 biến nhớ tạm, ví dụ **num**, cho **num** chạy từ 1 đến n và kiểm tra xem num có là ước số của đồng thời n, m hay không, nếu đúng thì gán giá trị này vào **gcd**.

Cách thứ 2 hay hơn xuất phát từ nhận xét: nếu n=m thì rõ ràng gcd = n. Giả sử n>m, khi đó gcd, là ước của n, m nên cũng sẽ là ước của n-m, bằng cách thay thế n bằng n-m chúng ta sẽ tìm tiếp với cặp n-m, m, cách này đi nhanh hơn đến kết quả cuối cùng. Thuật toán sau dựa trên ý tưởng trên và được coi là thuật toán chuẩn để tính ƯSCLN của 2 số tự nhiên n, m.

Thuật toán tính ước số chung lớn nhất.

```
Lặp cho đến khi m = n
     Nếu m > n thì
          m = m - n
     còn không thì
          n = n - m
Gán gcd = m.
```

Đoạn chương trình thể hiện thuật toán trên như sau:



Vòng lặp chính lặp cho đến khi m=n

Lệnh kiểm tra chính của vòng lặp: giảm số lớn hơn trong 2 số đi 1 giá trị bằng hiệu của 2 số này.

Kết quả gán cho biến **gcd**

Em hãy hoàn thiện chương trình này trên Scratch.

4. Bài toán kiểm tra số nguyên tố

Bài toán: yêu cầu nhập 1 số tự nhiên n từ bàn phím. Chương trình sẽ thông báo số này là nguyên tố hay hợp số.

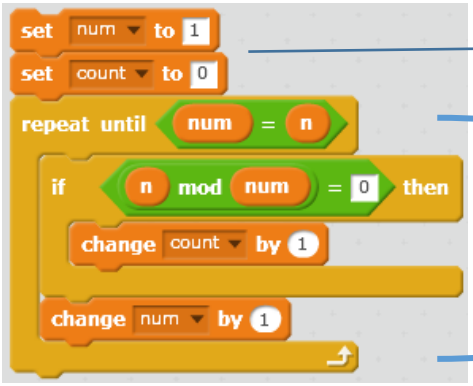
Ý tưởng của lời giải này là đếm số các ước số thực sự của n. Nếu số ước số thực sự < 2 thì n sẽ là nguyên tố.

Thuật toán sau sau đếm số các ước số thực sự của **n**. Sử dụng biến **count** để đếm số các ước số thực sự của **n**. Biến nhớ **num** được tạo ra dùng để chạy theo 1 vòng lặp từ 1 đến n-1 và kiểm tra xem **num** có phải là ước của **n** hay không. Nếu **num** là ước của **n** thì tăng **count** lên 1. Biến **count** được gán giá trị ban đầu = 0.

Thuật toán đếm số các ước số thực sự của một số tự nhiên **n** tính cả 1.

```
Gán num = 1, count = 0
Lặp cho đến khi num = n
     Nếu num là ước của n thì
          Tăng count lên 1 đơn vị.
     Tăng num lên 1 đơn vị.
```

Đoạn chương trình trong Scratch mô tả thuật toán trên như sau:



Gán các giá trị ban đầu.

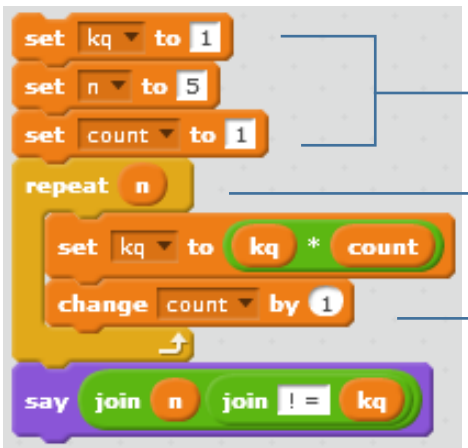
Vòng lặp chính. Lặp cho đến khi num = n thì dừng lại. Khi vòng lặp kết thúc, biến **count** chính là số cần tính.

Chương trình hoàn chỉnh cho phép nhập từ bàn phím 1 số và thông báo kết quả trên màn hình.

5. Bài toán tính nhanh 100!

Chúng ta đã biết cách tính **n giai thừa** n! = 1.2.3…. n (tích của n số tự nhiên đầu tiên, tính từ 1). Với n lớn thì việc tính n! rất khó. Máy tính sẽ giúp em tính rất nhanh bài toán này.

Đoạn chương trình sau mô tả việc tính n! Biến nhớ **count** sẽ tăng dần từ 1 đến n để góp phần tính tăng cho kết quả **kq**. Vòng lặp ngoài có n vòng. Các biến **count** và **kq** được gán giá trị ban đầu = 1.



Các thiết lập ban đầu.

Vòng lặp chính (n vòng)

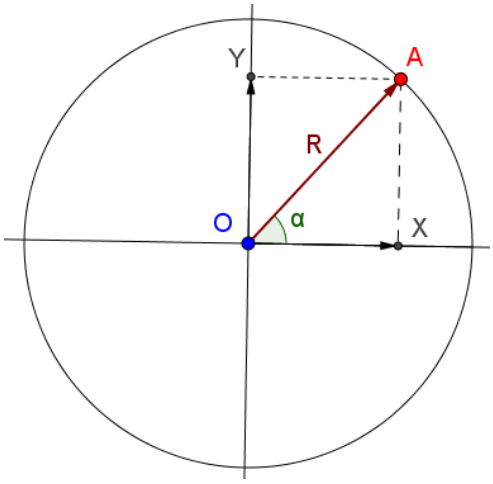
Nhóm lệnh chính trong vòng lặp: tăng kq theo phép nhân với count và và tăng count lên 1.

Hiển thị kết quả

Viết lại chương trình tính n! hoàn chỉnh, giá trị n được nhập từ bàn phím.

6. Bài toán vẽ đường tròn (mới)

Chúng ta quay lại bài toán vẽ đường tròn trong bài **Đồ họa 2** nhưng phân tích trên cơ sở toán học cao hơn.



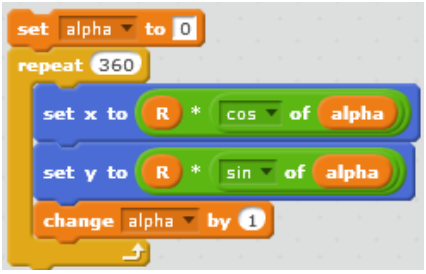
Trong hình bên, điểm A sẽ có tọa độ (x, y) được tính theo công thức sau:

x = R.cos(α).

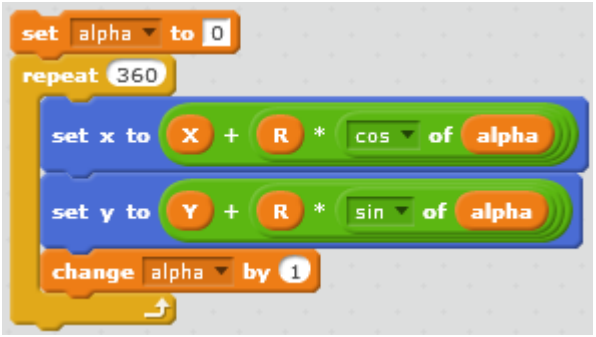
y = R.sin(α).

Trong ngôn ngữ lệnh Scratch, nếu thiết lập các biến nhớ X, Y, R, alpha thì 2 công thức trên viết như sau:

Đoạn chương trình cho điểm A chuyển động trên vòng tròn tâm (0,0) bán kính R có thể như sau:



Còn đây là chương trình cho nhân vật hiện thời chuyển động xung quanh điểm có tọa độ (X, Y), bán kính R.



Em hãy thiết lập chương trình điều khiển con Mèo luôn chạy vòng quanh con Chuột với bán kính 150, trong khi chạy Mèo luôn quay mặt về hướng Chuột.

Câu hỏi và bài tập

1. Viết chương trình nhập số m từ bàn phím và đếm số các ước số của m, tính cả 1 và n. Ví dụ nếu m = 10 thì thông báo đáp số là 4.

2. Viết đoạn chương trình với dữ liệu đầu vào là n, m và kiểm tra xem n, m có nguyên tố cùng nhau hay không? Hai số được gọi là nguyên tố cùng nhau nếu ƯSCLN của chúng = 1.

3. Viết đoạn chương trình nhập 3 số dương a, b, c và kiểm tra xem có thể vẽ được tam giác ABC với các cạnh có số đo a, b, c hay không.

4. Viết chương trình yêu cầu nhập 2 số tự nhiên từ bàn phím, sau đó hiện thông báo tính ƯSCLN và BSCNN của 2 số này.

Gợi ý: BSCLN của m, n được tính theo công thức: BSCNN = m.n /ƯSCLN.

5. Viết chương trình hiện trên màn hình tất cả các số nguyên tố < 100.

6. Viết chương trình thực hiện yêu cầu sau:

Thầy giáo yêu cầu học sinh nhập 1 số nguyên tố từ bàn phím. Nếu người dùng nhập không phải số nguyên tố thì thông báo "bạn nhập sai, không là số nguyên tố" và yêu cầu nhập lại. Nếu nhập đúng thì thông báo "Đúng rồi, bạn hoàn thành công việc" và dừng chương trình.

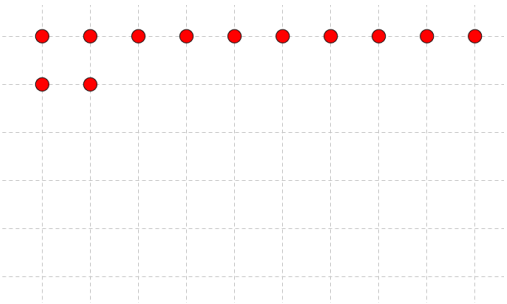
7. Số tự nhiên M được gọi là số hoàn hảo nếu M = tổng tất cả các ước số thực sự của mình kể cả 1, ví dụ 6 = 1 + 2 + 3 là số hoàn hảo.

Hãy viết chương trình hiện ra tất cả các số hoàn hảo < 10000.

8. Cho trước 3 số tự nhiên A, B, C. Viết đoạn chương trình tính và đưa ra tất cả các ước số chung của cả 3 số này.

9. Viết chương trình nhập số tự nhiên N và sau đó hiện ra tất cả các số nguyên tố nhỏ hơn hoặc bằng N.

10. Viết chương trình nhập 1 số tự nhiên bất kỳ trong phạm vi 1 đến 100, sau đó vẽ đúng số lượng hình tròn nhỏ trên lưới tương tự như trong hình sau.



Gợi ý: sử dụng lệnh **stamp** để vẽ các hình tròn này.

11. Viết chương trình yêu cầu người dùng nhập từ bàn phím 1 số tự nhiên N bất kỳ, sau đó thông báo số nguyên tố P đầu tiên lớn hơn hoặc bằng N.

Ví dụ nếu nhập N = 50 thì P = 53.

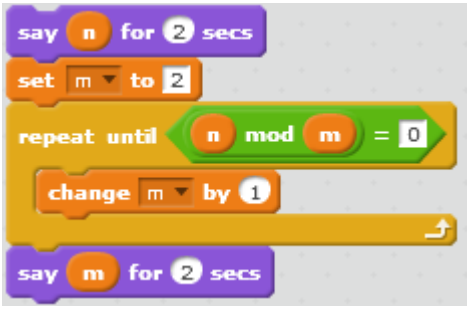
12. Viết chương trình yêu cầu người dùng nhập từ bàn phím 1 số tự nhiên N bất kỳ, sau đó thông báo số nguyên tố P lớn nhất không lớn hơn N.

Ví dụ nếu nhập N = 24 thì P = 23.

13. Viết chương trình nhập 1 số tự nhiên N từ bàn phím, sau đó chương trình sẽ liệt kê tất cả các ước số nguyên tố (khác nhau từng đôi một) của N, sắp xếp theo thứ tự tăng dần.

Ví dụ nếu nhập N = 24 thì chương trình in ra: 2 3.

14. Đoạn chương trình sau thực hiện các công việc gì và sẽ đưa ra kết quả là gì? Giải thích kết quả.



15. Dãy số Fibonaci F_n, n = 1, 2, 3, … được định nghĩa như sau:

$F_1 = 1$, $F_2 = 1$, $F_3 = 2$, $F_4 = 3$, …., $F_n = F_{n-1} + F_{n-2}$.

Viết chương trình yêu cầu người dùng nhập số n (n < 100) và in ra số Fibonaci thứ n.

16. Cặp số nguyên tố (P, Q) được gọi là cặp số nguyên tố sinh đôi nếu Q = P + 2. Ví dụ (3, 5). Viết chương trình liệt kê tất cả các cặp số nguyên tố sinh đôi < 1000.

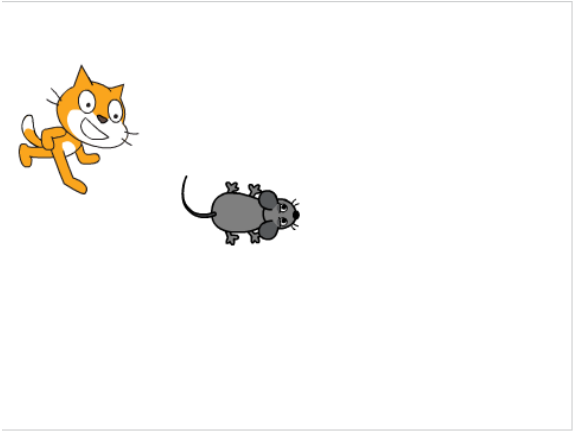
17. Cặp số thân thiết.

Hai số tự nhiên (m, n) được gọi là thân thiết nếu số này = tổng số các ước số thực sự của số còn lại (kể cả 1). Ví dụ một cặp số như vậy: (220, 284)

Viết 1 chương trình Scratch nhỏ để tìm ra tất cả các cặp số thân thiết trong phạm vi 1000, tức là tìm tất cả các cặp số thân thiết n, m với điều kiện n, m < 1000.

18. Giải bài tập đã cho ở cuối phần 6.

Em hãy thiết lập chương trình điều khiển con Mèo luôn di chuyển vòng quanh con Chuột với bán kính 150, trong khi chạy Mèo luôn quay mặt về hướng Chuột.



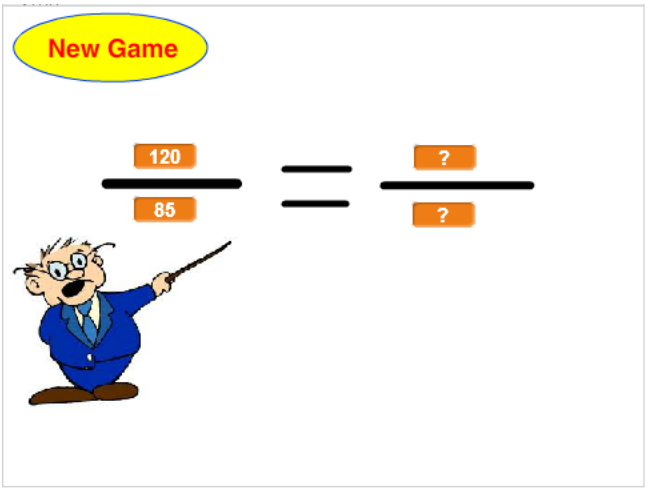
Mở rộng

1. Viết chương trình cho phép nhập số tự nhiên n từ bàn phím và in ra khai triển số n thành tích của các thừa số nguyên tố.

Ví dụ đầu vào là 12 thì thông báo ra là: 12 = 2.2.3.

2. Thiết kế trò chơi **Rút gọn phân số** sau.

Game Rut gon phan so.sb2

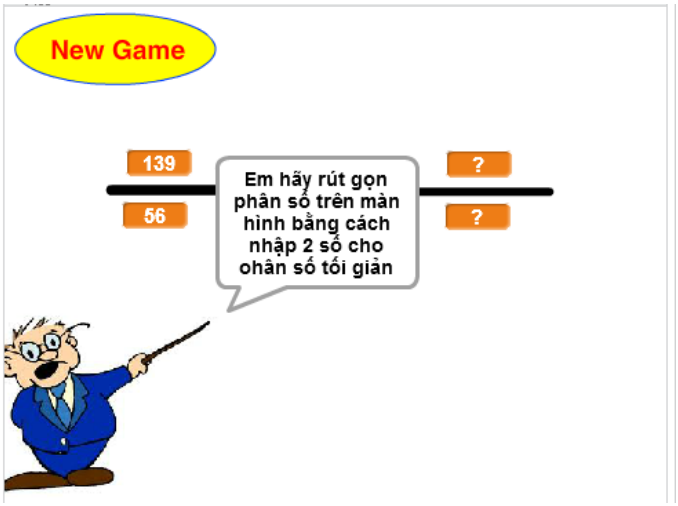


Mỗi lần nháy lên nút **New Game** sẽ bắt đầu trò chơi.

Thầy giáo sẽ đưa ra ngẫu nhiên 1 phân số trên màn hình và yêu cầu người chơi rút gọn phân số này và nhập 2 số p/q là phân số tối giản của phân số trên màn hình.

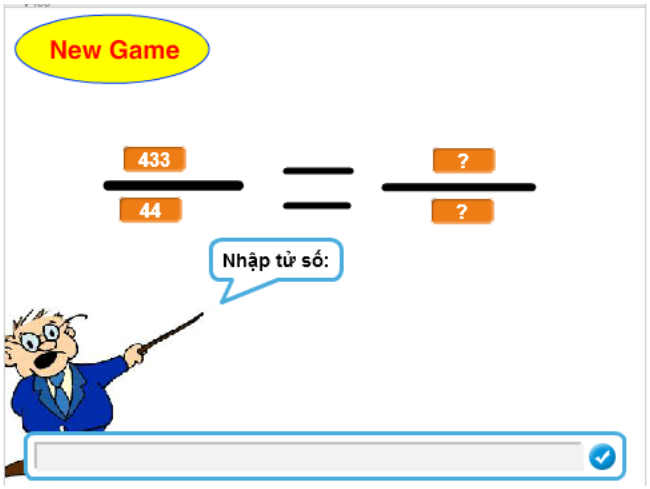
Sau khi nhập thầy giáo sẽ thông báo ngay là đúng hay sai, nếu sai thì cần nhập lại, nếu đúng thì kết thúc.

Ví dụ mỗi lần bắt đầu chơi, chương trình sẽ tự sinh 1 phân số gốc và yêu cần rút gọn phân số, tìm phân số tối giản. Giao diện sẽ như sau:



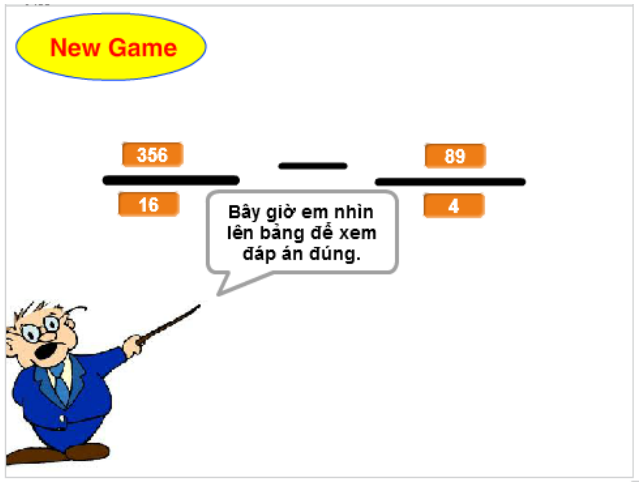
Sau khi hiện màn hình trên vài giây, giáo viên sẽ yêu cầu nhập tử số và mẫu số của phân số tối giản.

Giáo viên yêu cầu nhập tử số của phân số tối giản cần tìm.



Sau khi nhập xong tử số và mẫu số, giáo viên sẽ thôn báo ngay đúng hay sai.

Thông báo cuối cùng như sau trước khi tự động chuyển sang lượt chơi mới.

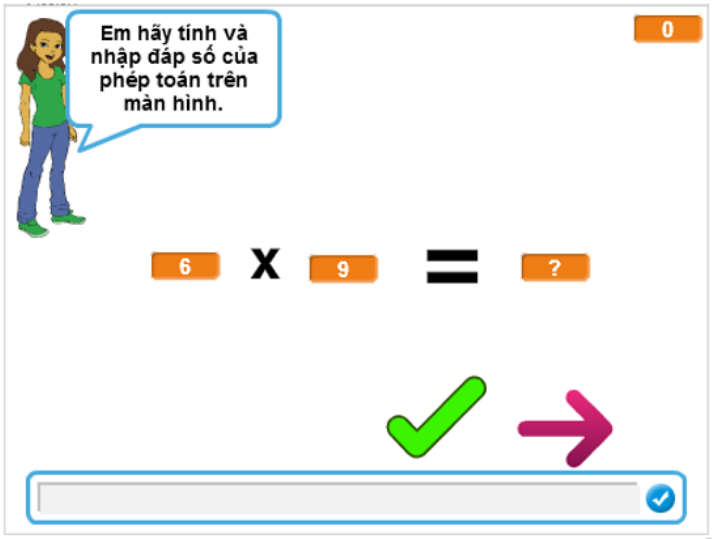


3. Thiết kế chương trình **Học toán** như sau:

Hoc toan.sb2

Giao diện và nhân vật được thiết kế như hình dưới đây:

- Nhân vật: Giáo viên, phép toán (+, -, x, :), dấu =, nút Check và nút Next.
- Phép tính được thể hiện ngay trên màn hình.



Các nhân vật của chương trình như sau:

Phần mềm hoạt động như sau:

- Phần mềm sẽ tự động sinh các phép tính +, -, x, : (ví dụ trong phạm vi 1000) ngay trên màn hình, GV thông báo và yêu cầu người chơi nhập đáp số. Khi người chơi nhập đáp số, số này sẽ hiện trên màn hình ở bên phải phép toán.

- Khi nhập xong, nút **Check** xuất hiện. Nháy vào nút này để kiểm tra Đúng hay Sai. GV sẽ thông báo đúng hay sai. Nếu đúng được + 5 điểm, nếu sai bị trừ -2 điểm. Nếu sai, kết quả sẽ được cập nhật lại cho đúng.

- Sau khi thông báo đúng, sai, nút **Next** xuất hiện. Nháy nút này sẽ tự động chuyển sang bài tập tiếp theo.

Bài 14. Xử lí xâu ký tự 1

Mục đích

- Giá trị không là số hoặc logic.
- Các phép tính, tính toán đơn giản với giá trị là chữ hoặc văn bản.
- Thực hiện 1 vài bài toán đơn giản xử lý chữ, ký tự, văn bản.

Bắt đầu

1. Trong các biểu thức giá trị sau, hãy đánh dấu các giá trị là số hoặc logic:

a. Heal The World

b. Min < Max

c. 321 + 231 + 123

d. Sông Mê Kông dài hơn sông Hồng

e. Nếu hôm nay trời mưa em sẽ học ở nhà

f. (x < 10) and (x > 1)

2. Em hiểu thế nào là 1 xâu ký tự? Các nội dung sau có phải là 1 dãy ký tự không?

Hanoi1 + Hanoi2

123456789

Hà Nội là thủ đô của nước Việt Nam

$x^2 + y^2 + z^2 = 192$

Giải phương trình $ax^2 + bx + c = 0$

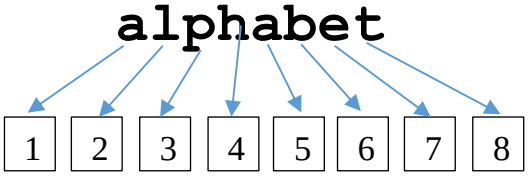
Nội dung bài học

1. Cách xâu ký tự được lưu trữ trong máy tính

Xâu ký tự được hiểu là 1 dãy các ký tự, ví dụ "Hà Nội", "English", "letters", "hòa bình". Dãy các ký tự tạo nên 1 xâu sẽ được đánh số từ 1.

Ví dụ từ alphabet.

Phân tích 1 xâu ký tự.



Chú ý:

- Dãy ký tự của xâu có thể chứa các ký tự đặc biệt như : ; / \ { } [ ] ( ) …..
- Dấu cách (Space) cũng được coi là 1 ký tự, ta dùng ký hiệu ␣ để chỉ dấu cách.

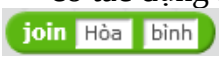
Tổng số các ký tự của 1 xâu được gọi là **độ dài** của xâu ký tự này.

2. Các hàm xử lý xâu ký tự trong Scratch

Trong môi trường Scratch có 3 hàm xử lý xâu ký tự sau:

Các hàm xử lý xâu ký tự.

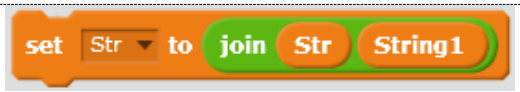
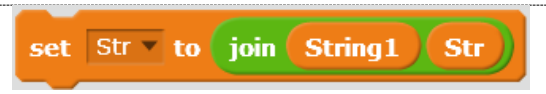
	(hàm) toán tử nối 2 xâu ký tự.
	(hàm) trả lại ký tự thứ <1> của xâu ký tự <world>
	(hàm) trả lại độ dài của xâu ký tự <world>

Toán từ  có tác dụng nối 2 xâu ký tự và trả lại kết quả là xâu được kết nối. Ví dụ lệnh  sẽ trả lại xâu ký tự "Hòa bình" là nối của 2 từ "Hòa " và "bình". Lệnh **join** có thể lồng nhau nhiều mức.

Hàm  sẽ trả lại 1 ký tự cụ thể trong 1 xâu, từ. Ví dụ lệnh  sẽ trả lại giá trị là chữ "t".

Hàm  trả lại độ dài của 1 xâu, từ cho trước. Ví dụ  trả lại số 7.

Sau đây là 1 số thao tác, lệnh đơn giản khác liên quan đến xâu ký tự.

Gán xâu ký tự **Str** là rỗng	
Gán giá trị của xâu **String1** cho xâu **Str**.	
Lấy ra 1 ký tự ngẫu nhiên của xâu **Str**.	
Bổ sung xâu **String1** vào cuối của xâu **Str**.	
Bổ sung xâu **String1** vào đầu của xâu **Str**.	

3. Spelling English Word. Bài toán học phát âm tiếng Anh

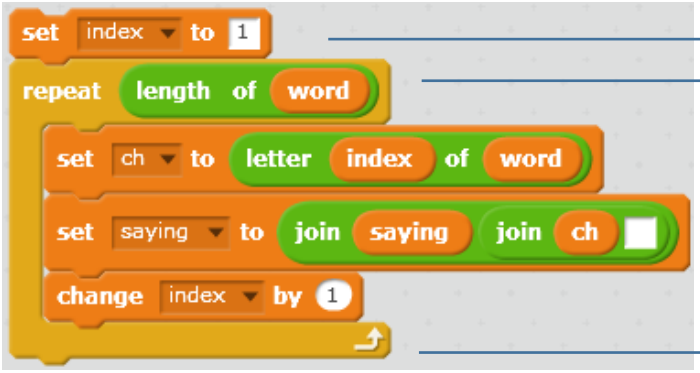
Em bắt đầu bài học bằng 1 ví dụ đơn giản sau: Bài học phát âm tiếng Anh (spelling word).

Thầy giáo yêu cầu học sinh nhập 1 từ tiếng Anh, sau đó thầy sẽ diễn giải cách phát âm từng chữ của từ tiếng Anh này.

Ví dụ: nếu học sinh nhập từ **peace** thì giáo viên sẽ đưa ra cách phát âm **"p e a c e"**.

Em tạo biến nhớ **word** để lưu trữ từ do người dùng nhập, sau đó phần mềm sẽ tách từng chữ của từ này và đưa lên màn hình. Sử dụng biến chạy **index** để đưa từng chữ của từ này ra màn hình. Biến nhớ **saying** để lưu kết quả cần đưa ra màn hình.

Đoạn chương trình chính như sau.



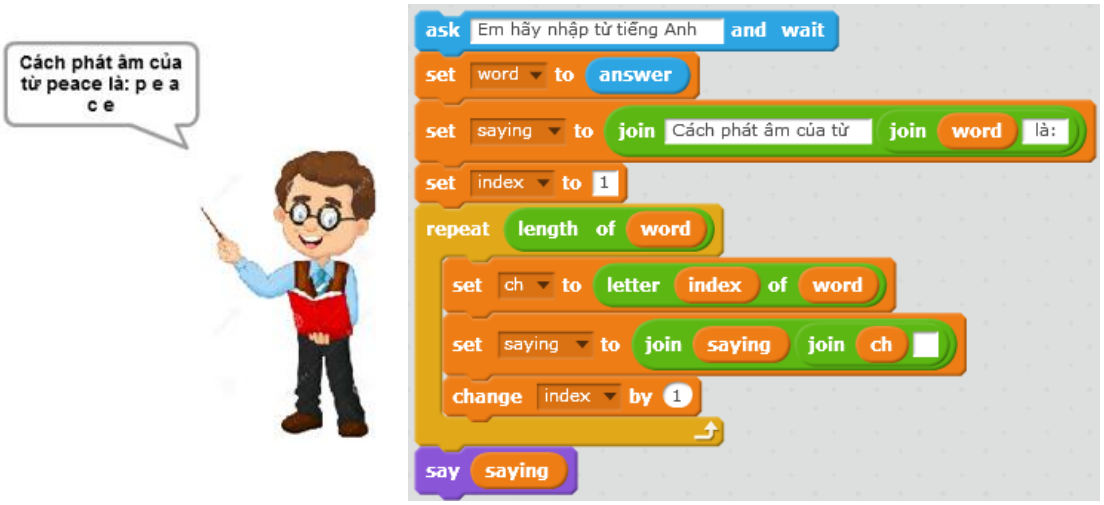
Gán giá trị ban đầu cho biến

Vòng lặp chính.

Với mỗi lần lặp, lấy ra **ch** = 1 ký tự tương ứng của xâu **word**, rồi đưa vào cuối của biến **saying**, và bổ xung thêm 1 khoảng trống.

Chương trình hoàn chỉnh của bài học này, kết quả hiện trong hình bên phải.

Spelling.sb2



4. Kiểm tra tính chất của từ, xâu ký tự

4.1. Kiểm tra 1 xâu ký tự có phải là nhị phân hay không.

Xâu nhị phân là xâu chỉ bao gồm các ký tự 0 hoặc 1.

Bài toán: cho trước 1 xâu ký tự **Str**, cần kiểm tra xem **Str** có phải là xâu nhị phân hay không.

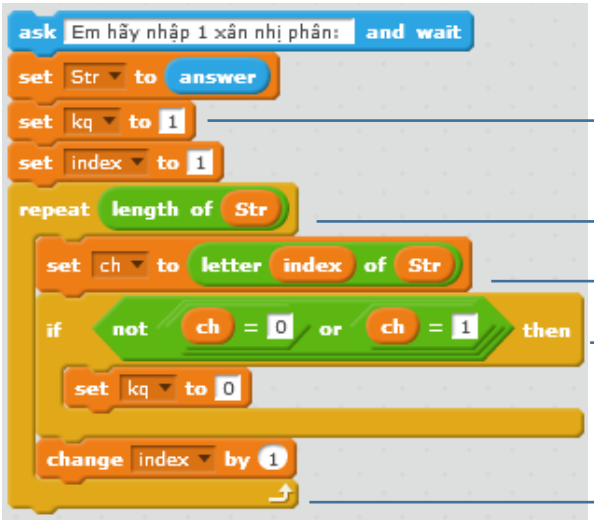
Để kiểm tra 1 xâu ký tự **Str** có phải là nhị phân hay không, em hãy thực hiện theo cách sau. Biến nhớ **kq** dùng để lưu kết quả kiểm tra, xâu là nhị phân nếu kq = 1, ngược lại nếu kq = 0 thì xâu không là nhị phân. Để kiểm tra ký tự **ch** là 0 hoặc 1 hay không chúng ta dùng biểu thức 

.

Bắt đầu chương trình, em gán kq = 0, trong quá trình kiểm tra trong vòng lặp nếu gặp 1 ký tự không là 0 hoặc 1 thì gán ngay kq = 1.

Phần lõi của chương trình như sau:

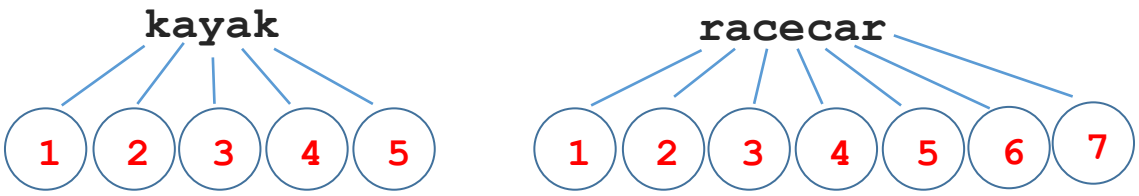
Kiem tra nhi phan.sb2



Em hãy viết hoàn thiện chương trình này.

4.2. Kiểm tra 1 xâu ký tự có phải là đối xứng (palindrome) hay không.

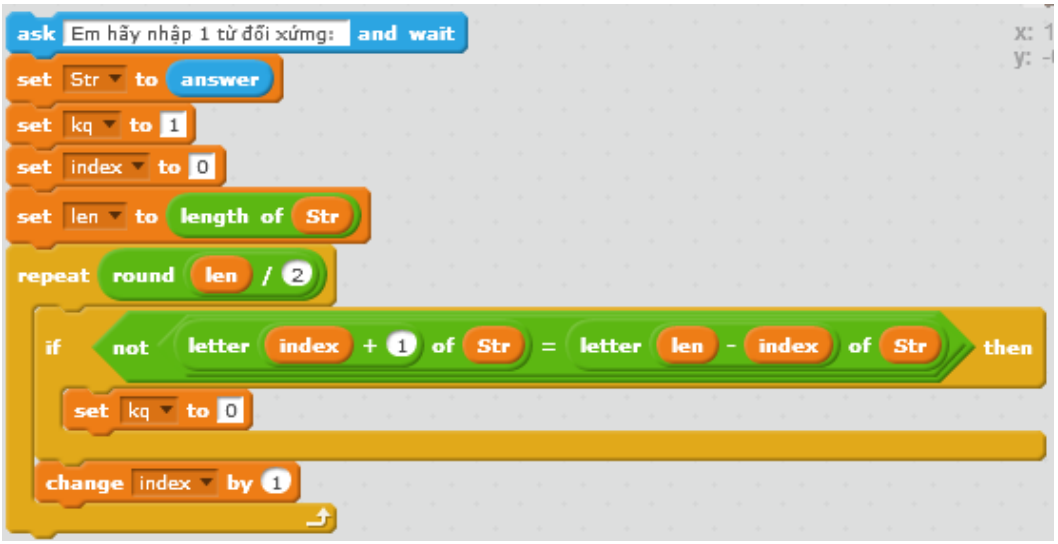
Một xâu được gọi là đối xứng (palindrome) nếu các ký tự, chữ tạo thành xâu này đối xứng qua trục thẳng đứng. Hay nói cách khác các từ nếu đọc xuôi hay ngược đều như nhau được gọi là palindrome. Ví dụ:
redivider, noon, civic, radar, level, rotor, kayak, reviver, racecar, madam, refer.



Phân tích: Nếu độ dài của xâu là **len** thì xâu đã cho sẽ là đối xứng nếu các cặp chữ với chỉ số (1, len), (2, len-1), … phải có giá trị bằng nhau. Do vậy chỉ cần kiểm tra theo 1 vòng lặp chỉ số từ 1 đến **len/2** là đủ. Xâu gốc ký hiệu là **Str**.

Chương trình

Kiem tra palindrome.sb2

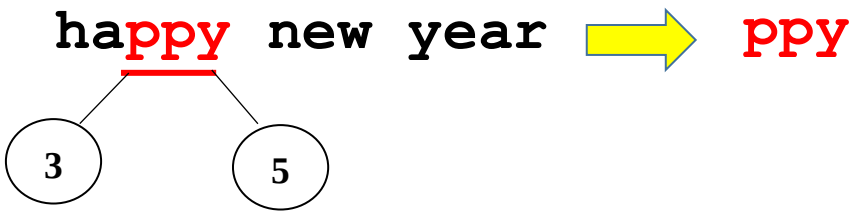


Em hãy viết hoàn thiện chương trình này.

5. Hàm lấy xâu con của 1 xâu hoặc từ

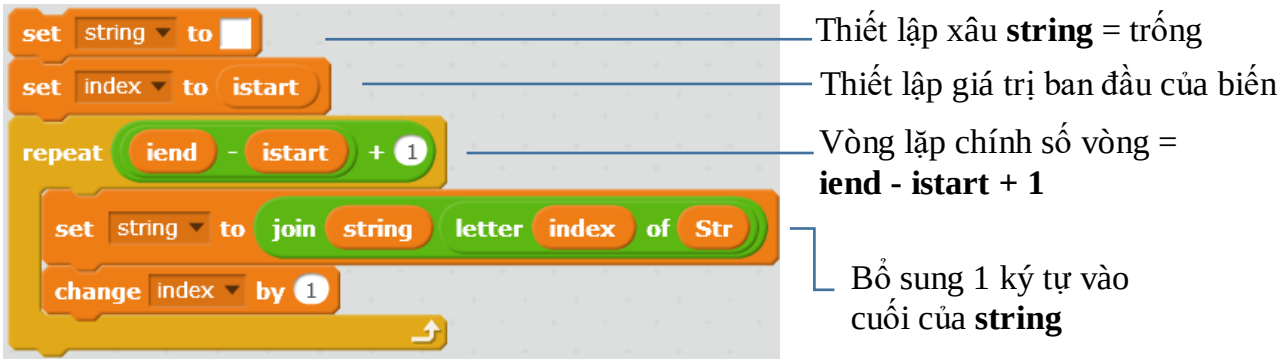
Bài toán: cho trước 1 xâu ký tự **Str** với độ dài **len**. Cần viết 1 chương trình để lấy ra 1 phần của xâu này (xâu con), tính từ vị tri **istart** đến vị trí **iend** của xâu này.

Ví dụ với xâu "happy new year", **istart** = 3, **iend** = 5, độ dài xâu con = **iend - istart + 1** = 3. Xâu con được trả lại được lưu trong biến **string** là "ppy"



Thuật toán của bài toán này khá đơn giản. Thiết lập 1 vòng lặp với số bước lặp **iend - istart + 1**, biến index bắt đầu từ vị trí **istart**, mỗi bước lấy 1 chữ từ xâu gốc và đưa vào cuối của xâu kết quả **string**.

Đoạn chương trình tính xâu con như sau.



Em hãy hoàn thiện chương trình này bổ sung phần yêu cầu nhập xâu ký tự gốc **Str**, nhập 2 chỉ số bắt đầu và kết thúc xâu con **istart** và **iend**.

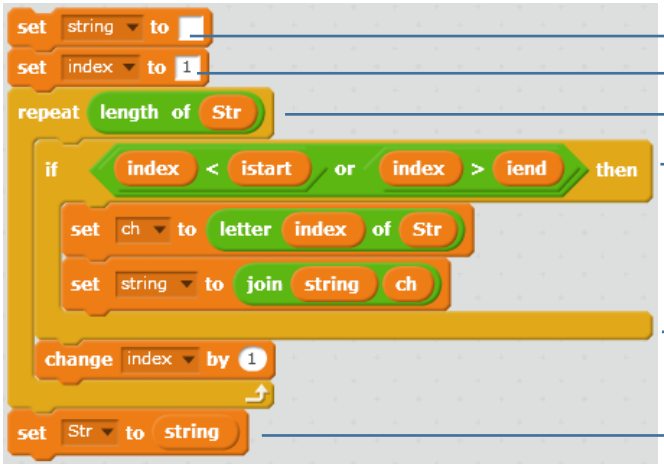
6. Hàm xóa 1 ký tự (1 phần) của xâu

Bài toán: cho trước 1 xâu ký tự **Str** với độ dài **len**. Cần viết chương trình xóa đi xâu con của xâu này, tính từ vị tri **istart** đến vị trí **iend**.

Ví dụ với xâu "happyness new year", **istart** = 6, **iend** = 9, xâu mẹ sau khi đã xóa xâu con có giá trị là "happy new year"

Thuật toán của bài toán này như sau. Tạo biến nhớ phụ **string** dùng để lưu tạm kết quả. Vòng lặp chính chạy suốt chiều dài của xâu gốc **Str**. Với mỗi ký tự của Str, chương trình kiểm tra nếu vị trí này < **istart** hoặc > **iend** thì đưa ký tự nào vào cuối của xâu **string** (ngược lại không làm gì cả).

Đoạn chương trình xóa xâu con như sau.



Thiết lập xâu **string** = trống

Thiết lập giá trị ban đầu của biến

Vòng lặp chính

Kiểm tra nếu vị trí ký tự hiện thời có chỉ số < **istart** hoặc > **iend** thì đưa vào cuối của xâu

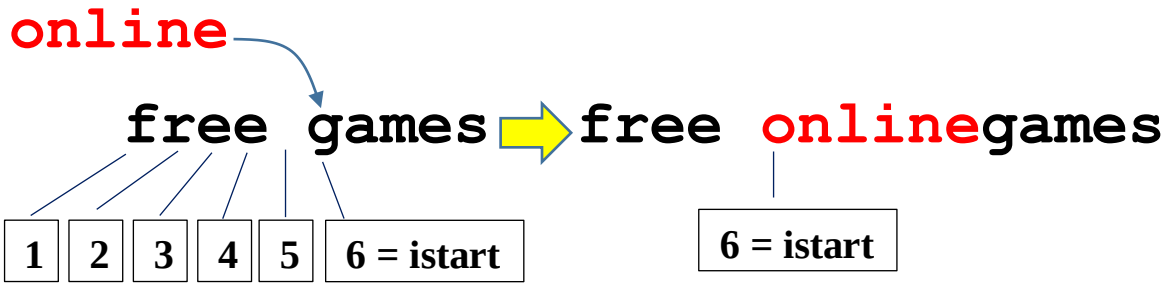
Gán **string** lại cho **Str**.

Em hãy hoàn thiện chương trình này bổ sung phần yêu cầu nhập xâu ký tự gốc **Str**, nhập 2 chỉ số bắt đầu và kết thúc xâu con cần xóa **istart** và **iend**.

7. Hàm chèn xâu (ký tự) vào xâu khác tại vị trí cho trước

Bài toán: cho 2 xâu ký tự: xâu gốc **Str** và xâu thứ 2 **Str1**. Cần chèn xâu **Str1** vào xâu gốc **Str** tại vị trí **istart**, kết quả ghi lại trong chính xâu gốc **Str**.

Để chuẩn bị chương trình chúng ta cùng tìm hiểu qua 1 ví dụ cụ thể. Xâu gốc **Str** = "free games", xâu cần chèn **Str1** = "online", vị trí chèn **istart** = 6.

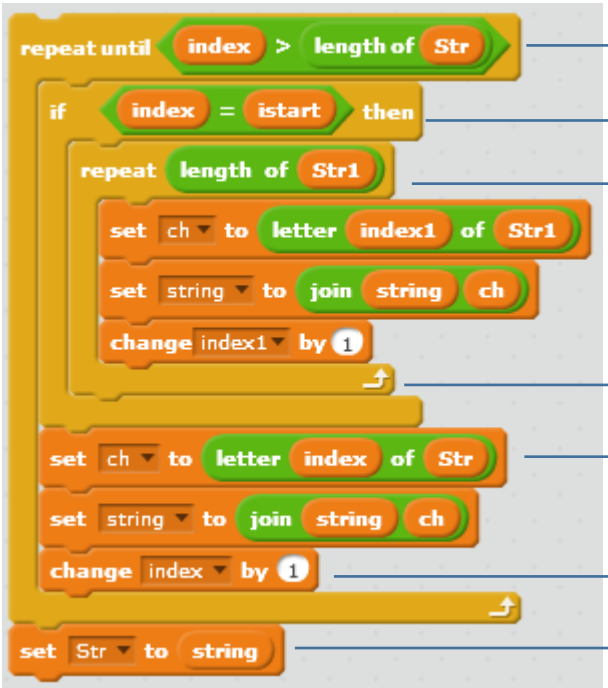


Đầu tiên chúng ta cho biến nhớ **index** chạy và gán từng chữ của xâu Str vào 1 biến trung gian **string**. Khi đến vị trí **istart**, quá trình này tạm dừng để chuyển sang việc bổ sung xâu **Str1** vào string. Công việc này dược thực hiện bằng 1 biến nhớ khác là **index1**. Khi đã bổ sung xong **Str1** thì quá trình bổ sung Str lại tiếp tục bằng **index**. Chương trình này sẽ có 2 vòng lặp lồng nhau.

Thuật toán trên có thể viết lại như sau:

```
Gán các giá trị ban đầu index =1, index1 = 1
Thực hiện lặp cho đến khi index > độ dài Str
     Nếu index = istart thì
          Thực hiện vòng lặp có số bước bằng độ dài
xâu Str1
               Bổ sung Str1 vào cuối string
     Gán ký tự thứ index của xâu Str vào string
Gán Str = string
```

Đoạn chương trình sau mô tả phần thuật toán chính của bài toán.



Vòng lặp ngoài, điều kiện dừng khi duyệt xong xâu Str

Kiểm tra điều kiện để thực hiện vòng lặp trong

Vòng lặp trong, bổ sung xâu Str1 vào cuối của string. Vòng lặp này chỉ thực hiện đúng 1 lần.

Các lệnh vòng lặp ngoài: bổ sung từng chữ của xâu Str vào cuối của string.

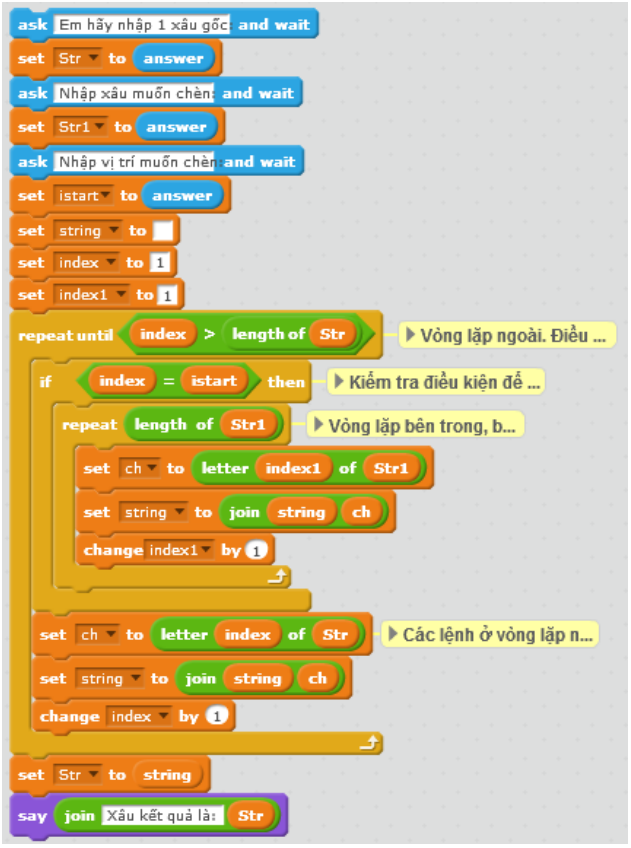
Gán trả lại string vào Str là xâu gốc ban đầu.

Chương trình hoàn chỉnh sẽ yêu cầu học sinh nhập lần lượt các thông số:

- Xâu gốc Str
- Xâu cần chèn Str1
- Vị trí cần chèn

Chương trình thực hiện công việc chèn và thông báo kết quả ra màn hình.

Chen xau con vao xau me.sb2



Câu hỏi và bài tập

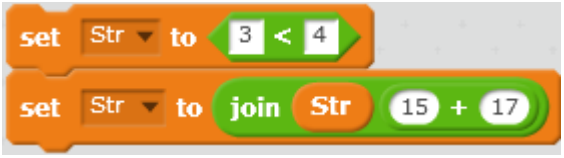
1. Các biểu thức nào dưới đây viết đúng?

$(m^2 + n^2)$ và $(m^2 - n^2)$

$(a + b)^2 = a^2 + 2ab + b^2$

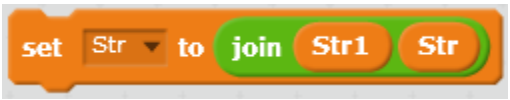
abc < bce

2. Sau 2 lệnh sau thì biến nhớ Str sẽ lưu trữ xâu ký tự gì?



3. Nếu ý nghĩa của 2 lệnh sau, sự giống nhau, khác nhau.

và



4. Viết chương trình nhập 1 xâu ký tự bất kỳ, sau đó thông báo:

- Số lượng ký tự là chữ số trong xâu trên.

- Số lượng ký tự là chữ cái trong xâu trên.

5. Dùng lệnh của Scratch để kiểm tra các biểu thức sau đúng hay sai (tất cả các giá trị đều được nhập như xâu ký tự)

100 > a

abc < 123

abcd > abc

ABC < abc

abc1d > abc0c

Em có nhận xét gì sau khi kiểm tra tính đúng sai của các biểu thức trên.

6. Cho trước 2 xâu ký tự Str1, Str2. Viết 1 chương trình trộn 2 xâu này xen kẽ theo từng ký tự, kết quả đưa ra xâu Str. Ví dụ:

Str1 = abcdeghik

Str2 = 0123456

Khi đó Str = 10b1c2d3e4g5h6ik

7. Sắp xếp các xâu ký tự sau theo thứ tự từ điển từ bé đến lớn:

abc 123 1ab bca 1100a a1100 11mn mn1100

8. Viết chương trình mô tả 1 trò chơi đơn giản sau.

Trên màn hình xuất hiện 2 từ tiếng Anh, từ thứ nhất đầy đủ, từ thứ hai thu được từ từ thứ nhất sau khi lấy đi 1 xâu con.

Ví dụ 2 từ ban đầu: convert | cvert

Chương trình yêu cầu người chơi nhập 1 xâu ký tự chính là phần của từ thứ nhất sau khi lấy đi để thành từ thứ 2.

Trong ví dụ trên từ cần nhập là **on**.

9. Cho trước 2 xâu ký tự Str1 và Str2.

Viết chương trình hiện ra tất cả các ký tự có chung ở cả 2 xâu trên. Chú ý các ký tự không phân biệt chữ in hoa hay

Ví dụ: Str1 = Hà nội. Str2 = Hoà Bình.

Khi đó các ký tự chung của 2 từ trên sẽ là H, à, n.

10. Viết chương trình thầy giáo lần lượt hỏi nhập Tên, Họ, Đệm, sau đó thể hiện trên màn hình dòng chữ "Xin chào bạn <Họ Đệm Tên> " của học sinh.

Mở rộng

1. Thiết kế 1 trò chơi **Ghép chữ tạo từ chính xác** sau.

Phần mềm sẽ đưa ra trên màn hình 2 từ, người chơi cần ghép 2 từ này lại với nhau để tạo thành 1 từ đúng.

Ví dụ nếu 2 từ cho ban đầu là **ped** và **wikiia** thì từ cần ghép đúng phải là **wikipedia**. Người chơi được yêu cầu nhập từ cần ghép từ bàn phím cho đến khi tạo được từ đúng thì dừng lại. Nếu sau 10 lần vẫn nhập sai thì thua và phần mềm sẽ đưa ra đáp án đúng.

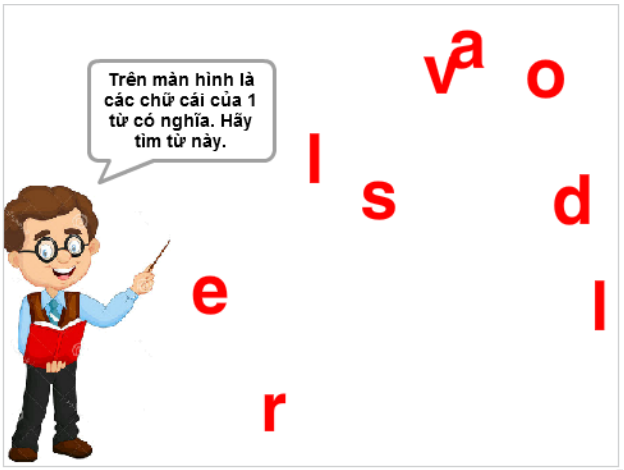
Game ghep chu tao tu.sb2

2. Thiết kế trò chơi **Xếp chữ** như sau.

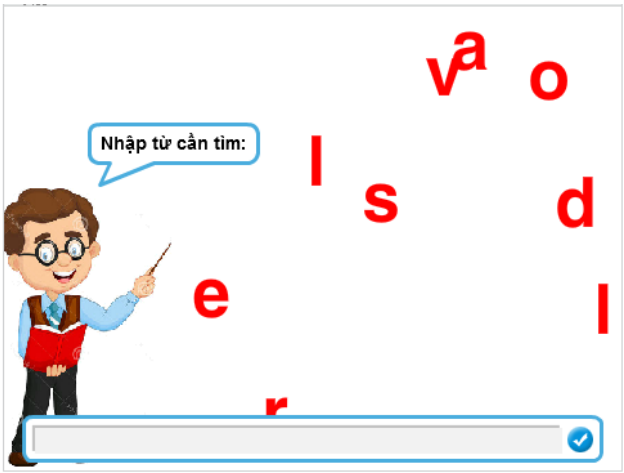
Phần mềm sẽ bắt đầu bằng hình ảnh một dãy chữ cái trên màn hình. Các chữ này có thể không cần thể hiện thẳng hàng. Nhiệm vụ của người chơi là xếp chúng lại thành 1 từ hoàn chỉnh.

Game xep chu.sb2

Chương trình sẽ hiển thị các chữ cái của 1 từ cần tìm một cách ngẫu nhiên trên màn hình. Thông báo ban đầu của chương trình như hình sau.



Sau đó chương trình sẽ yêu cầu ghép các chữ cái này lại thành 1 từ có ý nghĩa hoàn chỉnh. Quá trình này liên tục tiếp diễn cho đến khi người dùng nhập chính xác từ cần tìm.



Bài 15. Xử lí xâu ký tự 2

Mục đích

Học xong bài này bạn sẽ hiểu:

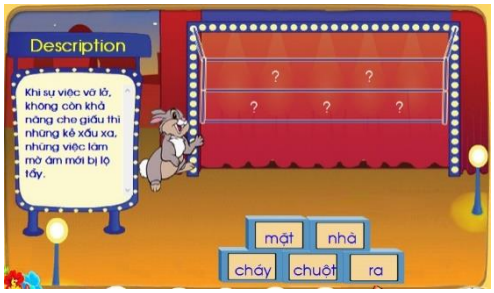
- Một số thuật toán đơn giản xử lý chữ và xâu ký tự.

- Thiết lập 1 vài trò chơi đơn giản với chữ và xâu ký tự.

Bắt đầu

Em hãy đọc để hiểu 1 số trò chơi liên quan đến chữ, xâu ký tự dưới đây. Em đã biết và đã từng chơi các trò chơi này chưa? Em hãy phát biểu suy nghĩ của mình xem có thể lập trình để mô phỏng chúng được hay không?

1. Trò chơi sắp xếp từ



Trên màn hình xuất hiện 1 dãy các từ. Nhiệm vụ của người chơi là cần sắp xếp lại các từ này theo 1 thứ tự nhất định có ý nghĩa nào đó, ví dụ, sắp xếp lại để trở thành 1 câu có nghĩa. Người chơi thao tác bằng cách kéo thả các chữ hoặc đánh số các từ trên màn hình.

2. Trò chơi đoán từ hangman

Trên màn hình hiện yêu cầu đoán 1 từ khi cho biết ý nghĩa của từ này. Người chơi đoán từ bằng cách nhập từng chữ cái cấu tạo nên từ này (bằng bàn phím hoặc chuột). Nếu nhập đúng thì chữ cái đó sẽ hiện trên màn hình tại đúng vị trí trong từ. Người chơi chỉ được phép nhập sai 1 số lần nhất định.

Nội dung bài học

1. Số nhị phân

Em có hiểu số nhị phân là gì không?

Trong các số sau, số nào là nhị phân?

```
345671    1011101   1001002   1010101
11011a1b  3220011   1111111   0000111
```

Hệ đếm thập phân, nhị phân.

Những số mà chúng ta làm việc hàng ngày đều là các số thập phân, hay còn gọi là số được viết trong hệ thập phân. Các số thập phân sử dụng 10 chữ số 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 để thể hiện. Nhưng trong máy tính chỉ có thể hiểu 2 chữ số là 0 và 1. Do vậy các số lưu trữ trong máy tính chỉ bao gồm các chữ số 0, 1. Các số này được gọi là số **nhị phân**. Hệ đếm chỉ dùng chữ số 0 và 1 gọi là **hệ nhị phân**. Hệ thập phân tiếng Anh gọi là **decimal**, hệ nhị phân gọi là **binary**.

Bảng sau cho ta biết tương ứng giữa 1 số số nhị phân và thập phân.

decimal	binary	decimal	binary	decimal	binary	decimal	binary
1	**1**	5	**101**	9	**1001**	13	**1101**
2	**10**	6	**110**	10	**1010**	14	**1110**
3	**11**	7	**111**	11	**1011**	15	**1111**
4	**100**	8	**1000**	12	**1100**	16	**10000**

2. Chuyển số nhị phân sang thập phân

Bài toán: cho 1 số nhị phân $\mathbf{a_n a_{n-1} \ldots a_1 a_0}$ (binary), cần chuyển đổi số này sang hệ thập phân.

Chúng ta quan sát qui luật biểu diễn số dưới dạng nhị phân và tìm ra qui luật tổng quát cho việc chuyển đổi số viết theo hệ nhị phân sang hệ thập phân.

100 → $100 = 1.2^2 + 0.2 + 0 = 4$ (hệ thập phân)

111 → $111 = 1.2^2 + 1.2 + 1 = 7$ (hệ thập phân)

Tổng quát chúng ta thấy:

$a_n a_{n-1} \ldots a_1 a_0$ (binary) = $a_n.2^{n-1} + a_{n-1}.2^{n-2} + \ldots + a_1.2 + a_0$ (decimal)

Nhìn vào công thức trên chúng ta chưa hình dung được thuật toán cần thực hiện trên máy tính. Chúng ta sẽ viết lại quá trình tính toán theo từng bước nhỏ để tính được số thập phân.

a_n

$2.a_n + a_{n-1}$

$2.(2.a_n + a_{n-1}) + a_{n-2}$

$2.(2.(2.a_n + a_{n-1}) + a_{n-2}) + an\text{-}3$

……

$2.(a_n.2^{n-3} + a_{n-1}.2^{n-4} + \ldots + a_0) + a_1$

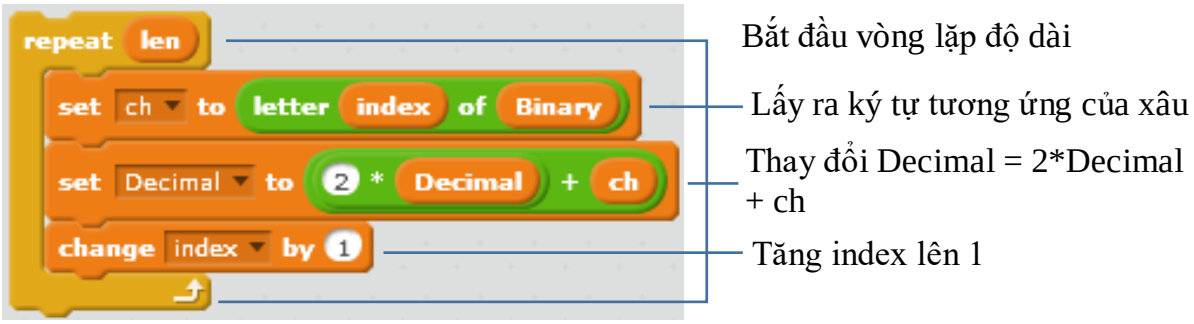
$2.(a_n.2^{n-2} + a_{n-1}.2^{n-3} + \ldots + a_1) + a_0$.

Như vậy sau đúng n+1 bước thì quá trình tính trên kết thúc và kết quả thu được số thập phân cần tìm.

Số nhị phân ban đầu được đưa vào biến nhớ **Binary**. Số thập phân là kết quả đuọc lưu trong biến nhớ **Decimal**. Các biến nhớ trung gian bao gồm **index, len** và **ch**.

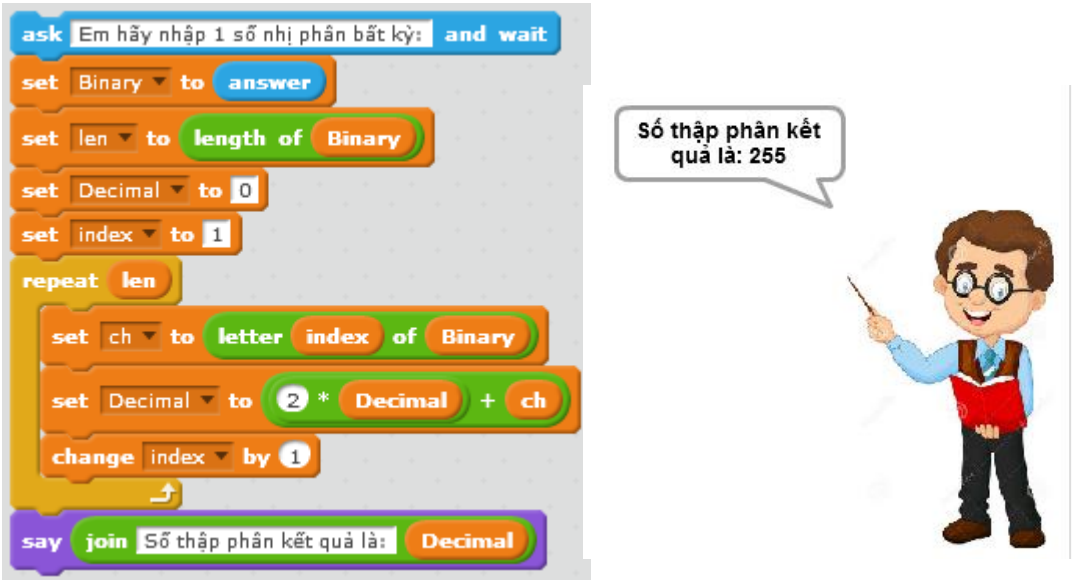
```
Gán ban đầu: Decimal = 0, index = 1
len = độ dài xâu Binary
Thực hiện vòng lặp với độ dài len
     ch = chữ số tương ứng chỉ số index của Binary
     Decimal = 2.Decimal + ch
     Tăng index lên 1
```

Đoạn chương trình mô tả thuật toán biến đổi hệ nhị phân sang thập phân Binary → Decimal như sau.



Chương trình hoàn chỉnh của bài toán này như sau:

Chuyen doi nhi phan sang thap phan.sb2



3. Chuyển số thập phân sang nhị phân

Bài toán: cho trước số thập phân, hãy biểu diễn số này sang hệ nhị phân.

Bài toán này giải quyết vấn đề ngược lại so với bài toán trên.

Để tìm được cách làm bài này, chúng ta lại quan sát công thức:

$a_n a_{n-1} \ldots a_1 a_0$ (binary) = $a_n.2^{n-1} + a_{n-1}.2^{n-2} + \ldots + a_1.2 + a_0$ (decimal)

Gọi số thập phân ban đầu là D, chúng ta sẽ phân tích quá trình tính toán để tính được lần lượt các số $a_0, a_1, \ldots, a_{n-1}, a_n$ như sau.

Số thập phân ban đầu: D

a_0 = D mod 2 (tính số dư); $a_n.2^{n-2} + a_{n-1}.2^{n-3} + \ldots + a_1$ = D/2 (làm tròn số); (gán D = D/2).

a_1 = D mod 2; D = D/2.

…….

a_{n-1} = D mod 2, D = D/2.

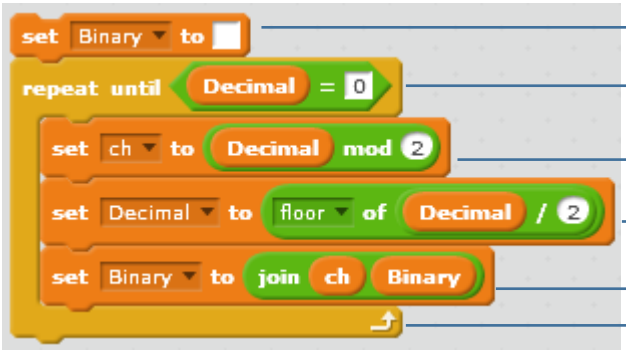
a_n = D mod 2, D/2 = 0, kết thúc.

Như vậy qui trình trên sẽ dừng lại khi D = 0.

Số thập phân đầu vào lưu trong biến nhớ **Decimal**. Xâu nhị phân đầu ra lưu trong biến nhớ **Binary**. Biến **ch** dùng để tính các giá trị trung gian a_i. Thuật toán chuyển đổi ở trên được viết tường minh như sau:

```
Gán Binary = rỗng
Thực hiện lặp cho đến khi Decimal = 0
     Đặt ch = Decimal mod 2;
     Decimal = Decimal / 2 (làm tròn xuống)
     Bổ sung ch vào bên trái của Binary
```

Đoạn chương trình mô tả thuật toán chuyển đổi số Nhị phân sang Thập phân trên được viết trong Scratch như sau:



Gán Binary = rỗng.

Vòng lặp chính với điều kiện dừng Decimal = 0

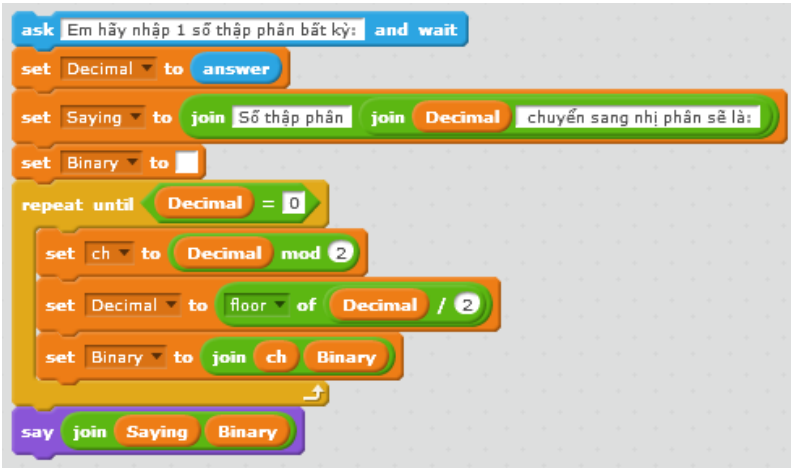
Tính ch = số dư Decimal chia 2

Giảm Decimal khi chia cho 2

Bổ sung **ch** vào **bên trái** của Binary

Chương trình hoàn chỉnh của bài toán này như dưới đây.

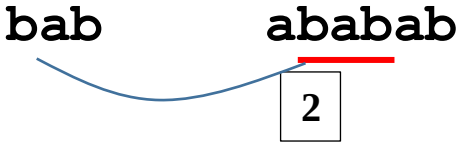
Chuyen doi thap phan sang nhi phan.sb2



4. Kiểm tra 1 ký tự / từ có nằm trong 1 từ khác hay không

Bài toán: Cho trước 2 xâu **Str1** và **Str**. Cần kiểm tra xem xâu **Str1** có nằm trong xâu Str không, nếu có thì cần chỉ ra vị trí ký tự đầu tiên của **Str1** trong **Str**. Nếu không nằm trong thì trả lại giá trị 0.

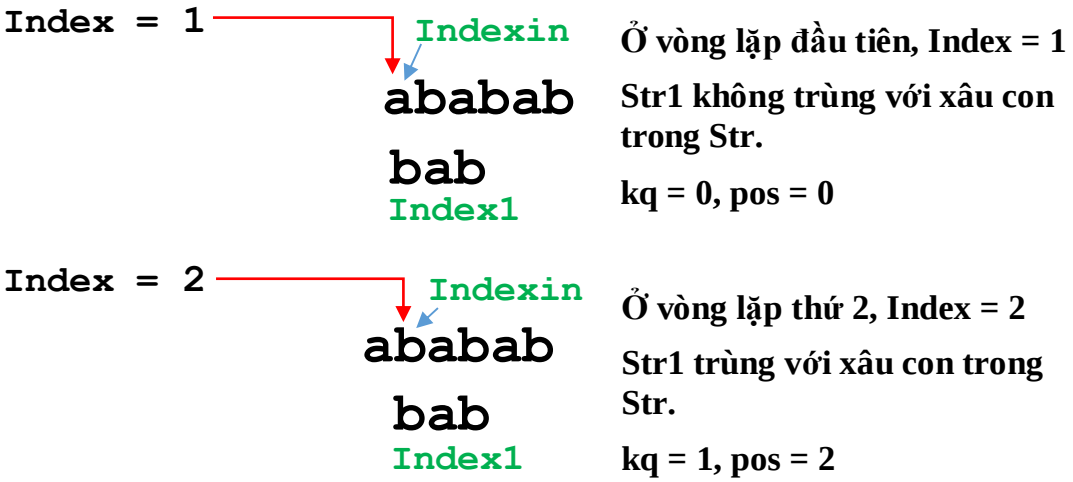
Ví dụ : 2 xâu là **bab** và **abababab**.



Xâu **bab** mặc dù được tìm thấy trong xâu **ababab** 2 lần nhưng hàm sẽ trả lại vị trí đầu tiên, tức là giá trị 2.

Giả sử 2 xâu **Str1** và **Str** có độ dài là **len1** và **len**. Kết quả của thuật toán được trả lại trong biến **pos**. Nếu không tìm thấy (Str1 không nằm trong Str) thì **pos = 0**, ngược lại **pos > 0** là vị trí đầu tiên **Str1** nằm trong **Str**.

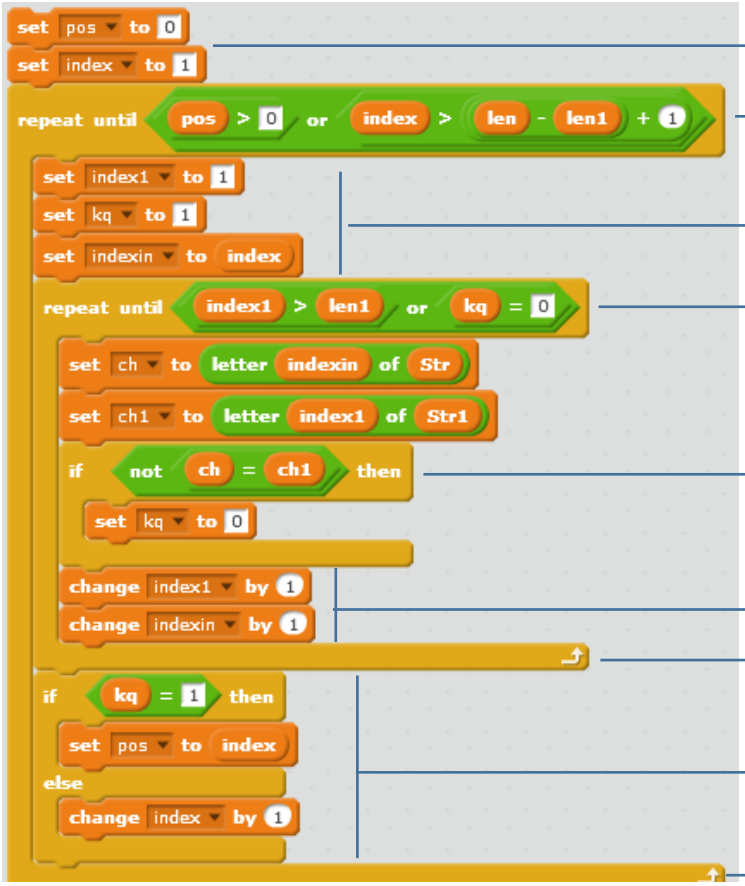
Ý tưởng của thuật toán như sau: Bắt đầu từ vị trí đầu tiên của xâu Str, tiến hành kiểm tra lần lượt các vị trí bắt đầu từ 1 bằng biến **index**. Với mỗi vị trí như vậy cần tiến hành kiểm tra **len1** phần từ tiếp theo xem có trùng với **Str1** không. Nếu trùng thì lập tức dừng vòng lặp, gán giá trị **pos** = **index**. Nếu không trùng thì tiếp tục tăng index và kiểm tra tiếp. Trong vòng lặp kiểm tra bên trong sử dụng thêm các biến nhớ: **indexin**, **index1, ch1, kq**. Biến **kq** có ý nghĩa **kq = 1** nếu đã tìm thấy **Str1** trong **Str**. Trong vòng lặp trong, sử dụng biến nhớ **indexin** để chạy trên xâu **Str** đồng thời với biến **index1** chạy trên **Str1**.



Mô tả thuật toán trên bằng lời tường minh như sau:

```
Gán các giá trị ban đầu: pos = 0, index = 1
Thực hiện lặp cho đến khi pos > 0 hoặc index > len-len1+1
    Gán index1 = 1, kq = 1, indexin = index
    Lặp cho đến khi index1 > len1 hoặc kq = 0
        Đặt ch = ký tự thứ index của Str
        Đặt ch1 = ký tự thứ index1 của Str1
        Nếu ch khác ch1 thì gán kq = 0
        Tăng index1 lên 1
        Tăng indexin lên 1
    Nếu kq = 1 thì
        Gán pos = index
    nếu không thì
        Tăng index lên 1
```

Đoạn chương trình Scratch mô tả thuật toán trên như sau:



Thiết lập các giá trị ban đầu: pos = 0, index = 1

Vòng lặp ngoài, chính

Thiết lập các giá trị ban đầu cho vòng lặp trong, kiểm tra sự trùng nhau giữa Str1 và 1 phần của Str

Vòng lặp trong

Kiểm tra 2 ký tự tương ứng của Str1 và Str

Tăng các biến chạy trong vòng lặp trong

Kiểm tra kết quả của vòng lặp trong, chuẩn bị tăng biến index cho vòng lặp ngoài.

Em hãy hoàn thiện chương trình đầy đủ của bài toán này.

Chương trình sẽ yêu cầu học sinh nhập 2 dữ liệu:

- Xâu dữ liệu gốc Str.

- Xâu dữ liệu cần kiểm tra Str1

Chương trình sẽ thông báo kết quả có tìm thấy hay không. Nếu tìm thấy thì chỉ ra vị trí mà **Str1** nằm trong **Str**.

Câu hỏi và bài tập

1. Viết lại chương trình biến đổi xâu nhị phân sang số thập phân, tính và điền kết quả trong bảng sau:

Xâu nhị phân	**Số thập phân**
110011	
1010101010101	
1111000001111	
100000001	

2. Viết lại chương trình biến đổi số thập phân sang xâu nhị phân, tính và điền kết quả trong bảng sau:

Số thập phân	**Xâu nhị phân**
37	
980	
1001	
50000	

3. Viết chương trình nhập từ bàn phím 1 xâu ký tự bất kỳ Str. Sau đó tiến hành đổi chỗ 2 phần tử bất kỳ trong xâu trên và hiển thị kết quả trên màn hình.

4. Viết chương trình thực hiện công việc sau:

Nhập từ bàn phím 1 xâu ký tự bất kỳ Str, sau đó tiến hành hoán vị ngẫu nhiên các ký tự của Str, kết quả đưa vào xâu Str1. Kết quả thể hiện ra màn hình ca 2 xâu Str và Str1.

5. Viết chương trình thực hiện công việc sau:

- Nhập 1 xâu ký tự bất kỳ **Str** từ bàn phím.

- Thực hiện việc thiết lập xâu **Str1** bằng cách thay đổi ngược lại thứ tự các ký tự của xâu **Str**. Ví dụ nếu ban đầu **Str** = "abcdef" thì kết quả xâu **Str1** = "fedcba".

- Thể hiện kết quả xâu **Str1** trên màn hình.

6. Viết chương trình thực hiện trò chơi đơn giản sau:

Yêu cầu người dùng nhập 1 chữ cái bất kỳ, ví dụ nhập chữ c.

Chương trình sẽ thông báo ngay số 3 là thứ tự của chữ cái này trong bảng chữ cái tiếng Anh.

Nếu ký tự nhập không là chữ cái thì chương trình phát tiếng động "Ốp".

7. Mở rộng các chương trình trò chơi 4 cho bảng chữ cái tiếng Việt (33 chữ cái).

8. Viết chương trình thực hiện trò chơi đơn giản sau:

Yêu cầu người dùng nhập 1 chuỗi chữ số bất kỳ, ví dụ nhập chuỗi 123.

Chương trình sẽ thông báo ngay xâu ký tự bao gồm dãy các chữ cái lấy theo thứ tự từ bảng chữ cái tiếng Anh và ghép lại thành 1 từ. Trong ví dụ trên, từ tiếng Anh được đưa ra là abc.

Nếu xâu ký tự nhập không đúng thì chương trình phát tiếng động "Ốp".

9. Giả sử có xâu ký tự cho trước độ dài **n** và lưu trong biến nhớ **Str**. Viết chương trình sinh tự động các xâu con tách ra từ Str. Số lượng xâu được sinh là ngẫu nhiên trong khoảng 2 - (n-1). Các xâu con này được xếp ngẫu nhiên không theo thứ tự ban đầu được tách ra.

Ví dụ: xâu ban đầu là **abcdegh**.

Chương trình sẽ sinh tự động 3 xâu con như sau: **cde, gh, ab**.

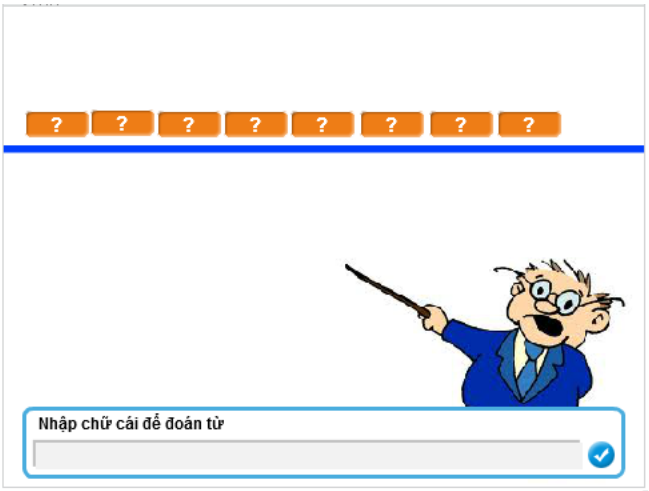
10. Viết chương trình **sửa lỗi tên tiếng Việt** như sau:

- Yêu cầu nhập từ bàn phím 1 tên đầy đủ.
- Sau đó chương trình sẽ xóa đi các ký tự trống ở đầu, cuối hoặc bên trong tên này.
- Thông báo tên đúng sau khi đã sửa lỗi.

Ví dụ nếu nhập tên là:	`"  Bùi  Tiến Thành  "`
thì tên đúng sau khi sửa là:	`"Bùi Tiến Thành"`

Mở rộng

1. Thiết kế trò chơi **tìm từ (hangman)** dạng đơn giản sau.



Game. Hangman.sb2

Từ cần tìm được lưu trong biến nhớ Word (được gán cứng trong chương trình). Giao diện chương trình là trò chơi dự đoán từ này. Từ Word có độ dài <= 8.

Trên màn hình sẽ thể hiện các ô tương ứng với độ dài của từ cần tìm. Các chữ chưa được đoán sẽ hiện dấu ?, các chữ đã đoán rồi sẽ hiện chính xác trên màn hình.

Chương trình sẽ liên tục đưa ra câu hỏi: "Nhập chữ cái để đoán từ". Nhiệm vụ của người chơi là nhập 1 chữ cái. Nếu chữ cái này có trong từ thì nó sẽ hiện ra trên màn hình (có thể 1 hoặc nhiều). Nếu không có thì chương trình thông báo "không có chữ cái này" và yêu cầu nhập tiếp tục, cho đến khi nào lật được hết các chữ cái của từ đã cho.

2. Mở rộng trò chơi **Hangman** trên cho từ với độ dài bất kỳ (trong phạm vi giới hạn cho phép của màn hình, ví dụ < 12).

Bài 16. Làm việc với List 1

Mục đích

Học xong bài này bạn sẽ:

- Hiểu ý nghĩa của List (mảng, dãy giá trị), biết cách thiết lập dãy trong Scratch.

- Ứng dụng List giải 1 số bài tập đơn giản.

- Thuật toán sắp xếp, đổi chỗ, tìm min, max trong dãy.

Bắt đầu

Em đã từng xem, từng biết các thông tin được liệt kê như một danh sách, ví dụ như danh sách học sinh lớp học, danh sách các món ăn, danh sách các tỉnh, thành phố.

Số TT	Họ và tên thí sinh	
1	PHẠM VIỆT	HOÀNG
2	NGUYỄN ĐỖ	VĂN
3	NGUYỄN ĐỖ	VŨ
4	VÕ THỊ THANH	DUYÊN
5	VÕ ANH	KIỆT
6	NGUYỄN TRƯƠNG	TUẤN
7	NGUYỄN ĐÌNH	BẢO
8	LÊ NHẤT	LINH
9	LÊ VIẾT	QUANG
10	HUỲNH NGUYỄN ANH	THƯ
11	TRẦN ĐÌNH	HIỂN
12	PHẠM NHẬT	TƯỜNG
13	LÊ HOÀNG	VŨ
14	NGUYỄN THỊ THANH	HƯƠNG
15	VƯƠNG THỊ ÁNH	MINH

THỰC ĐƠN 1

Mực chiên
Tôm hấp
Gà rô ty
Cá thu sốt
Dạ dày bóp
Hoa lơ xào nạc
Rau luộc
Canh chua cá
Cơm + Tráng miệng

Các tỉnh			Thành phố
An Giang	Hà Nam	Quảng Nam	Cần Thơ
Bà Rịa - Vũng Tàu	Hà Tĩnh	Quảng Ngãi	Đà Nẵng
Bắc Giang	Hải Dương	Quảng Ninh	Hải Phòng
Bắc Kạn	Hậu Giang	Quảng Trị	Hà Nội
Bạc Liêu	Hòa Bình	Sóc Trăng	TP HCM
Bắc Ninh	Hưng Yên	Sơn La	
Bến Tre	Khánh Hòa	Tây Ninh	
Bình Định	Kiên Giang	Thái Bình	
Bình Dương	Kon Tum	Thái Nguyên	
Bình Phước	Lai Châu	Thanh Hóa	
Bình Thuận	Lâm Đồng	Thừa Thiên Huế	
Cà Mau	Lạng Sơn	Tiền Giang	
Cao Bằng	Lào Cai	Trà Vinh	
Đắk Lắk	Long An	Tuyên Quang	
Đắk Nông	Nam Định	Vĩnh Long	
Điện Biên	Nghệ An	Vĩnh Phúc	
Đồng Nai	Ninh Bình	Yên Bái	
Đồng Tháp	Ninh Thuận	Phú Yên	
Gia Lai	Phú Thọ		
Hà Giang	Quảng Bình		

Em hãy liệt kê thêm các danh sách thông tin mà em đã từng biết đến ở nhà, ở trường và ngoài xã hội.

Chúng ta đã được làm quen với khái niệm biến nhớ, là công cụ để lưu trữ các thông tin giúp người lập trình có thể tạo ra được các chương trình hiệu quả hơn. Tuy nhiên mỗi biến nhớ chỉ lưu trữ được 1 giá trị tại 1 thời điểm. Nếu như, ví dụ, chúng ta cần lưu trữ đồng thời tên của tất cả học sinh trong lớp học, thì phải làm như thế nào. Rõ ràng từng biến nhớ cụ thể không thể giải quyết được yêu cầu trên.

Bài học này sẽ giúp em hiểu được một công cụ mới để thực hiện yêu cầu trên đây.

Nội dung bài học

1. Biến nhớ kiểu danh sách (List)

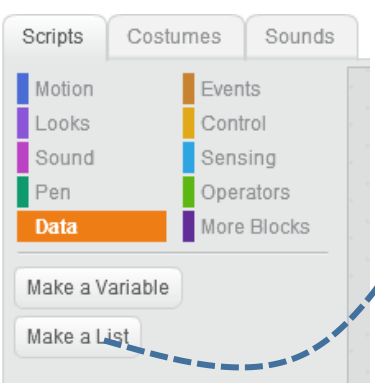
Biến nhớ kiểu danh sách dùng để lưu trữ một dãy giá trị, có thể là số hoặc chữ. Ví dụ dãy tên học sinh sau được đánh số từ 1 đến 6, có thể được lưu trong 1 biến nhớ kiểu danh sách (list).

Hà	Bình	Nguyệt	Vũ Hùng	Vân	Bình
1	**2**	**3**	**4**	**5**	**6**

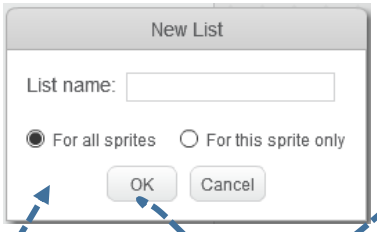
Để mô tả 1 dãy (số, chữ) tổng quát, người ta hay dùng ký hiệu: a_1, a_2, …., a_n. Số hạng tổng quát sẽ ký hiệu là a_i.

Trong Scratch, biến nhớ kiểu danh sách không cần khai báo trước kiểu dữ liệu và số lượng phần tử. Thao tác tạo ra các biến nhớ kiểu danh sách rất đơn giản, trong nhóm lệnh Dữ liệu (Data).

Việc khởi tạo biến nhớ danh sách rất đơn giản, bởi nút lệnh **Make a List** từ nhóm lệnh **Data**.



Nút **Make a List** dùng để khởi tạo 1 biến nhớ dạng List.



Giao diện tạo biến nhớ sạng List. Có 2 loại: biến nhớ chung và biến nhớ riêng cho mỗi nhân vật.

Xuất hiện các lệnh làm việc với biến nhớ List.

Cũng giống như biến nhớ bình thường, biến nhớ dạng danh sách có thể là biến nhớ chung hoặc riêng. Nếu là biến nhớ chung thì tất cả các nhân vật đều có thể sử dụng, cập nhật và thay đổi dữ liệu. Nếu là biến nhớ riêng thì chỉ có nhân vật là chủ mới có các quyền truy cập, sử dụng và thay đổi thông tin.

Việc nhập dữ liệu vào biến nhớ List có thể thực hiện bằng lệnh 

hoặc có thể nhập trực tiếp trên sân khấu của Scratch.

Lệnh 

có tính năng bổ sung thêm 1 thông tin vào cuối danh sách của biến nhớ dạng List.

Ví dụ lệnh 

sẻ bổ sung thêm 1 tên học sinh là "Việt Anh" vào cuối của biến nhớ HS (biến nhớ dạng List).

Chú ý: khác với biến nhớ thông thường, biến dạng List nếu khai báo là dùng riêng cho nhân vật sẽ không hiện trong các thuộc tính của nhân vật, do đó các nhân vật khác sẽ không thể truy cập hay sử dụng các biến nhớ riêng này.

Em hãy giải thích ý nghĩa của các lệnh sau và ghi sang cột bên phải.

Chúng ta sẽ cùng nhau tìm hiểu sâu sắc hơn các ứng dụng của biến nhớ List trong các hoạt động tiếp theo.

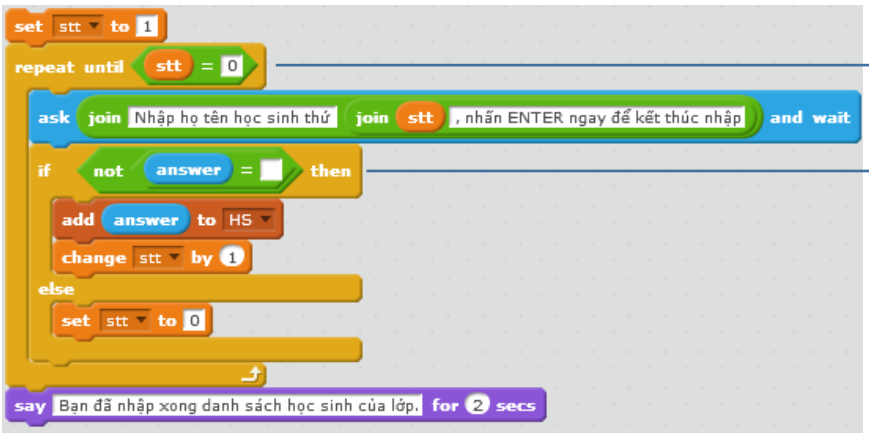
2. Nhập danh sách học sinh lớp học

Em hãy thực hiện chương trình đơn giản sau.

Chương trình yêu cầu người dùng nhập họ tên học sinh trong lớp học vào một danh sách, trong thông báo ghi rõ là đang nhập học sinh thứ mấy. Để kết thúc chỉ cần nhập 1 dữ liệu rỗng, tức là bấm Enter ngay.

Biến nhớ danh sách lưu trữ tên học sinh sẽ được đặt tên là HS. Chúng ta sử dụng thêm 1 biến nhớ nữa **Stt** dùng để lưu số thứ tự của học sinh hiện thời đang nhập. Dùng ngay biến Stt để kiểm tra khi nào thì kết thúc quá trình nhập liệu từ bàn phím.

Chương trình hoàn chỉnh của bài toán này như sau:



Điều kiện kiểm tra vòng lặp là stt = 0

Kiểm tra nếu dữ liệu nhập vào khác rỗng thì bổ sung vào danh sách HS, nếu người dùng không nhập, chỉ nhấn Enter thì đặt stt = 0 để kết thúc quá trình nhập.

3. Các thao tác trực tiếp trên danh sách

Có nhiều cách để thao tác với dữ liệu của biến nhớ danh sách: thông qua các lệnh, nhập trực tiếp trên màn hình Scratch hoặc nhập thông qua tệp (Text File)..

Các thao tác trực tiếp với dữ liệu trên biến nhớ danh sách: bổ sung, chèn, xóa, ...

	Bổ dung dữ liệu <thing> vào cuối của biến nhớ. Lệnh này sẽ làm cho dãy dữ liệu của biến nhớ tăng thêm 1 phần tử nằm cuối danh sách.

	Lệnh này sẽ xóa, hủy khỏi danh sách 1 phần tử được xác định trước. Nếu chỉ số của phần tử ghi trên lệnh nằm ngoài phạm vi của dãy (ví dụ = 0 hoặc > độ dài dãy) thì lệnh không có tác dụng. Một lựa chọn khác của lệnh nếu thiết lập tham số **all** thì lệnh có dạng sau  sẽ xóa toàn bộ dữ liệu của biến nhớ này (hay đưa biến nhớ về trạng thái như lúc mới khởi tạo).
	Lệnh này sẽ chèn thêm 1 phần tử có giá trị <thing> vào trước phần tử thứ <1> của dãy. Nếu thay thế chỉ số cụ thể bằng <random>  thì lệnh sẽ chèn vào 1 vị trí bất kỳ của dãy. Sau lệnh này dãy sẽ tăng lên 1 phần tử.
	Lệnh này sẽ thay thế phần tử thứ <1> cjủa dãy bằng giá trị <thing>. Nếu thay thế chỉ số cụ thể bằng <random>  thì lệnh sẽ thay thế vào vào 1 vị trí bất kỳ của dãy. Chú ý: lệnh này không làm tăng số phần tử của dãy.

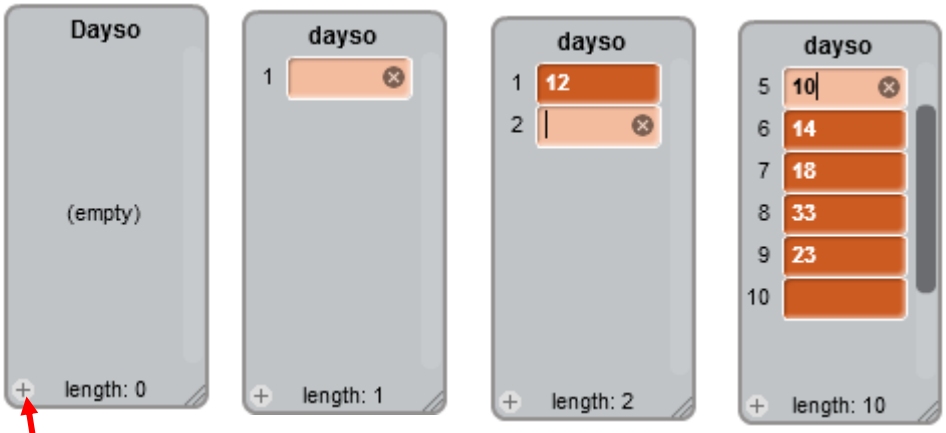
Các lệnh sau sẽ có chức năng truy cập, khai thác dữ liệu danh sách:

	(hàm) trả lại giá trị cụ thể của một phần tử của dãy. Nếu tham số thay bằng <random>  , lệnh sẽ trả lại 1 phần tử bất kỳ trong dãy.
	(hàm) trả lại độ dài của dãy.
	(hàm logic) kiểm tra xem dãy có chứa phần tử được ghi tại vị trí <thing> hay không.
	Lệnh này có tính năng hiển thị tất cả các phần tử của dãy.

Nhập dữ liệu trực tiếp trên màn hình Scratch

Có thể nhập nhanh và trực tiếp dữ liệu kiểu danh sách trên màn hình của Scratch. Muốn vậy chúng ta đặt chế độ hiển thị biến nhớ này trên màn hình.

Hình ảnh của biến nhớ danh sách sẽ hiện ra như hình dưới đây. Bây giờ chúng ta có thể thao tác trực tiếp trên biến nhớ này để nhập dữ liệu.

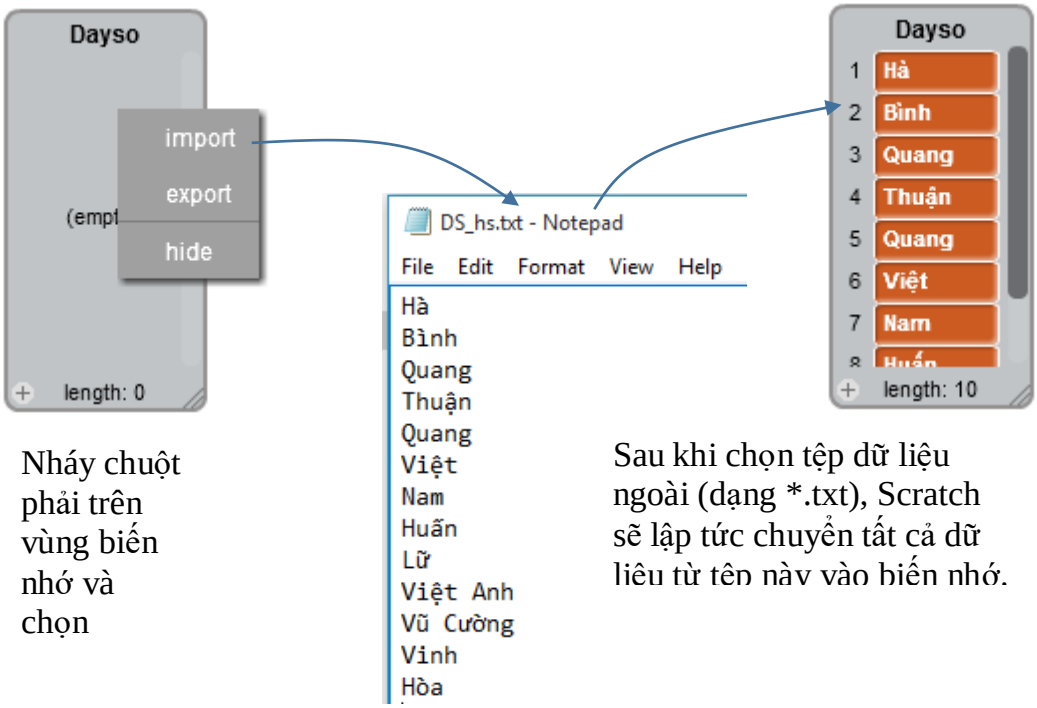


Nháy nút này để bắt đầu tiến hành nhập liệu.

Bước tiếp theo chúng ta nhập trực tiếp dữ liệu theo từng phần từ của danh sách. Có thể dùng các phím lên, xuống để chuyển lên, xuống trong danh sách. Tiến hành nhập cho đến khi xong. Muốn xóa 1 phần tử nháy dấu x bên cạnh ô này.

Chuyển nhập dữ liệu từ Plain Text File

Một cách nhập dữ liệu nhanh và hiệu quả nữa là nhập trước dữ liệu vào 1 tệp văn bảng dạng plane text (tệp *.txt), mỗi đơn vị dữ liệu ghi trên 1 dòng, sau đó chuyển nhập nhanh vào danh sách của Scratch. Qui trình nhập này được mô tả trong hình sau.



Nháy chuột phải trên vùng biến nhớ và chọn

Sau khi chọn tệp dữ liệu ngoài (dạng *.txt), Scratch sẽ lập tức chuyển tất cả dữ liệu từ tệp này vào biến nhớ.

Các bước thực hiện như sau:

- Dữ liệu được nhập trong 1 tệp văn bàn dạng *.txt (plain text).

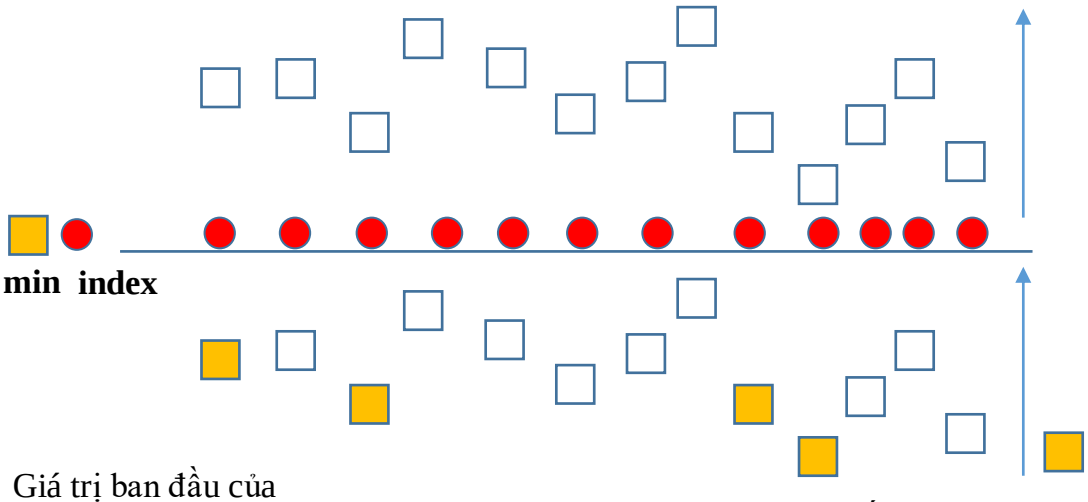
- Trên màn hình sân khấu đặt chế độ hiện biến dữ liệu dạng List.

- Nháy chuột phải lên hình ảnh của biến dãy, chọn lệnh **import**, sau đó chọn tệp văn bản đã có dữ liệu. Dữ liệu từ văn bản này sẽ được đưa vào List.

4. Bài toán tìm Min, Max

Bài toán: cho trước 1 dãy số chứa trong 1 biến nhớ danh sách. Yêu cầu cần tìm và hiển thị phần tử nhó nhất (Min) và lớn nhất (Max) của dãy này.

Thuật toán, hay cách giải quyết bài toán này như sau:



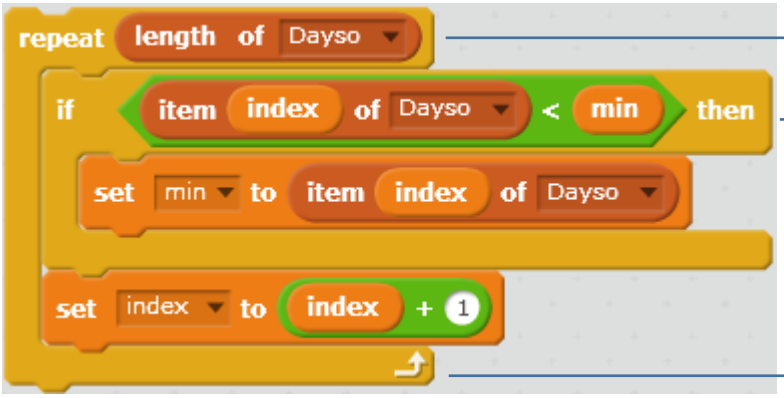
Giá trị ban đầu của biến nhớ **min**, gán cho phần tử đầu tiên của dãy

Khi kết thúc, **min** có giá trị = phần tử nhỏ nhất của dãv

Tạo ra 2 biến nhớ: biến **index** để chạy dọc theo dãy số, biến **min** để lưu kết quả so sánh trung gian. khi di chuyển dọc theo dãy số, so sánh **min** và giá trị phần tử của dãy, nếu giá trị này < **min** thì lập tức thay thế **min** bằng giá trị này. Trong ví dụ mô phỏng trên, biến nhớ **min** được thay đổi 3 lần. Nếu ký hiệu a[i] phần tử của dãy ta có thuật toán được mô tả như sau:

```
Gán index = 1, min = phần tử đầu tiên.
Lặp đúng độ dài của dãy
     nếu min < a[index] thì gán min = a[index]
     index = index + 1
```

Thuật toán mô tả bằng lệnh Scratch như sau:

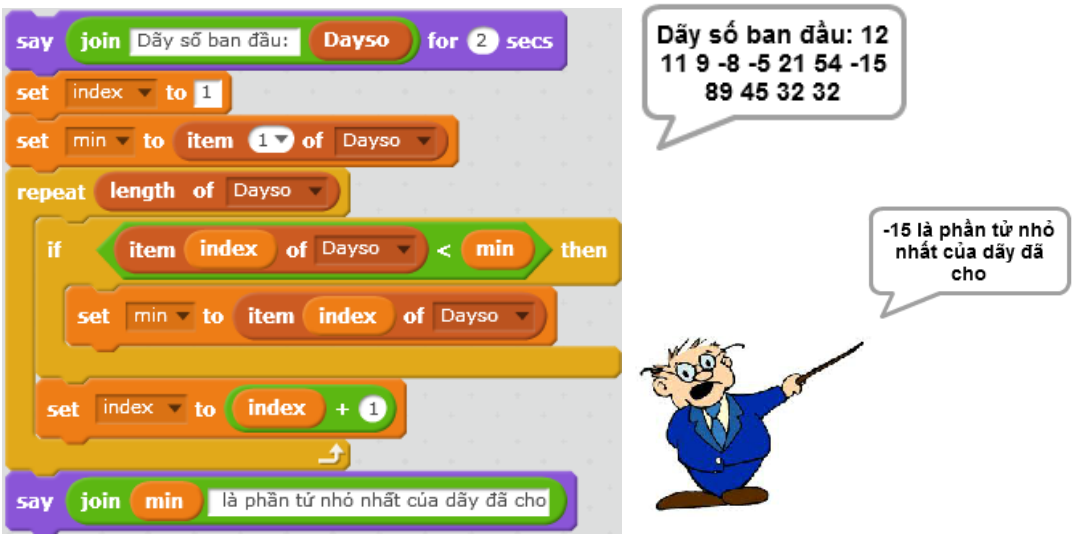


Vòng lặp với số lần lặp = độ dài dãy số.

So sánh phần tử tương ứng của dãy với **min**, nếu < **min** thì thay thế **min** bằng giá trị này.

Chương trình hoàn chỉnh như sau:

Min-Max.sb2



Em hãy viết chương trình tương tự tìm giá trị phần tử lớn nhất (Max) của 1 dãy số cho trước.

5. Bài toán tìm kiếm giá trị trong dãy

Tìm kiếm thông tin là 1 bài toán lớn có rất nhiều ứng dụng rộng rãi trên thực tế. Trong hoạt động này chúng ta cùng tìm hiểu một vài trường hợp đơn giản nhất của công việc này.

a) Bài toán tìm 1 phần tử

Cho trước 1 danh sách dữ liệu, cần tìm ra 1 phần tử thỏa mãn 1 điều kiện nào đó. Ví dụ thực tế của bài toán này rất nhiều. Ví dụ:

- Tìm trong lớp 1 bạn nam có chiều cao lớn hơn 1,7m.

- Tìm trong dãy 1 số là số nguyên tố.

Chúng ta cùng tìm hiểu và giải quyết bài toán này.

Bài toán cụ thể có thể đơn giản hóa như sau: Giả sử dãy dữ liệu là dãy số, cần tìm 1 phần tử trong dãy trên có giá trị = **a**. Dãy số trên được lưu trong biến nhớ danh sách có tên **dayso**.

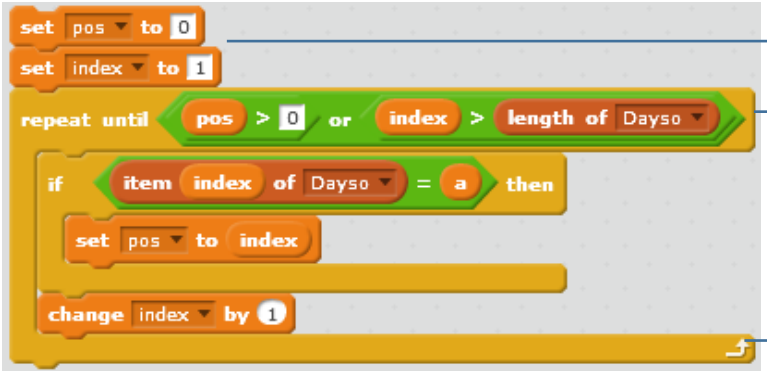
Đầu vào (Input): Danh sách **dayso**, giá trị **a**.

Đầu ra (Output): Thông báo "không tìm thấy" hoặc "có" và chỉ ra số thứ tự của phần tử được tìm thấy có giá trị **a**.

Chúng ta sử dụng biến nhớ **pos** để lưu lại chỉ số của phần tử tìm thấy trong dãy. Nếu pos = 0 tức là không tìm thấy.

Thuật toán đơn giản này mô tả như sau:

```
Gán giá trị ban đầu pos = 0, index = 1
Lặp cho đến khi pos > 0 hoặc hết độ dài của dãy
     Nếu a = dayso[index] thì gán pos = index
          nếu không thì index = index + 1
```

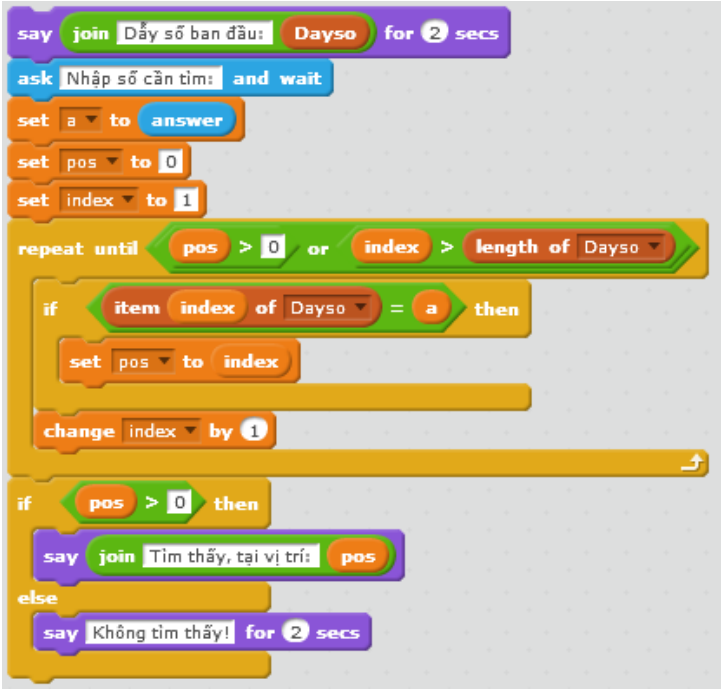


Gán các giá trị ban đầu: **pos** = 0, **index** = 1.

Vòng lặp chính có điều kiện dừng nếu pos > 0.

Chương trình hoàn chỉnh như sau:

Tim kiem 1 - 1 phan tu.sb2



b) Bài toán tìm tất cả các phần tử

Cho trước 1 danh sách dữ liệu, cần tìm ra tất cả các phần tử của dãy thỏa mãn 1 điều kiện nào đó, đếm số lượng các phần tử đã tìm được. Ví dụ:

- Tìm trong lớp tất cả các bạn có tên "Hùng".

- Tìm tất cả các số chia hết cho 3 trong dãy số cho trước.

Cách làm bài này tương tự như bài toán tìm 1 phần tử, chỉ khác là chúng ta sẽ duyệt dãy từ đầu đến cuối và có thêm 1 biến nhớ nữa **count** dùng để đếm số lượng các phần tử tìm được. Nếu **count = 0** tức là không tìm thấy.

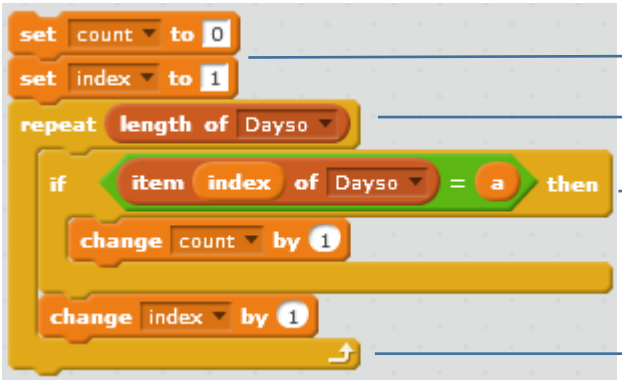
Có 2 loại yêu cầu cho bài toán này:

1. Đếm số phần tử thỏa mãn điều kiện tìm kiếm.

2. Liệt kê tất cả các phần tử thỏa mãn điều kiện tìm kiếm.

Chúng ta cùng thực hiện bài toán 1 ở trên. Bài toán 2 sẽ đưa vào dưới dạng 1 bài tập có gợi ý.

Đây là thuật toán của bài toán 1, đếm số phần tử thỏa mãn điều kiện tìm kiếm, được mô tả bằng lệnh Scratch.



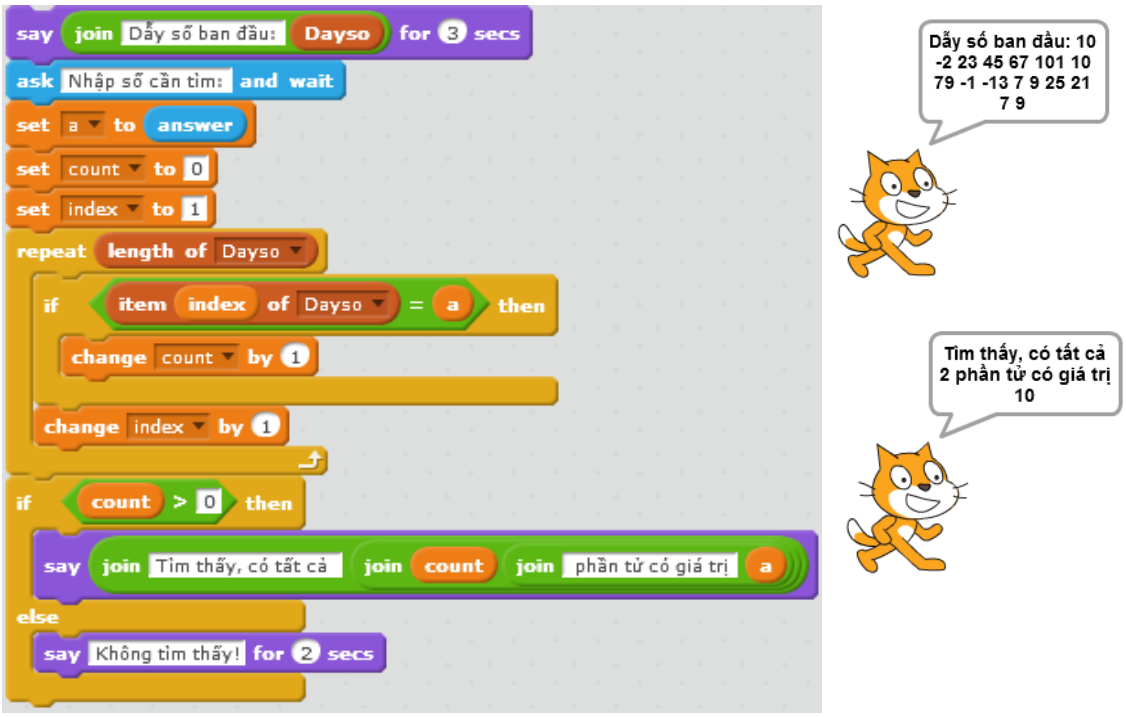
Chương trình chính yêu cầu như sau:

- Dãy số cho trước đã được nhập từ trước và

- Chương trình yêu cầu nhập giá trị số cần tìm, sau đó đưa ra kết quả: hoặc thông báo không tìm thấy, hoặc thông báo đã tìm thấy và số lượng phần tử tìm được.

Chương trình hoàn chỉnh như sau.

Tim kiem 2 - tat ca phan tu.sb2



6. Bài toán sắp xếp dãy

Sắp xếp 1 dãy dữ liệu cho trước là 1 bài toán rất hay gặp trên thực tế. Ví dụ:

- Sắp xếp lớp đứng xếp hàng theo thứ tự từ thấp đến cao.

- Sắp xếp tên học sinh trong lớp theo thứ tự ABC (thứ tự từ điển).

- Sắp xếp các tỉnh thành phố theo diện tích, dân số.

Chúng ta sẽ cùng nhau tìm hiểu bài toán sắp xếp dãy số trong hoạt động này. Hãy bắt đầu từ 1 ví dụ đơn giản: sắp xếp dãy 4 số sau theo thứ tự tăng dần.

Dãy số gốc: **10 7 9 4**

1. Tìm phần tử nhỏ nhất là 4, đổi chỗ với 7, thu được dãy: **4 10 9 7**

2. Tìm phần tử nhỏ nhất trong 3 số cuối, đó là 7, đổi chỗ cho 10, thu được: **4 7 9 10**

Thuật toán sử dụng trên được gọi là **Sắp xếp chọn**.

Đổi chỗ 2 phần tử của dãy

Trong cách làm trên chúng ta thấy thao tác lõi là "đổi chỗ" hai phần tử trong dãy, hay tổng quát hơn là đổi chỗ 2 biến nhớ bất kỳ trong bộ nhớ máy tính.

Cho 2 biến nhớ **m**, **n** với các giá trị đã gán. Tìm cách đổi giá trị giữa 2 biến nhớ này. Cách thực hiện phổ biến nhất là sử thêm 1 biến nhớ trung gian, ký hiệu là **tg**. Các bước như sau:

Gán m cho tg, tg = m

Gán n cho m, m = n

Gán tg cho n, n = tg

Đoạn chương trình đổi chỗ 2 phần tử thứ **i, j** của dãy số **dayso** như sau.

Bài toán sắp xếp dãy: Thuật toán chọn

Giả sử dãy cần sắp xếp theo thứ tự tăng dần là: $a_1, a_2, \ldots, a_n$

Thuật toán chọn thực hiện như sau:

Thuật toán sắp xếp chọn.

Bước 1. Tìm trong các số $a_2, a_3, \ldots, a_n$ phần tử nhỏ nhất, giả sử là a_j. Đổi chỗ a_1, a_j nếu $a_1 > a_j$

Bước 2. Tìm trong các số $a_3, a_4, \ldots, a_n$ phần tử nhỏ nhất, giả sử là a_j. Đổi chỗ a_2, a_j nếu $a_2 > a_j$

.....

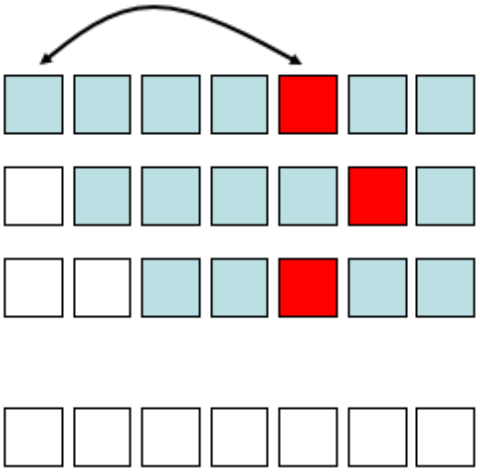
Bước n-2. Tìm trong các số a_{n-1}, a_n phần tử nhỏ nhất, giả sử là a_j. Đổi chỗ a_{n-2}, a_j nếu $a_{n-2} > a_j$

Bước n-1. Đổi chỗ a_{n-1}, a_n nếu $a_{n-1} > a_n$.

Có thể mô tả thuật toán này theo cách viết sau:

```
Vòng lặp n-1 lần, cho i chạy từ 1 đến n-1
     Chọn ra phần tử nhỏ nhất trong các số a_{i+1}, a_{i+2}, …, a_n
     Giả sử chỉ số này là j
     Nếu a_i > a_j thì đổi chỗ a_i, a_j
```

Sơ đồ của thuật toán chọn có thể mô tả như trong hình ảnh sau.



Tìm phần tử nhỏ nhất, đổi chỗ với a_1

Tìm phần tử nhỏ nhất trong số từ a_2 đến a_n, sau đó đổi chỗ với a_2

Tìm phần tử nhỏ nhất trong số từ a_3 đến a_n, sau đó đổi chỗ với a_3

Sau n-1 bước, việc sắp xếp đã hoàn thành.

Để thiết kế chương trình chúng ta tạo ra các biến nhớ sau:

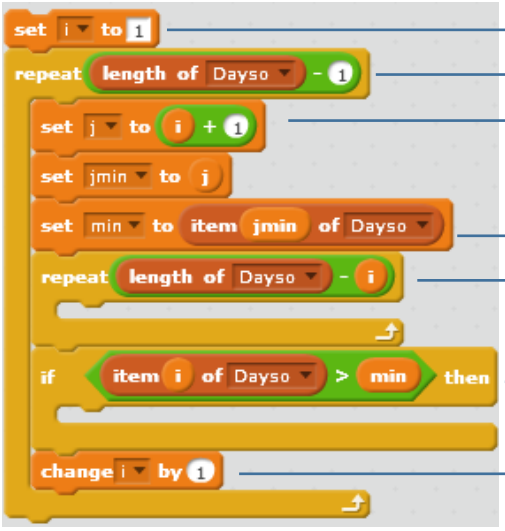
i, j - các biến nhớ chạy ở vòng ngoài và vòng lặp trong.

jmin - chỉ số phần tử nhỏ nhất được tìm thấy ở vòng lặp bên trong.

min - giá trị phần tử nhỏ nhất, dùng trong vòng lặp trong tìm min.

tg - biến nhớ trung gian dùng cho việc đổi chỗ 2 phần tử.

Sơ đồ thuật toán được mô tả như sau:



Thiết lập thông số ban đầu cho biến chạy **i = 1**.

Vòng lặp ngoài theo i, chạy từ 1 đến n-1

Chuẩn bị cho vòng lặp trong. Thiết lập các tham số ban đầu cho **j, jmin** và **min**.

Vòng lặp trong với j chạy từ i+1 đến n

Ra khỏi vòng lặp trong, kiểm tra phần tử thứ i với min.

Tăng i lên 1, vòng lặp

Toàn bộ đoạn chương trình chính như sau:

Sap xep day so.sb2

Vòng lặp ngoài, n-1 bước.

Thiết lập các giá trị ban đầu của thuật toán.

Vòng lặp trong thứ i, n-i bước.

Tìm phần từ nhỏ nhất trong các số từ a_i đến a_n.

Kiểm tra và đổi chỗ phần tử nhỏ nhất đó với a_i.

Em hãy hoàn thiện chương trình trên.

Câu hỏi và bài tập

1. Tạo 1 biến danh sách và nhập trực tiếp thành 1 dãy số.

2. Tạo 1 biến danh sách và nhập trực tiếp 1 danh sách tên học sinh trong lớp.

3. Thực hiện bài toán chèn 1 phần tử vào dãy đã sắp xếp đúng: cho trước 1 dãy số đã được sắp xếp theo thứ tự tăng dần, cho trước 1 số P. Hãy viết đoạn chương trình chèn số P này vào dãy trên sao cho dãy vẫn được sắp xếp đúng.

4. Cho trước 1 dãy số, hãy viết chương trình để sắp xếp lại dãy số này theo các yêu cầu sau:

(a) Dãy giảm dần.

(b) Các số âm lên trước, các số dương ở phía sau.

(c) Các số chẵn lên trước, số lẻ ở phía sau.

5. Cho trước 1 danh sách tên học sinh trong lớp. Hãy viết chương trình để sắp xếp lại danh sách học sinh này theo thứ tự từ điển.

6. Cho trước 1 dãy số. Hãy viết chương trình sinh 1 dãy số là 1 hoán vị ngẫu nhiên của dãy số ban đầu.

7. Bài toán tìm kiếm tất cả.

Cho trước dãy số a_1, a_2, …. , a_n và số a.

Hãy viết chương trình nhập dãy số a[i] vào 1 biến dãy **Dayso**, nhập a từ bàn phím và thông báo trên màn hình:

- Có tìm được số nào trong dãy trên bằng a hay không.

- Nếu có thì đếm có bao nhiêu số thỏa mãn tìm kiếm như vậy.

- Liệt kê tất cả các số đã tìm thấy ra màn hình.

8. Viết chương trình cho phép nhập từ bàn phím số n và thông báo, thể hiện ra tất cả các số nguyên tố < n.

9. Cho trước dãy số a_1, a_2, …. , a_n được nhập vào biến List **Dayso**.

Hãy viết chương trình sắp xếp lại dãy này theo thứ tự tăng dần bằng thuật toán được mô phỏng dưới đây (theo ngôn ngữ Scratch).

```
Gán n = độ dài Dayso.
Vòng lặp ngoài i chạy từ 1 đến n-1
     Vòng lặp trong, j chạy từ i+1 đến n
          Nếu ai > aj thì
               Đổi chỗ 2 phần tử ai và aj.
Kết thúc, hiển thị dãy ai.
```

10. Viết chương trình tạo 1 biến List và sinh tự động 100 phần tử đầu tiên của dãy này là dãy số chẵn đầu tiên.

11. Viết chương trình tạo 1 biến List và sinh tự động 50 phần tử đầu tiên của dãy này là dãy các số nguyên tố đầu tiên.

12. Cho trước 2 dãy số cho trước **Dayso1** và **Dayso2**. Viết chương trình tạo ra biến List **Dayso**, **Dayso** thu được bằng cách hợp 2 dãy **Dayso1** và **Dayso2** và sau đó sắp xếp lại theo thứ tự tăng dần.

Mở rộng

Thiết kế 1 chương trình ứng dụng thao tác với dữ liệu là danh sách học sinh như sau:

Khi khởi động chương trình, Mèo sẽ thông báo tên các học sinh hiện có trong danh sách.

Trên màn hình có 3 nút lệnh.

Khi nháy lên nút **Input Data**, quá trình nhập trực tiếp dữ liệu từ bàn phím bắt đầu. Nhập liên tục cho đến khi nhấn Enter không nhập gì cả.

Khi nháy lên nút **Delete Data**, chương trình sẽ yêu cầu người dùng nhập tên học sinh muốn xóa, sau đó kiểm tra và xóa tên này trong danh sách.

Khi nháy lên nút **View Data**, chương trình hiển thị toàn bộ danh sách học sinh hiện có.

Bài 17. Làm việc với List 2

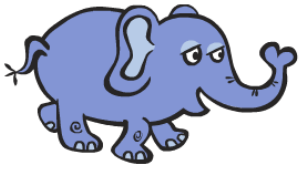
Mục đích

Trong bài này bạn sẽ học và biết:

- Một số ứng dụng đơn giản của dữ liệu List
- Một số thuật toán nâng cao đối với dũ liệu List
- Ứng dụng thực tế của danh sách.

Bắt đầu

Em hãy đọc yêu cầu của 1 trò chơi **Tìm hiểu động vật** sau đây và suy nghĩ xem có thể thiết kế hệ thống dữ liệu như thế nào để có 1 chương trình, bài học hay.



Phần mềm sẽ ra liên tục các câu hỏi, em cần trả lời trực tiếp bằng cách gõ bàn phím. Nếu gõ đúng tên con vật thì hình ảnh của động vật này sẽ hiện trên màn hình. Nếu gõ không đúng, gõ sai thì thông báo "sai" hoặc hình ảnh sẽ không hiện.

Cứ như vậy cho đến khi em nhấn Enter ngay thì trò chơi kết thúc.

Các gợi ý bằng câu hỏi:

- Giả sử em có 10 hình ảnh con vật dùng làm dữ liệu cho trò chơi này, em sẽ thiết kế bộ dữ liệu như thế nào.
- Có 1 cách đơn giản nhất là tạo ra 10 biến nhớ và 10 nhân vật trên sân khấu dùng để lưu trữ tên của các con vật và thể hiện hình ảnh con vật.
- Nếu giả sử em muốn tăng số lượng con vật lên thì em sẽ phải làm gì? Có cách nào khi tăng con vật mà không phải tăng thêm biến nhớ hoặc nhân vật không?

Chúng ta hãy tìm hiểu bài toán này và nhiều bài toán tương tự khác trong bài học này.

Nội dung bài học

1. Tìm hiểu động vật

Animal.sb2

Trong hoạt động đầu tiên này, chúng ta sẽ quay trở lại bài toán, trò chơi Tìm hiểu động vật và cùng xem nếu sử dụng các biến nhớ danh sách thì chương trình sẽ trở nên rất mạch lạc, sáng sủa.

Chương trình sẽ liên tục đưa ra câu hỏi "Em thích con vật gì?". Người chơi trả lời bằng cách nhập tên con vật từ bàn phím.

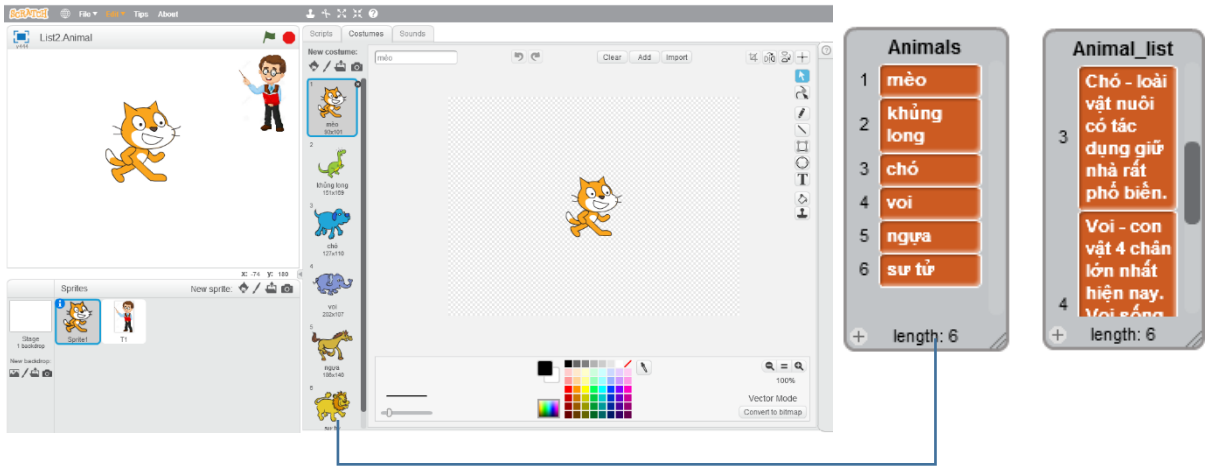
Nếu nhập đúng hình ảnh con vật sẽ hiện trên màn hình và giáo sư sẽ thông báo thông tin kỹ hơn về động vật này.

Trò chơi kết thúc khi người chơi nhấn Enter ngay mà không nhập gì.

Chúng ta sẽ sử dụng 2 biến danh sách chính là **Animal** và **Animal_list**. Biến Animal là danh sách tên các con vật cần tìm hiểu. Biến **Animal_list** sẽ lưu thông tin tra cứu chi tiết tương ứng của các con vật đã có trong danh sách **Animal**. Ví dụ 1 bảng thông tin như sau. Chú ý bảng này có thể có số dòng lớn tùy ý.

Animal	**Animal_list**
mèo	Mèo - loài vật nuôi trong nhà rất phổ biến trong mọi gia đình.
khủng long	Khủng long - loài vật khổng lồ nhất trên trái đất, đã tuyệt chủng từ cách đây hàng triệu năm
chó	Chó - loài vật nuôi có tác dụng giữ nhà rất phổ biến.
voi	Voi - con vật 4 chân lớn nhất hiện nay. Voi sống nhiều ở châu Phi và châu Á.
ngựa	Ngựa - loài vật thường dùng để kéo xe, chở đồ, rất phổ biến trên thế giới.
sư tử	Sư tử - là động vật ăn thịt lớn nhất trên cạn, được mệnh danh là chúa rừng xanh.

Tiếp theo chúng ta sẽ thiết kế cách lưu trữ hình ảnh các con vật. Chỉ cần tạo 1 nhân vật cho công việc này. Hình ảnh các con vật khác nhau sẽ tương ứng với trang phục của nhân vật này.



Thiết lập hệ thống trang phục (costume) với tên trùng khớp với bảng **Animal**.

Bây giờ chúng ta sẽ thiết kế sơ đồ hoạt động của chương trình này. Sơ đồ này có thể tóm tắt trong hình sau, bắt đầu từ **sân khấu**.

Yêu cầu học sinh nhập tên 1 con vật từ bàn phím. Thông tin nhập sẽ được lưu trong biến nhớ **name**.

Nếu **name** khác rỗng → xử lý tiếp.

Nếu **name** = rỗng → dừng chương trình.

Gửi thông điệp: **Go**

Gửi thông điệp: **End**

Chuyển điều khiển sang nhân vật **thầy giáo**.

Tìm kiếm thông tin trong bảng **Animal_list** về con vật ghi trong biến nhớ name. Nếu tìm thấy thì hiển thị thông tin con vật này trên màn hình thông qua lệnh **say**.

Chuyển điều khiển sang nhân vật **con mèo**.

Chuyển trang phục sang tên lưu trong biến nhớ **name** và thể hiện.

Chuyển điều khiển sang nhân vật **con mèo**.

Ẩn, không thể hiện trên màn hình.

Chúng ta sẽ bắt đầu từ sân khấu. Sân khấu trong Scratch cũng có cửa sổ lệnh của riêng mình. Trong bài toán này, sân khấu sẽ bắt đầu chương trình với sự kiện

.

Khi bắt đầu chương trình, sân khấu sẽ thực hiện các lệnh sau:

- Yêu cầu liên tục người dùng nhập tên con vật cần tra cứu.

- Nếu người dùng không nhập gì, nhấn **Enter** thì gửi thông điệp End và toàn bộ chương trình kết thúc.

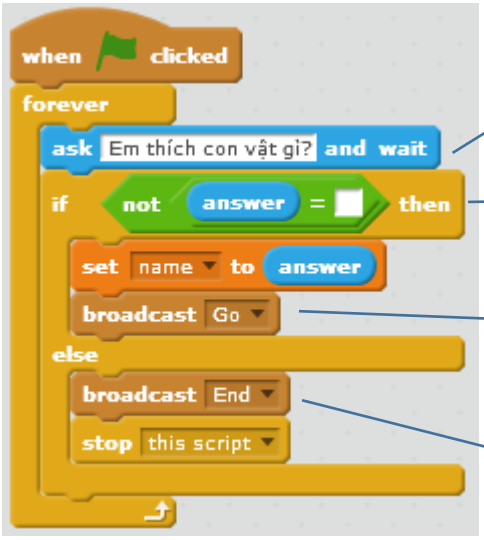
- Nếu tên con vật được nhập thì gán tên con vật vào biến nhớ **name** và gửi thông điệp **Go** và chuyển điều khiển sang 2 nhân vật chính của chương trình.

Nhận được thông điệp Go, hai nhân vật Mèo và Thầy giáo sẽ xử lý thông tin khác nhau.

Mèo: chuyển costume đến tên con vật để thể hiện.

Thầy giáo: tìm tên con vật trong danh sách **Animal_list** để hiện thông tin thuộc tính của con vật này.

Đây là chương trình của sân khấu.



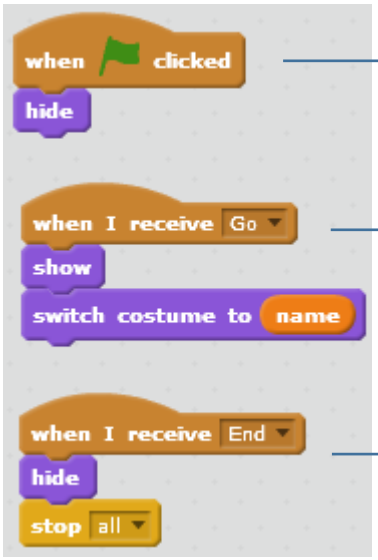
Thực hiện lệnh yêu cầu nhập tên con vật trong 1 vòng lặp vô hạn.

Kiểm tra xem dữ liệu nhập có rỗng hay không.

Nếu dữ liệu nhập không rỗng thì lưu kết quả nhập vào biến **name** và truyền thông điệp **Go** cho 2 nhân vật của chương trình.

Nếu dữ liệu nhập làg rỗng thì truyền thông điệp **End**.

Đây là cửa sổ lệnh của nhân vật con mèo (con mèo có 5 trang phục khác).



Khi bắt đầu chương trình, nhân vật không hiển thị trên màn hình.

Khi nhận thông điệp **Go**, chuyển trang phục sang giá trị ghi trong biến nhớ **name** và hiển thị trên màn hình.

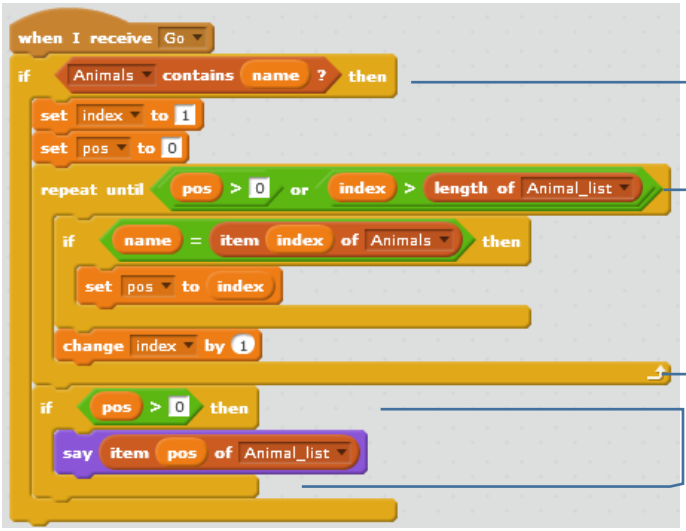
Khi nhận thông điệp **End**, ẩn không hiển thị và dừng toàn bộ chương trình.

Chúng ta sẽ cùng xem Thầy giáo cần xử lý như thế nào.

Khi nhận thông điệp **Go**, nhân vật thầy giáo cần thực hiện xử lý biến nhớ **name**, kiểm tra xem **name** có tên trong bảng **Animal** không. Nếu có thể tìm ra chỉ số chính xác dòng chưa tên tương ứng với biến **name** này, sau đó sẽ thể hiện trên màn hình dữ liệu có trong bảng **Animal_list**. Chỉ số chính xác dòng chứa tên con vật tương ứng với **name** được lưu trong biến nhớ **pos**.

Dựa trên biến nhớ name, thầy giáo sẽ tìm tên tương ứng trong bảng **Animals**, sau đó tìm thuộc tính tương ứng trong bảng **Animal_list**.

Cửa sổ lệnh của thầy giáo.



Kiểm tra điều kiện tên con vật trong biến **name** có nằm trong danh sách **Animal** hay không, nếu có mới xử lý tiếp.

Vòng lặp này có tính năng tìm ra giá trị **pos** là vị trí của biến **name** trong danh sách **Animal**.

Thông báo thông tin chi tiết về con vật, lấy thông tin từ bảng **Animal_list**.

2. Bài toán sinh hoán vị, tập con ngẫu nhiên

Trên thực tế có rất nhiều bài toán đòi hỏi việc sinh ngẫu nhiên các tập hợp con, sinh ngẫu nhiên các hoán vị, … Trong hoạt động này chúng ta sẽ cùng giải quyết một vài bài toán đó trong môi trường Scratch.

Bài toán 1. Cho trước 1 xâu ký tự **Str**, cần sinh ra 1 xâu **Strout** là hoán vị ngẫu nghiên của xâu **Str** ban đầu.

Bài toán 2. Cho trước dãy số (hoặc chữ) $a_1, a_2, ….., a_n$. Cần sinh ra 1 hoán vị ngẫu nhiên của dãy này, đưa ra dãy sau: $b_1, b_2, ….., b_n$ là hoán vị của dãy ban đầu.

Bài toán 3. Cho trước dãy số (hoặc chữ) $a_1, a_2, ….., a_n$. Cần sinh ra một dãy ngẫu nhiên m phần tử của dãy trên: $b_1, b_2, …, b_m$ với m cho trước.

Chúng ta sẽ giải quyết các bài toán trên.

Bài toán 1. Cho trước 1 xâu ký tự **Str**, cần sinh ra 1 xâu **Strout** là hoán vị ngẫu nghiên của xâu **Str** ban đầu.

Ý tưởng ban đầu của thuật toán này khá đơn giản: lấy ngẫu nhiên các ký tự từ xâu **Str** và bổ sung vào xâu **Strout**. Yêu cầu của bài toán là sau khi thực hiện giá trị xâu **Str** cần giữ nguyên, không thay đổi. Sử dụng các lệnh cụ thể của Scratch có thể viết lại cách làm trên dưới dạng cây lệnh như sau:

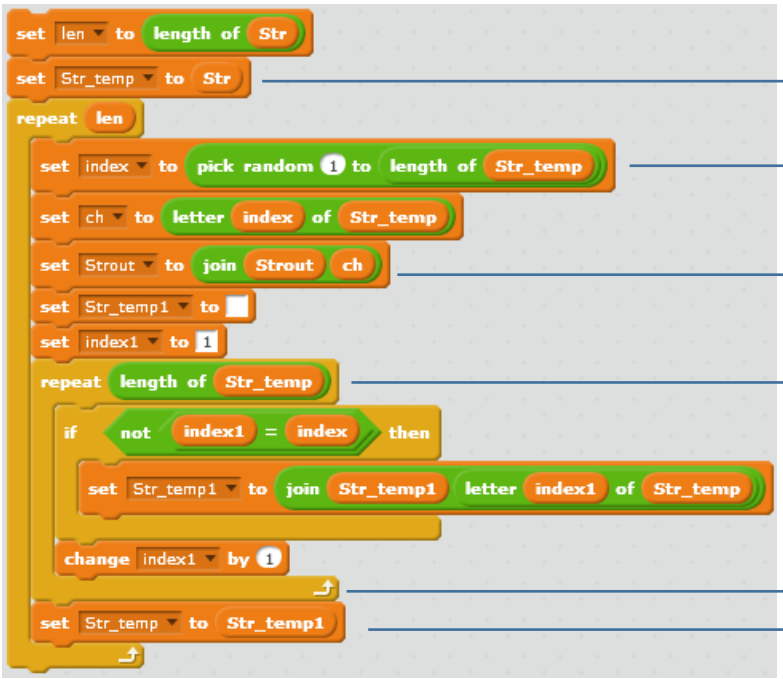
```
Đặt Strout = rỗng, len = độ dài xâu Str
Đặt Str_temp = Str
Thiếp lập lặp với số vòng lặp = len
    Lấy ra 1 ký tự bất kỳ của xâu Str_temp
    Đưa ký tự này vào cuối của Strout
    Xóa ký tự này từ Str_temp
```

Trong Scratch không có lệnh xóa 1 ký tự của xâu, do đó dòng cuối cùng của thuật toán trên cần thực hiện như 1 chương trình. Thuật toán xóa 1 ký tự có chỉ số **index** khỏi 1 xâu **Str_temp** như sau:

```
Đặt Str_temp = rỗng, index1 = 1, len = độ dà xâu Str
Thiếp lập lặp với số vòng lặp = len
     Kiểm tra: nếu index1 <> index thì
          Lấy ra ký tự thứ index1 của Str
          Đưa ký tự này vào cuối của Str_temp
     Tăng index lên 1
Thiết lập Str = Str_temp
```

Kết hợp 2 thuật toán trên, chúng ta có chương trình trong Scratch.

Hoan vi xau.sb2



Gán **Str** cho **Str_temp**

Lấy 1 ký tự ngẫu nhiên từ **Str_temp** và đưa vào cuối **Strout**.

Xóa ký tự này khỏi **Str_temp** kết quả đưa vào **Str_temp1**.

Gán lại **Str_temp1** cho **Str_temp**

Bài toán 2. Cho trước dãy số (hoặc chữ) $a_1, a_2, ….., a_n$. Cần sinh ra 1 hoán vị ngẫu nhiên của dãy này, đưa ra dãy sau: $b_1, b_2, ….., b_n$ là hoán vị của dãy ban đầu.

Input: Dãy List, **Output:** Dãy Listout

Bài toán 2 cách thực hiện tương tự bài toán 1, chỉ có điểm khác là thực hiện trên biến danh sách, chứ không phải trên biến xâu ký tự.

Cách đơn giản: lấy từ phần tử ngẫu nhiên của List và đưa vào cuối của Listout, sau đó xóa phần tử này khỏi danh sách gốc List.

Lặp m lần (= length(List)).

Lấy 1 phần tử ngẫu nhiên của **List** và đưa vào cuối của **Listout**, sau đó xóa phần tử này từ **List**.

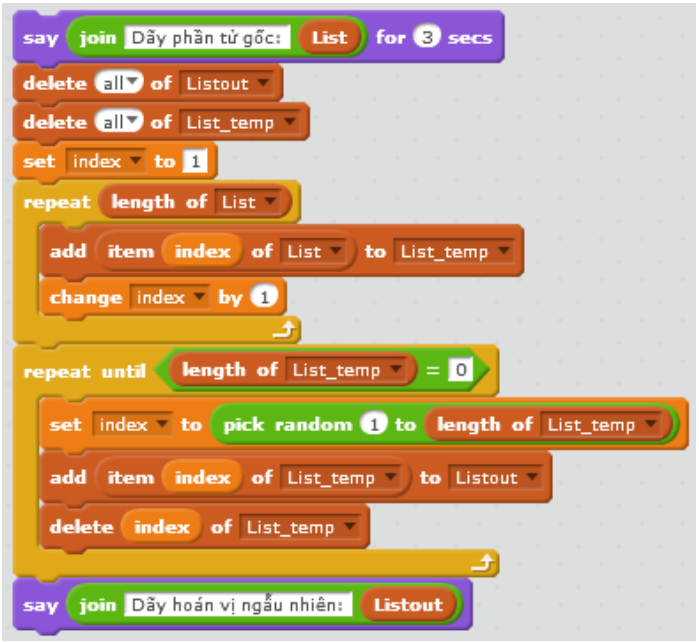
Cách trên sẽ xóa dữ liệu từ danh sách gốc.

Chương trình dưới đây được viết lại hoàn chỉnh, danh sách gốc List không bị xóa dữ liệu. Do vậy cần tạo thêm 1 biến danh sách trung gian List_temp.

Khi bắt đầu chương trình cần tạo 2 dãy rỗng là **Listout** và **List_temp**. Sao chép toàn bộ dữ liệu từ dãy ban đầu **List** sang **List_temp**, sau đó chỉ thao tác trên dãy **List_temp**.

Chương trình trên Scratch như sau:

Hoan vi List.sb2



Bài toán 3. Cho trước dãy số (hoặc chữ) $a_1, a_2, ..., a_n$. Cần sinh ra một dãy ngẫu nhiên m phần tử của dãy trên: $b_1, b_2, ..., b_m$ với m cho trước.

Input: Dãy **List**, số m; **Output:** Dãy **Listout**.

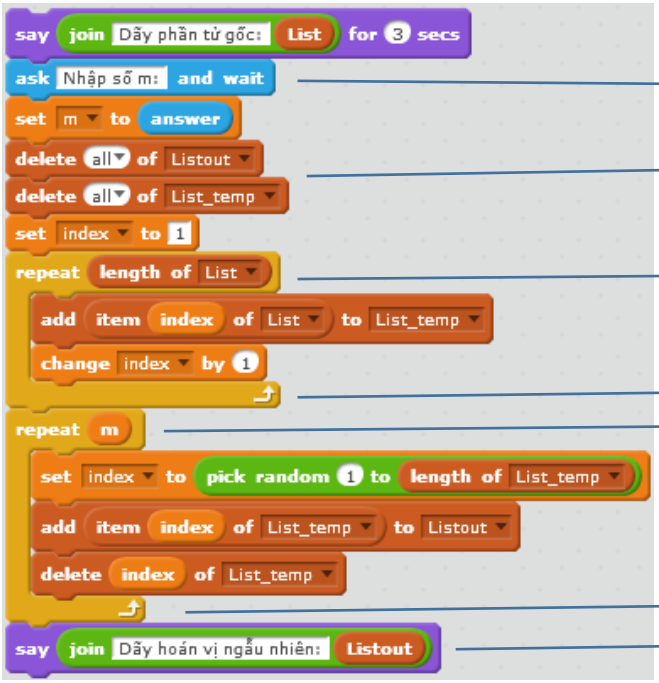
Ý tưởng và cách thực hiện bài toán này tương tự bài toán 2 và khá đơn giản như sau, chú ý rằng dãy gốc List yêu cầu không bị xóa.

Thuật toán lấy ra ngẫu nhiên **m** phần tử từ 1 dãy gốc ban đầu **List** được viết như sau:

```
Đặt dãy List_temp và Listout = rỗng.
Gán từng phần tử của dãy List vào List_temp.
Thiết lập vòng lặp độ dài m
     Lấy ra 1 phần tử ngẫu nhiên của List_temp.
     Đưa phần tử này vào cuối của danh sách Listout.
     Xóa phần tử này từ List_temp.
```

Chương trình hoàn chỉnh trên Scratch như sau.

Sinh tap con cua day.sb2



Nhập số m từ bàn phím.

Thiết lập 2 dãy số rỗng

Gán dãy **List** vào **List_temp**

Vòng lặp m lần: lấy 1 phần từ ngẫu nhiên trong **List_temp**, đưa phần tử này vào **Listout**, sau đó xóa khỏi **List_temp**.

Đưa Listout ra màn hình

3. Từ điển sinh bài kiểm tra trắc nghiệm

Sinh cau hoi trac nghiem.sb2

Chúng ta sẽ cùng tìm hiểu một ứng dụng nữa của dữ liệu danh sách, khi chúng được dùng như dữ liệu từ điển để sinh ngẫu nhiên các bài kiểm tra trắc nghiệm.

Mô mình bài toán cụ thể như sau.

Giả sử có 2 bảng dữ liệu, bao gồm 2 danh sách quốc gia và thủ đô tương ứng của quốc gia đó. Ví dụ 1 bảng như vậy.

Quốc gia	Thủ đô
Argentina	Buenos Aires
Armenia	Yerevan
Australia	Canberra
Austria	Viên
Azerbaijan	Baku
Croatia	Zagreb
Cuba	La Habana
Síp	Nicosia
Cộng hòa Czech	Praha
Venezuela	Caracas
Việt Nam	Hà Nội
Liên bang Nga	Moscow
Vương quốc Anh	Luân Đôn
Hoa kỳ	Washington
Uruguay	Montevideo
Uzbekistan	Tashkent

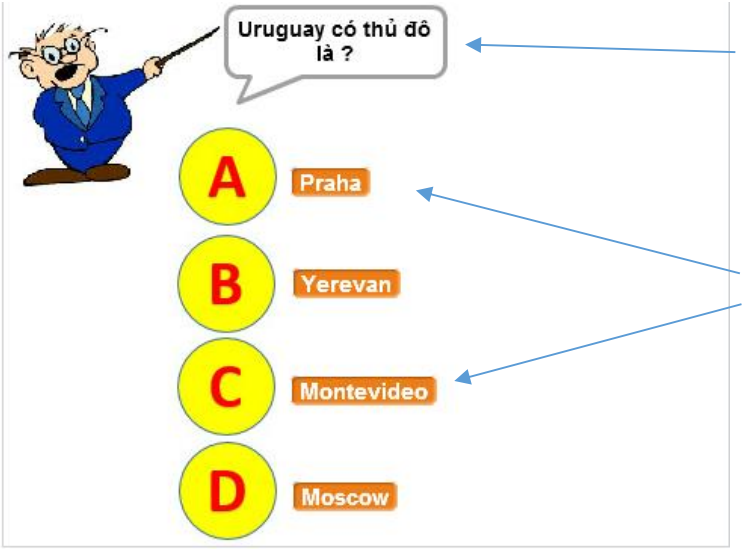
Chương trình cần đưa ra các loại câu hỏi trắc nghiệm sau, dữ liệu được lấy ngẫu nhiên từ các bảng dữ liệu trên.

(a) <Thủ đô> là thủ đô của nước nào?

(b) <Quốc gia> có thủ đô là ?

Các câu hỏi được sinh ngẫu nhiên, nội dung câu hỏi là loại trắc nghiệm khách quan (1 đúng nhiều sai) cũng được sinh ngẫu nhiên, lấy dữ liệu từ 2 bảng trên. Phần mềm sẽ tự động sinh bài kiểm tra trắc nghiệm theo 1 trong 2 loại câu hỏi như trên.

Giao diện câu hỏi hiện trên màn hình có thể như hình sau:

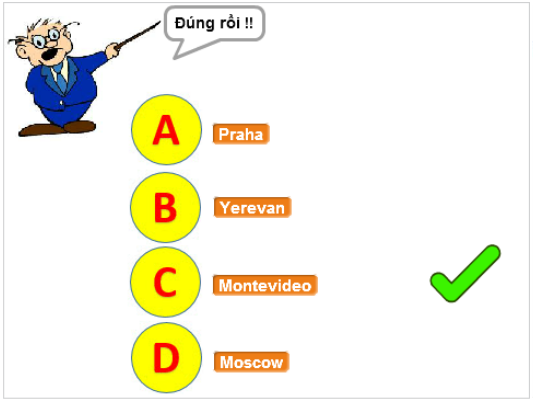


Trên màn hình có 4 nút A, B, C, D. Bên cạnh là bộ 4 dữ liệu được lấy ra và hiển thị ngẫu nhiên trên màn hình.

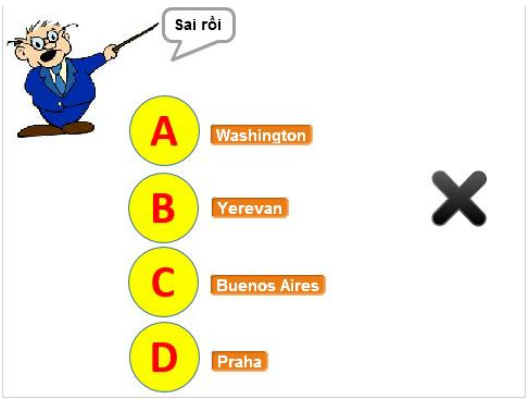
Người thực hiện bài kiểm tra bằng cách nháy chuột lên 1 trong các đáp án.

Chương trình sẽ thông báo ngay đúng hay sai, hiển thị dấu đúng, sai và chuyển sang câu hỏi tiếp theo.

Giao diện khi làm bài có dạng như sau:

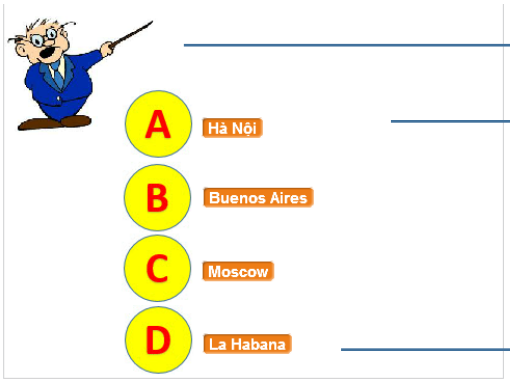


Giao diện khi nháy chọn phương án đúng, chú ý dấu tích bên cạnh phương án đã nháy.



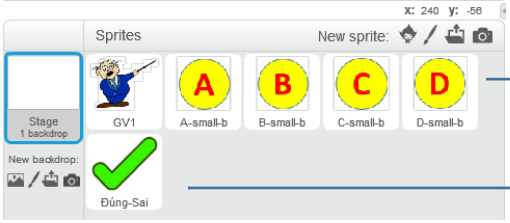
Giao diện khi nháy chọn phương án sai, chú ý dấu tích chéo bên cạnh phương án đã nháy.

Mô hình thiết kế nhân vật và sân khấu như sau:



Nhân vật thầy giáo sẽ ra câu hỏi trắc nghiệm trên màn hình.

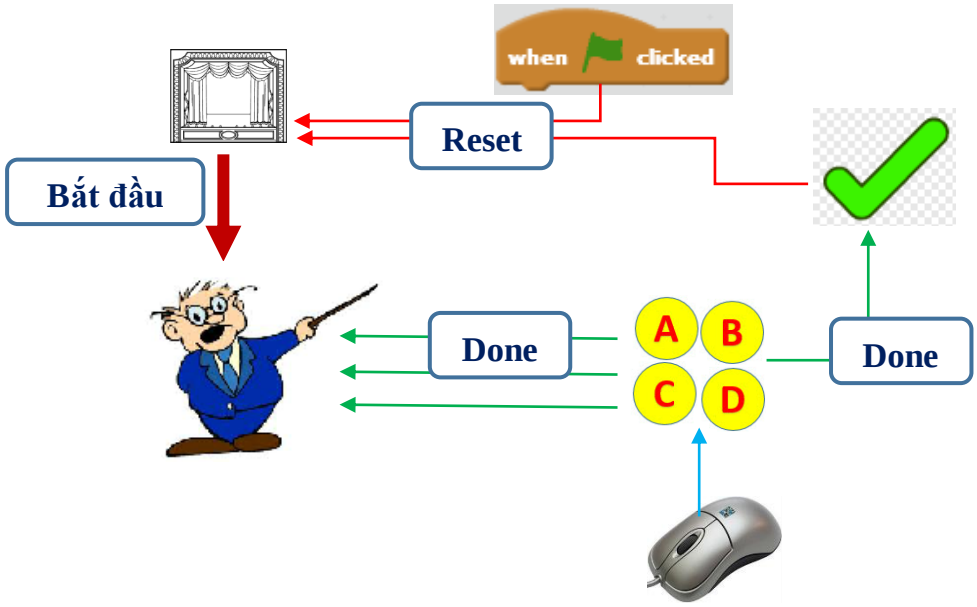
Vị trí các nút A, B, C, D và 4 biến nhớ dùng để thể hiện bộ dữ liệu được sinh ngẫu nhiên cho câu hỏi trắc nghiệm.



Nhân vật và sân khấu chính: thầy giáo, 4 nút A, B, C, D và nút dấu Đúng - Sai.

Sân khấu sẽ tham gia giải quyết bài toán này.

Sơ đồ hoạt động của chương trình như sau:



Sau đây là giải thích cho sơ đồ trên.

- Bắt đầu chương trình, thông điệp **Reset** (làm lại) được đưa ra và chuyển tới sân khấu nền để tính toán, xử lý các thông tin ban đầu.

- Nhận được thông điệp **Reset**, sân khấu sẽ thực hiện việc sinh ngẫu nhiên dạng câu hỏi trắc nghiệm, sinh ngẫu nhiên bộ 4 phương án, trong đó có 1 phương án đúng. Dữ liệu được lấy từ 2 bảng **dsQG** và **dsThudo**. Sau khi sinh xong bộ dữ liệu sẽ phát đi thông điệp **Bắt đầu (Start)**. Chú ý là sau khi bộ dữ liệu được sinh, các thông tin này sẽ tự động hiển thị trên màn hình.

- Thầy giáo khi nhận được thông điệp **Bắt đầu** thì thực hiện công việc sau: thể hiện thông tin câu hỏi trên màn hình và chờ người chơi trả lời.

- Người sử dụng trả lời câu hỏi do thầy giáo đưa ra bằng cách nháy chuột lên 1 trong các nút A, B, C, D.

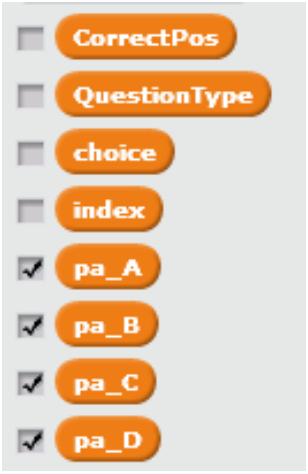
- Nút ABCD (1 trong A, B, C, D) khi nhận được cảm biến chuột thì lập tức gán biến nhớ choice cho phương án mà người dùng trả lời, sau đó phát thông điệp **Done** (đã xong). Thông điệp này sẽ phát đồng thời cho thầy giáo và nút dấu Đúng-Sai.

- Thầy giáo khi nhận thông điệp **Done** sẽ lập tức kiểm tra xem đáp án người dùng đã chọn là đúng hay sai và thông báo đúng / sai ngay trên màn hình.

- Nút dấu Đúng - Sai khi nhận thông điệp **Done** sẽ lập tức kiểm tra xem đáp án người dùng đã chọn là đúng hay sai và tính toán, thay đổi trang phục và dùng lệnh **stamp** (sau khi đã **show**) để in dấu Check (đúng) hoặc Chéo (sai) tại vị trí tương ứng mà người dùng đã nháy chuột.

- Nút Đúng - Sai chờ 3 giây sau thì phát thông điệp **Reset**. Khi nhận thông điệp Reset thì nút dấu Đúng - Sai sẽ ẩn bằng cách thực hiện lệnh **hide**.

Gợi ý thiết kế chương trình theo bộ dữ liệu, biến nhớ sau:



Có 2 kiểu câu hỏi, lưu trong bảng **QuestionBank**. Các phương án được lấy ngẫu nhiên từ bảng **dsQG** và lưu trong bảng **Listout**. Các giá trị cụ thể của các phương án lưu trong các biến **pa_A, pa_B, pa_C, pa_D**. Số thứ tự phương án đúng lưu trong biến **CorrectPos**. Phương án do người dùng lựa chọn lưu trong biến **choice**.

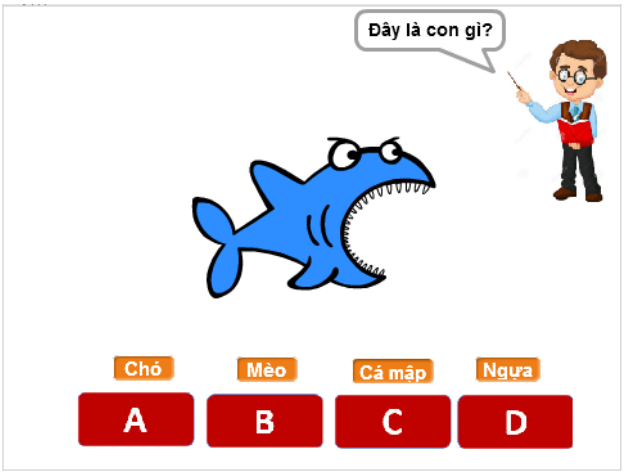
Các biến nhớ dạng List của chương trình.

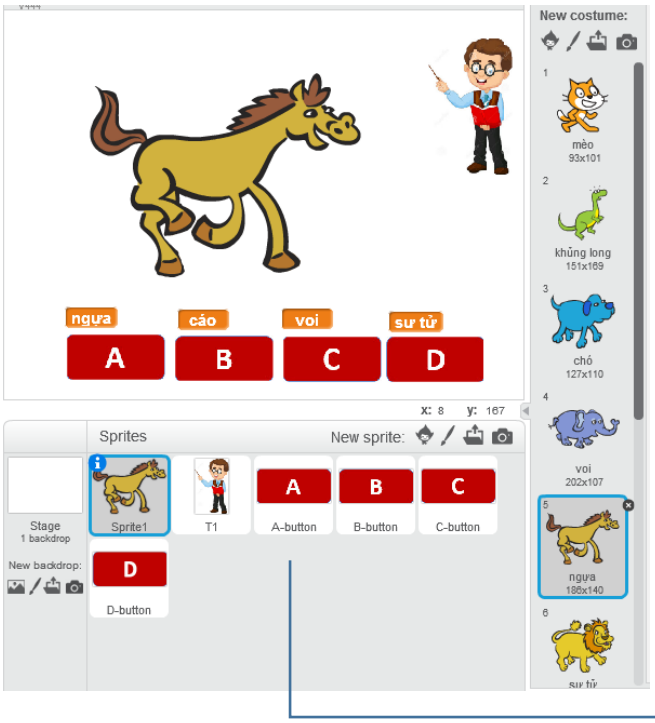
Bài tập: em hãy viết chương trình hoàn chỉnh cho bài toán trên.

4. Bài luyện trắc nghiệm có hình ảnh

Bài toán trong hoạt động này có ý nghĩa gần tương tự như trong hoạt động 3 về dữ liệu sinh bài kiểm tra trắc nghiệm. Dữ liệu và hình ảnh của hoạt động này lấy từ bài toán Tìm hiểu động vật của hoạt động 1.

Puzzle.sb2

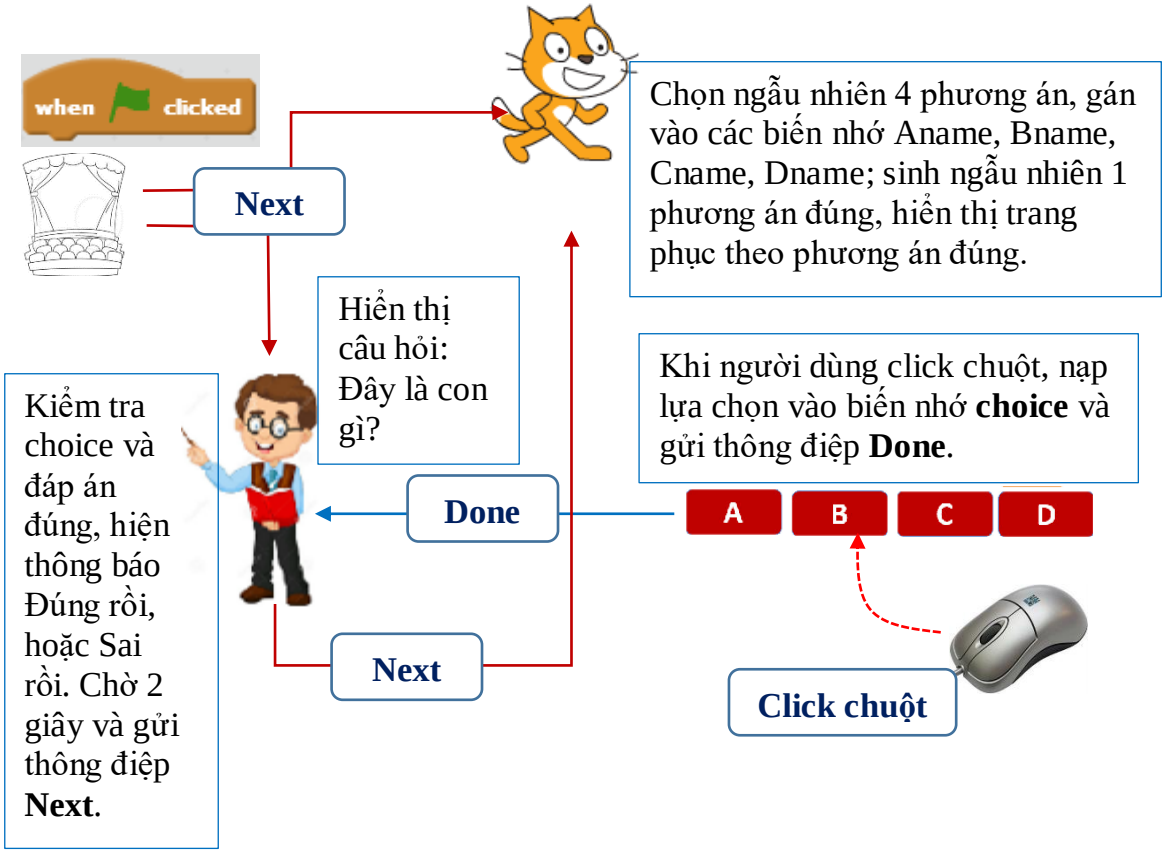




Hệ thống trang phục của nhân vật này tương thích với 2 bảng dữ liệu chính: **Animal** và **Animal_list**.

Hệ thống sân khấu và nhân vật tham gia hoạt động trong bài toán này: nhân vật chính, thầy giáo, 4 nút phương án và sân khấu

Sơ đồ làm việc của chương trình thể hiện trong mô hình sau.



Bài tập: em hãy viết chương trình hoàn chỉnh cho bài toán trên.

5. Sử dụng nhiều danh sách có liên kết

Trong ví dụ trên chúng ta sử dụng 2 biến nhớ dạng danh sách ds_QG và ds_Thudo có liên quan đến nhau theo nghĩa 1-1, nghĩa là mỗi dòng của bảng này tương ứng với chính dòng đó của bảng kia. Đây là cách tạo quan hệ đơn giản giữa hai bảng. Thực tế các quan hệ giữa các bảng có thể phức tạp hơn. Chúng ta hãy quan sát 1 ví dụ của quan hệ như vậy.

Giả sử chúng ta có 3 biến nhớ dạng danh sách như sau:

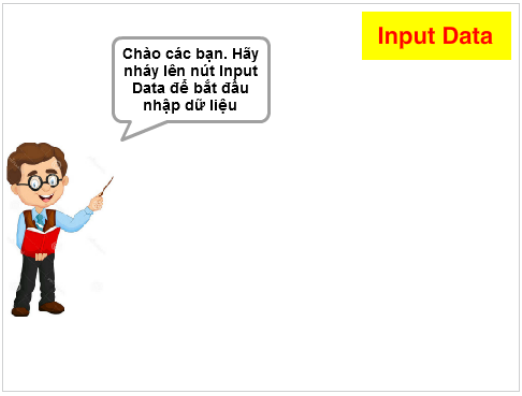
Tỉnh	**Chỉ số**
Thái Nguyên	1
Hà Nội	2
Thanh Hóa	3
Thái Bình	2
Quảng Ngãi	3
Lâm Đồng	4
Cần Thơ	5
Kiên Giang	5
Gia Lai	4
Cao Bằng	1

Vùng
1. Trung du Bắc Bộ
2. Đồng bằng Bắc Bộ
3. Trung Bộ
4. Tây Nguyên
5. Nam Bộ

Bài toán 1. Viết chương trình nhập các bảng dữ liệu có liên kết trên.

Giả sử đã có danh sách các vùng miền, cần lập chương trình nhập danh sách các tỉnh, thành phố với chỉ số liên kết vùng miền chính xác.

Nhap bang co lien ket.sb2



Đầu tiên chương trình thông báo vị trí của nút lệnh nhập dữ liệu. Khi nháy lên nút Input Data, em bắt đầu nhập dữ liệu. Việc nhập dữ liệu được tiến hành theo 2 bước:

Bước 1: nhập tên tỉnh, thành phố (không được trùng với tên đã có).

Bước 2: nhập chỉ số vùng, miền.

Em hãy thiết kế và hoàn thiện chương trình này.

Bài toán 2. Bài toán tra cứu dữ liệu

Giả sử đã có danh sách các vùng miền, cần lập chương trình yêu cầu nhập chỉ số vùng miền, chương trình sau đó sẽ đưa ra danh sách tất cả các tỉnh thành phố thuộc vùng miền này.

Đầu tiên chương trình yêu cầu em nhập 1 số là chỉ số vùng, miền muốn tra cứu dữ liệu.

Sau đó chương trình sẽ tính toán và hiện kết quả tra cứu là số lượng và danh sách cụ thể các tỉnh, thành phố thuộc vùng, miền đã nhập.

Nếu nhập không đúng chỉ số vùng, miền, chương trình sẽ thông báo lỗi nhập liệu.

Em hãy thiết kế và hoàn thiện chương trình này.

Câu hỏi và bài tập

1. Cho trước 1 dãy số tự nhiên bất kỳ, hãy viết chương trình thực hiện các công việc sau:

- Đếm và liệt kê các số chẵn của dãy trên.

- Đếm và liệt kê các số là nguyên tố trong dãy trên.

2. Giả sử chúng ta đã có 1 dãy số hoặc chữ: $a_1, a_2, \ldots, a_n$. Viết chương trình nhập số tự nhiên m và sinh ra ngẫu nhiên các số $b_1, b_2, \ldots, b_m$ lấy theo thứ tự từ trái sang phải của dãy ban đầu.

3. Dãy số Fibonaci là dãy số 1, 1, 2, 3, 5, 8, 13, …. tức là dãy $a_1, a_2, \ldots, a_n$ …. thỏa mãn điều kiện:

$a_1 = a_2 = 1$

$a_n = a_{n-1} + a_{n-2}$ với $n > 2$

Hãy viết chương trình nhập số m từ bàn phím và thể hiện trên màn hình m số hạng đầu tiên của dãt Fibonaci này.

4. Viết chương trình thực hiện công việc sau:

Yêu cầu nhập từ bàn phím danh sách tên học sinh trong lớp học, việc nhập kết thúc khi người dùng không nhập gì và nhấn ENTER. Khi đó chương trình sẽ thông báo số lượng học sinh đã nhập và hiện danh sách học sinh đã nhập.

5. Viết chương trình thực hiện công việc sau:

Chương trình sẽ liên tục yêu cầu nhập 1 số từ bàn phím, việc nhập kết thúc khi người dùng không nhập gì và nhấn ENTER. Chương trình sẽ đưa các số đã nhập vào một biến dãy số cho trước. Tuy nhiên chương trình sẽ phải có chức năng tự lọc không để trùng số trong dãy, nếu 1 số khi nhập bị trùng thì chương trình sẽ bỏ qua và không đưa số này vào dãy. Khi nhập xong, chương trình sẽ hiện danh sách dãy số đã nhập.

6. Giả sử cho trước 2 biến dạng dãy số, biến **Names** lưu danh sách tên học sinh, biến **Provinces** lưu tên tỉnh, thành phố là quê hương của các bạn này. Giả sử mỗi bạn học sinh này có 1 ảnh.

Thiết kế chương trình thực hiện các công việc sau:

- Thiết lập 1 nhân vật là HS, từ nhân vật này sẽ tạo ra các costumes là dãy hình ảnh tương ứng với dãy hình ảnh học sinh mà chúng ta đang có.

- Khi chạy chương trình sẽ hỏi người dùng nhập 1 tên học sinh, nếu tên này có trong danh sách thì sẽ hiện:

+ Tên học sinh này.

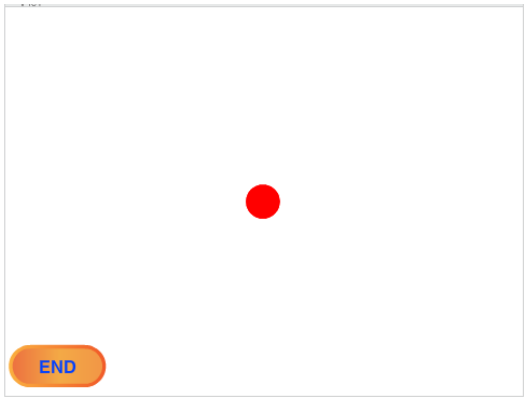
+ Tỉnh, thành phố là quê của của HS này.

+ Hiện hình ảnh học sinh ở giữa màn hình.

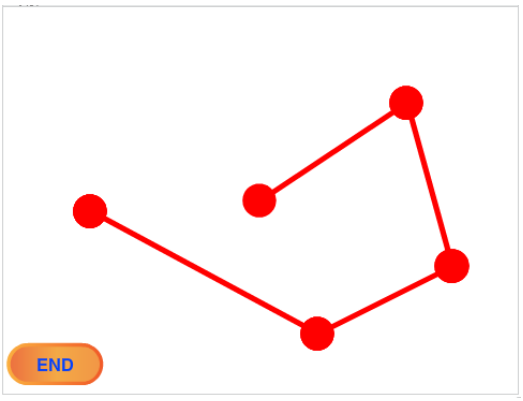
- Nếu người dùng nhập tên không đúng thì có thông báo "nhập sai" trên màn hình.

7. Viết chương trình **Click and Connect** sau:

Chương trình bao gồm 2 nhân vật: nút lệnh End và hình tròn đỏ.



Khi bắt đầu chương trình, hình ảnh nút End và hình tròn như trên.



Mỗi lần nháy chuột trên màn hình (bên trong sân khấu), chương trình sẽ nối 1 đoạn thẳng từ vị trí nháy chuột lần trước. Các tọa độ của điểm đã nháy sẽ được lưu trong 2 dãy số dạng List. Nháy lên nút End sẽ kết thúc chương trình.

8. Tạo và nhập trước trong chương trình 2 dãy dữ liệu:

a) Dãy các từ tiếng Anh.

b) Dãy các giải nghĩa từ này bằng tiếng Việt.

Viết chương trình sinh tự động các câu hỏi trắc nghiệm dạng sau:

Type 1: Từ tiếng Anh nào có ý nghĩa như thể hiện dưới đây.

Type 2: Chọn cách giải thích đúng ý nghĩa của từ tiếng Anh sau.

Cả 2 trường hợp trên đều có 4 phương án, trong đó có 1 phương án đúng.

Thiết kế giao diện của chương trình sao cho đẹp nhất.

9. Tạo và nhập trước 1 dãy dữ liệu là các từ tiếng Anh.

Viết chương trình thực hiện công việc sinh tự động các câu hỏi trắc nghiệm như sau:

- Lấy ngẫu nhiên 1 từ tiếng Anh trong dãy trên, từ này là phương án đúng.

- 3 phương án sai sẽ được tự động sinh từ phương án đúng bằng cách bỏ đi 1 ký tự hoặc hoán đổi vị trí 2 ký tự.

- Sắp xếp 4 từ này một cách ngẫu nhiên.

Đưa ra giao diện để người dùng lựa chọn từ tiếng Anh đúng.

Thiết kế giao diện của chương trình sao cho đẹp nhất.

10. Thiết lập 2 dãy dữ liệu.

- Dãy 1 lưu trữ tên các cuốn sách.

- Dãy 2 lưu trữ tên tác giả sách tương ứng.

Viết chương trình hiển thị liên tục và ngẫu nhiên câu hỏi sau "Tác giả cuốn sách ….. là ai?".

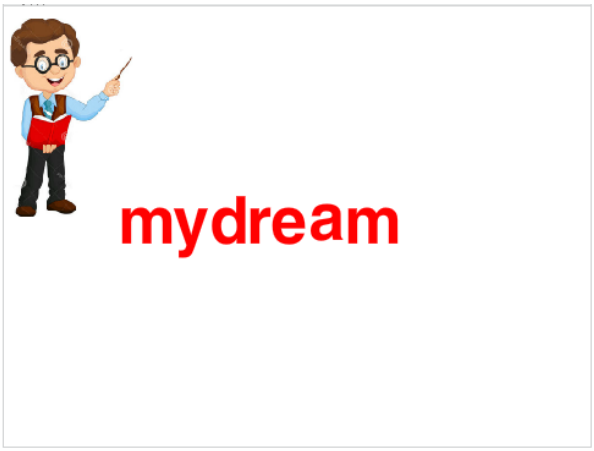
Người dùng sẽ phải nhập tên tác giả, nếu nhập đúng chương trình sẽ thông báo đúng và vỗ tay, nếu nhập sai, chương trình sẽ thông báo sai.

Mở rộng

Hãy thiết kế và viết chương trình mô phỏng bài học phát âm tiếng Anh có âm thanh sau.

Phần mềm yêu cầu nhập 1 từ tiếng Anh bất kỳ, sau đó thông báo và thể hiện cách phát âm của từ này với âm thanh thật.

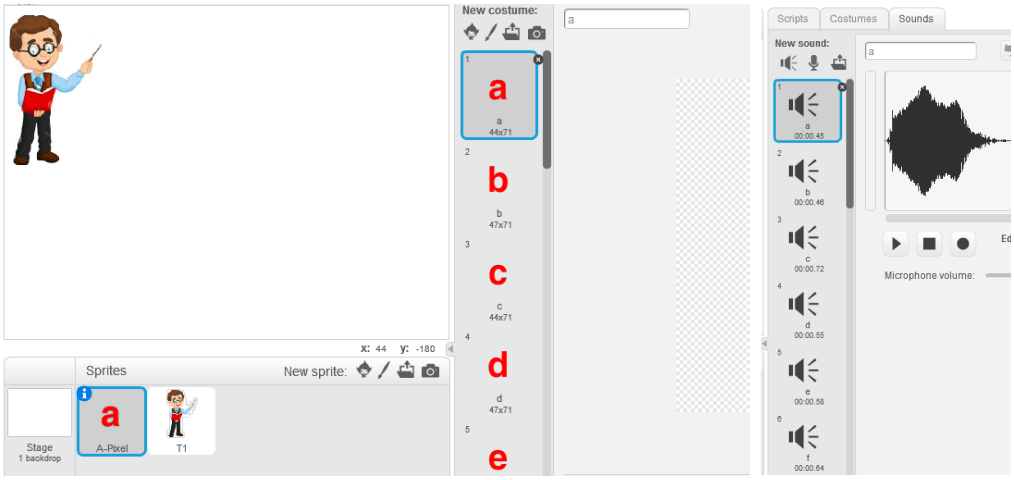
Spelling Words.sb2



Yêu cầu bổ sung của chương trình:

Khi thực hiện việc phát âm, chương trình sẽ đưa ra màn hình lần lượt các chữ cái của từ đã cho ngay trên màn hình, đưa chữ và âm thanh đồng thời.

Gợi ý: Chương trình chỉ cần có 2 nhân vật: thầy giáo và chữ cái. Chữ cái sẽ có 1 bộ trang phục đầy đủ bảng chữ cái tiếng Anh và đầy đủ các âm thanh phát âm của các chữ cái này.


New costume:
Scripts
Costumes
Sounds
New sound:
Microphone volume:
x: 44 y: -180
Sprites
New sprite:
Stage
1 backdrop
A-Pixel
T1

Bài 18. Thủ tục 1

Mục đích

Học xong bài này em sẽ:

- Biết và hiểu được vai trò của thủ tục như 1 nhóm con chương trình nhằm giúp giải quyết công việc dễ hơn.

Bắt đầu

Tư duy chia, phân rã một bài toán lớn thành các bài toán nhỏ hơn là một trong các kỹ thuật, tư duy khoa học cơ bản, cốt lõi.

Trên thực tế khi gặp một công việc lớn, con người luôn tìm cách chia nhỏ công việc đó ra để làm từng bước. Khi làm xong các bài toán nhỏ thì cũng đồng thời giải quyết được bài toán lớn.

- Khi được giao quét 1 ngôi nhà lớn, em sẽ chia thành những phòng nhỏ và quét dọn từng phòng một.

- Nếu phải bê 1 chồng sách lớn em sẽ khuân và bê chuyển từng cuốn sách hoặc từng vài ba cuốn sách cho nhẹ.

- Khi được giao nhiều bài tập về nhà, em thường chia ra mỗi ngày làm 1 ít bài tập để cuối tuần khỏi dồn lại thành nhiều.

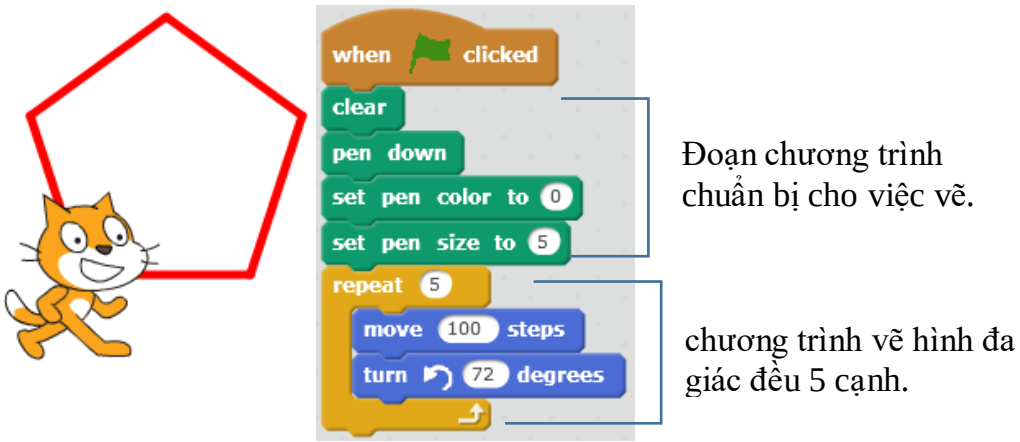
Em có thể chỉ ra rất nhiều ví dụ thực tế nữa.

Trong bài toán tin học, việc chia 1 công việc lớn thành nhiều việc nhỏ hơn để làm từng việc được gọi chung là "chia để trị". Đây là một kỹ thuật hay dùng nhất trong các bài toán tin học, nhất là lập trình máy tính.

Nội dung bài học

1. Có thể rút gọn chương trình bằng cách nào?

Chúng ta hãy bắt đầu hoạt động này bằng chương trình vẽ 1 đa giác đều đã biết.

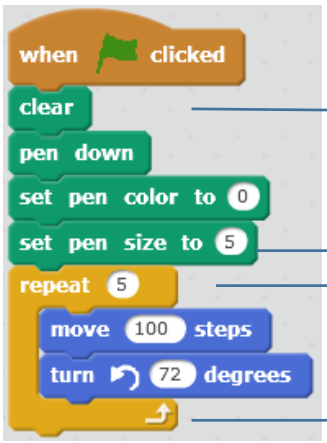


Đoạn chương trình, như em thấy, có thể chia làm 2 phần với chức năng tách biệt rõ rệt.

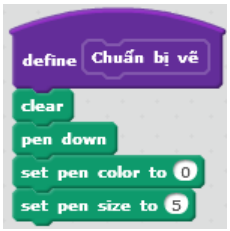
- Phần 1 bao gồm các lệnh thiết lập ban đầu cho việc vẽ.

- Phần 2 là các lệnh vẽ hình đa giác đều.

Chương trình trên bây giờ có thể tách làm 2 phần, phần 1 có tên **"Chuẩn bị vẽ"**, phần 2 có tên **"Vẽ đa giác đều 5 cạnh"**.



Thiết lập nhóm lệnh có tên **Chuẩn bị vẽ**



Thiết lập nhóm lệnh có tên **Vẽ đa giác đều 5 cạnh**

Khi đó chương trình gốc của chúng ta bây giờ rút gọn lại chỉ còn 2 "lệnh".

Em có nhận xét gì về hoạt động trên?

Các gợi ý:

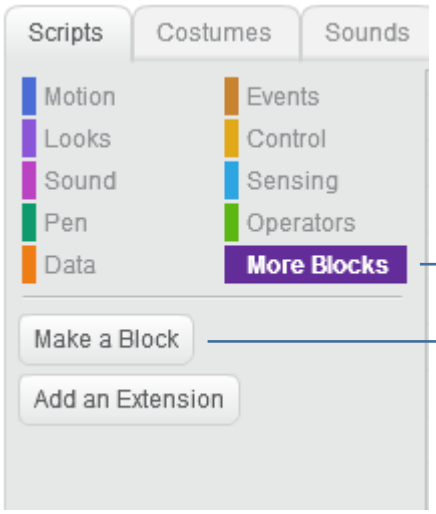
- Nhóm các lệnh vào thành 1 lệnh mới?

- Việc phân nhóm thế này có lợi ích gì?

2. Thiết lập và sử dụng khái niệm thủ tục trong Scratch

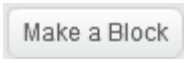
Trong hoạt động này, em sẽ thực hiện thao tác thiết lập các thủ tục cụ thể cho một chương trình cụ thể. Chúng ta hãy bắt đầu từ chương trình vẽ hình đa giác đều 5 cạnh trên.

Lệnh thiết lập thủ tục có trong nhóm lệnh **More Blocks** như hình sau.



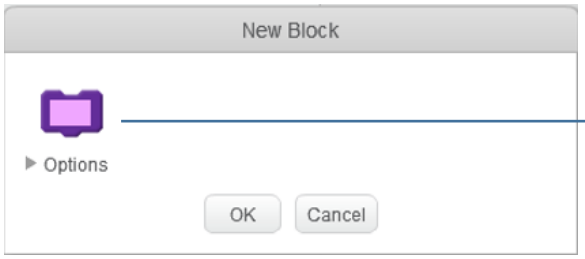
Nhóm lệnh **More Blocks** (thêm các lệnh khác), trong đó có lệnh làm việc với thủ tục.

Lệnh khởi tạo thủ tục.

Để khởi tạo 1 thủ tục mới, nháy nút **Make a Block** 

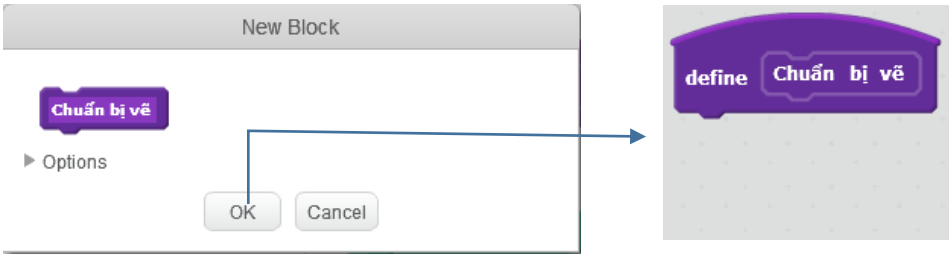
trong nhóm lệnh More Blocks.

Cửa sổ **New Block** xuất hiện.



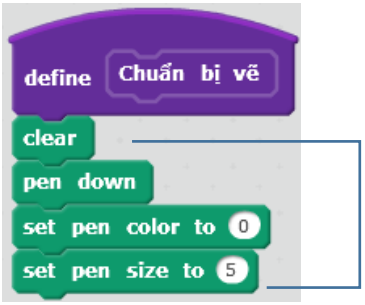
Nhập tên thủ tục tại vị trí này. Nhập xong nhấn nút OK.

Ví dụ em tạo ra thủ tục với tên "Chuẩn bị vẽ" như hình sau.



Nháy nút OK em sẽ thấy xuất hiện lệnh **define** trong cửa sổ lệnh.

Bây giờ em hãy kéo thả nhóm lệnh có chức năng là các lệnh chuẩn bị cho việc vẽ vào khung của lệnh **define** là lệnh định nghĩa thủ tục như sau. Khung bên dưới của lệnh **define** dùng để định nghĩa nội dung của thủ tục.



Kéo thả nhóm các lệnh này vào dưới và gắn với lệnh định nghĩa thủ tục (**define**).

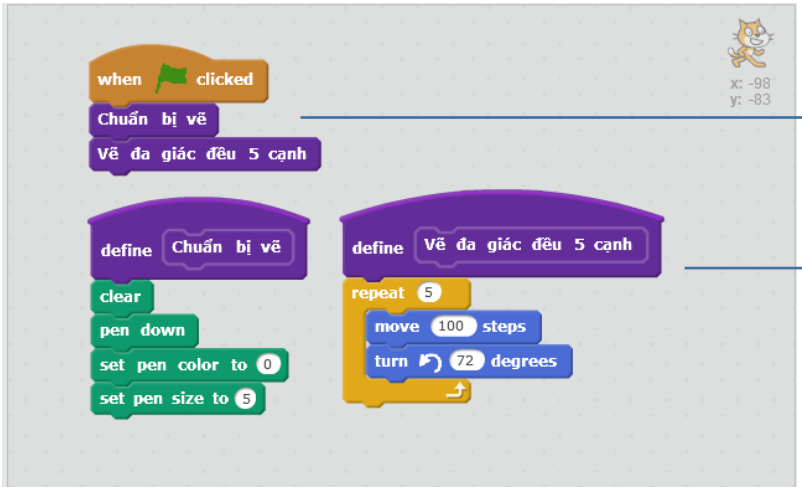
Tương tự em hãy tạo thêm 1 thủ tục nữa có tên **Vẽ đa giác đều 5 cạnh** và định nghĩa thủ tục này như sau:

Bây giờ, khi nhìn vào khu vực mẫu lệnh của nhóm More Blocks chúng ta sẽ thấy xuất hiện 2 lệnh tương ứng với 2 thủ tục vừa tạo ra.

2 lệnh tương ứng với 2 thủ tục vừa khởi tạo xuất hiện trong khung lệnh.

Các lệnh mới này sẽ được sử dụng hoàn toàn giống như các lệnh khác mà em đã biết, và có thể sử dụng nhiều lần.

Bây giờ trong khung chương trình chính, em chỉ cần sử dụng 2 lệnh trên để tạo thành đoạn chương trình chính thức của bài toàn. Cửa sổ lệnh của nhân vật có khuôn dạng như sau:



Đoạn chương trình chính chỉ còn 2 lệnh.

2 thủ tục được định nghĩa cùng trên 1 màn hình lệnh.

Theo em sử dụng thủ tục mang lại những lợi ích nào? Chọn các phương án em thấy đúng, giải thích vì sao.

A. Nhìn chương trình thấy dễ hiểu hơn, đơn giản hơn.

B. Chương trình chính được viết ngắn gọn hơn.

C. Sử dụng thủ tục sẽ làm cho số lệnh được viết ra ít hơn, ngắn gọn hơn.

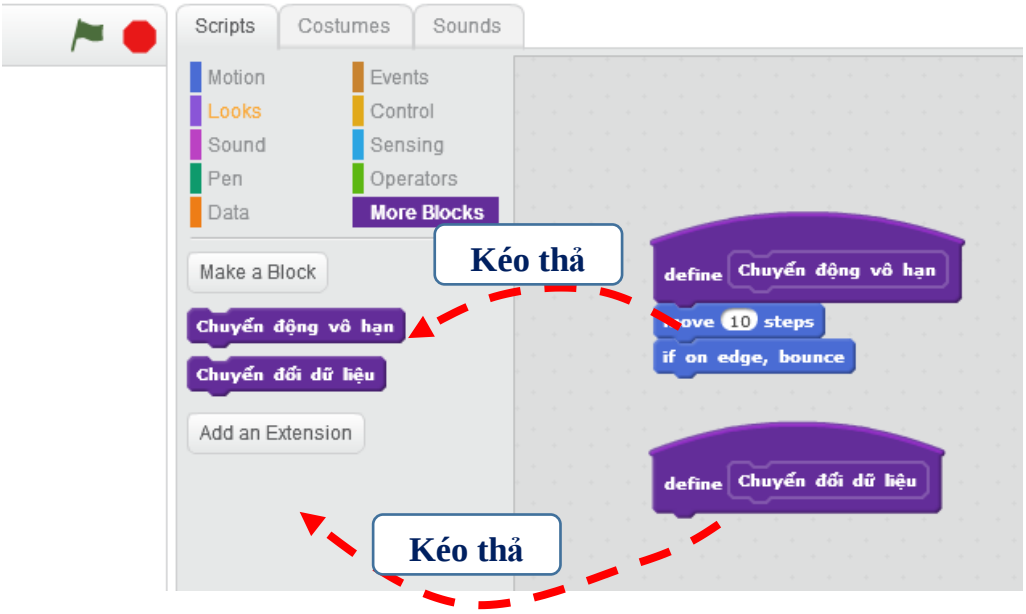
D. Mỗi thủ tục có thể hiểu là 1 bài toán con.

E. Làm cho chương trình trở nên dễ hiểu, có cấu trúc hơn.

3. Các thao tác khác với thủ tục

1. Xóa thủ tục

Muốn xóa 1 thủ tục chúng ta kéo thả dòng lệnh **define** thủ tục nào vào khung điều khiển lệnh ở giữa màn hình. Chú ý trước khi xóa thủ tục cần hủy hết các dòng lệnh của thủ tục này trong cửa sổ lệnh.



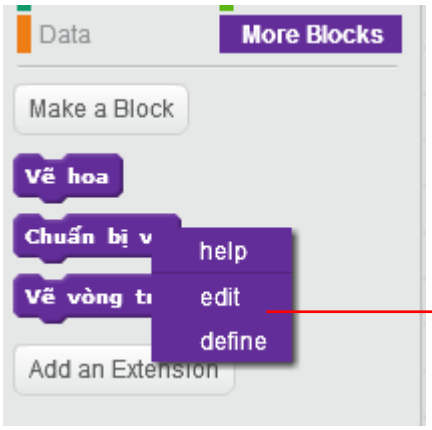
Chú ý:

- Nếu chỉ kéo thả 1 dòng lệnh define thủ tục thì sẽ xóa thủ tục này.

- Nếu kéo thả thủ tục và các lệnh gắn kèm theo thì sẽ xóa cả thủ tục và các lệnh này.

2. Đổi tên thủ tục

Muốn đổi tên 1 thủ tục cần thực hiện lệnh: nháy chuột phải lên dòng chứa thủ tục đó tại khung điều khiển lệnh, sau đó chọn chức năng **edit**.



Chức năng đổi tên thủ tục đã xác định trước đó.

4. Một số ví dụ đơn giản về thủ tục

Trong hoạt động này chúng ta sẽ cùng thực hiện các bài toán thiết kế các thủ tục cụ thể.

1. Thủ tục vẽ đa giác đều n cạnh với độ dài cạnh d, các số n, d cho trước

Thủ tục này đã được mô tả trong hoạt động trên. Chúng ta sử dụng lệnh **Make a Block** để tạo ra 1 thủ tục có tên **Vẽ đa giác đều**, sau đó kéo thả các lệnh tương ứng vào khung của thủ tục này như sau.

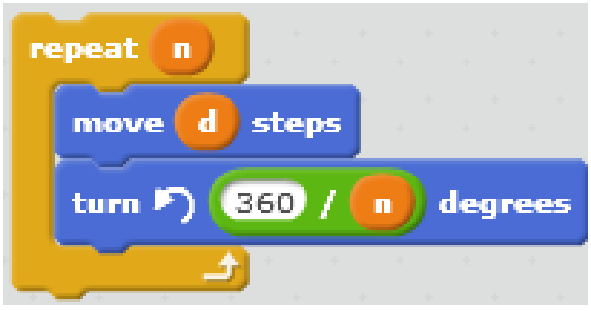
Nội dung của thủ tục **Vẽ đa giác đều**. Các biến nhớ **n** và **d** được khởi tạo từ trước.

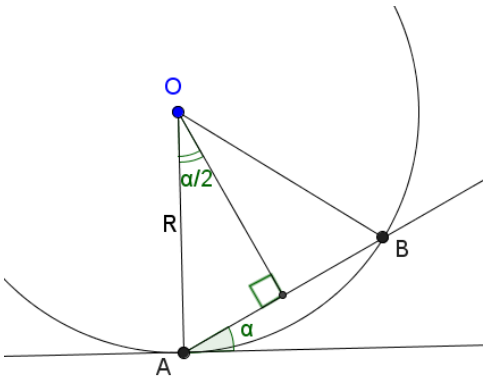
Sau đây là minh họa cho việc chạy thủ tục này. Trước khi chạy các biến nhớ **n** và **d** cần gán trước các giá trị.

2. Thủ tục vẽ vòng tròn biết tâm và bán kính

Thủ tục này yêu cầu có 3 tham số là (**Tx, Ty**) - tọa độ tâm của vòng tròn và R - bán kính vòng tròn cần vẽ.

Để phân tích thủ tục này chúng ta xuất phát từ thủ tục vẽ đa giác đều trong hoạt động trước



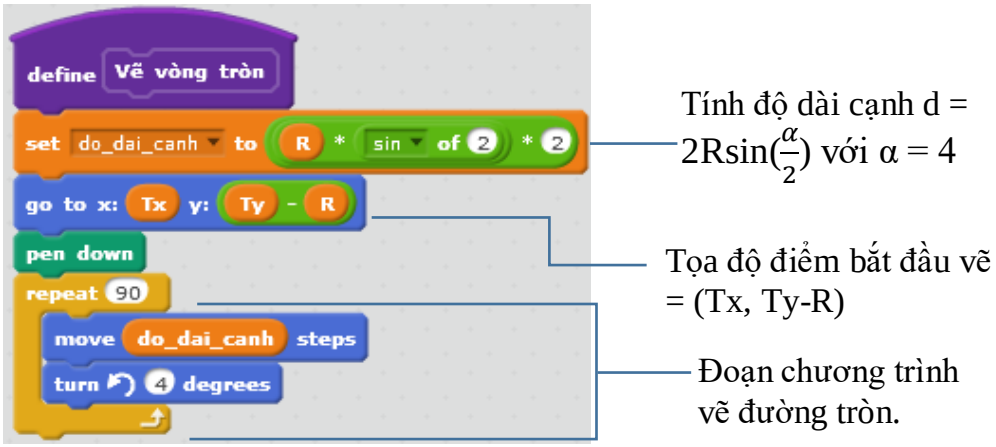


Cạnh AB được tính theo công thức:

$AB = 2R\sin(\frac{\alpha}{2})$

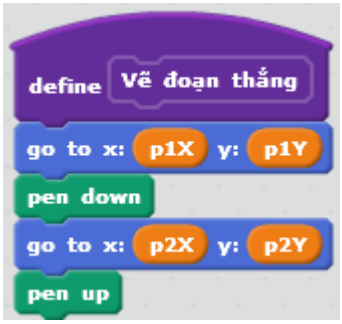
Để tạo được hình tròn, chúng ta chỉ cần tăng số **n** trong thủ tục trên lên 1 số khá lớn, ví dụ 90, 180, 360. Tuy nhiên độ dài cạnh **d** cần tính lại chính xác. Cạnh d có thể

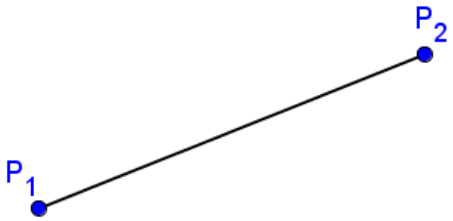
tính qua bán kính R theo công thức đã ghi ở trên: d = 2Rsin($\frac{\alpha}{2}$). Ví dụ với n = 90, khi đó α = 360/90 = 4. Do đó theo công thức trên ta sẽ có **d = 2Rsin(2)**.



3. Thủ tục vẽ đường thẳng đi qua 2 điểm A và B cho trước

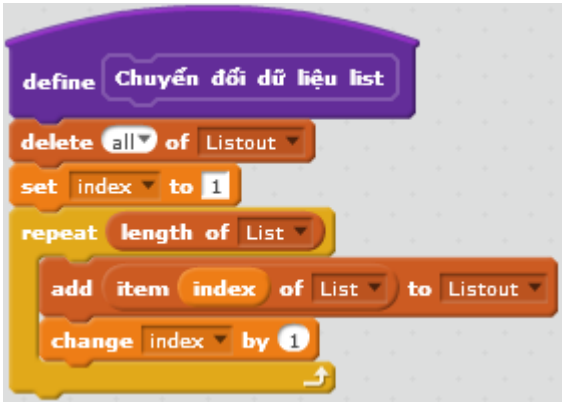
Giả sử điểm A có tọa độ (P_1X, P_1Y), điểm B có tọa độ (P_2X, P_2Y), khi đó thủ tục vẽ đoạn thẳng nối 2 điểm này như sau:





3. Thủ tục gán dữ liệu từ 1 biến nhớ danh sách này sang biến nhớ danh sách khác

Giả sử cho trước 1 biến nhớ kiểu danh sách **List**, cần xây dựng thủ tục chuyển dữ liệu từ **List** sang **Listout**.



Thuật toán của thủ tục này đơn giản như sau:

- Xóa nội dung của dãy Listout.
- Chuyển đổi lần lượt các phần tử của dãy **List** sang **Listout**.

4. Thủ tục sinh ngẫu nhiên 1 hoán vị các số 1, 2, 3, ..., n

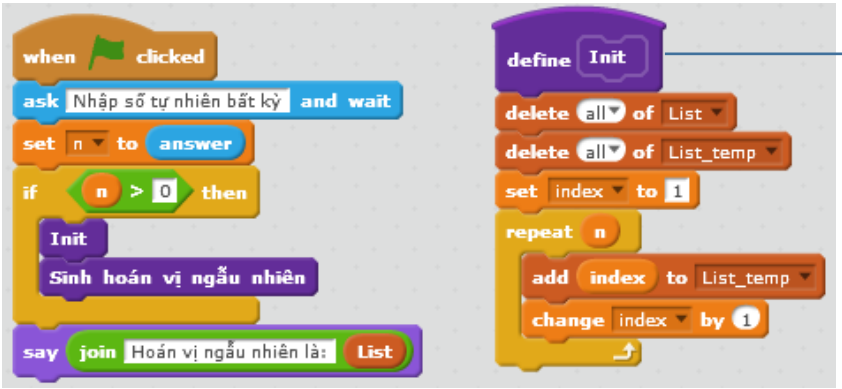
Cho trước số tự nhiên **n**, thủ tục cần xây dựng sẽ sinh ngẫu nhiên 1 hoán vị của dãy số 1, 2, 3, …, n và kết quả đưa vào biến nhớ **List**.

Để thực hiện thủ tục này, chúng ta cần tạo ra thêm 1 dãy phụ nữa, ví dụ List_temp. Đoạn chương trình của thủ tục như sau.

Dãy các số từ 1 đến n đã được đưa vào dãy **List_temp** từ trước đó.

Thuật toán của thủ tục: chọn ngẫu nhiên 1 số từ **List_temp**, đưa số này vào **List**, sau đó xóa số này từ **List_temp** và lặp lại quá trình trên cho đến khi dãy **List_temp** trở nên rỗng.

Dưới đây là đoạn chương trình hoàn chỉnh để sử dụng thủ tục trên.



Các lệnh chuẩn bị ban đầu được đưa vào 1 thủ tục **Init**, khởi tạo 2 dãy **List**, **List_temp** và đưa dãy 1..n vào **List_temp**.

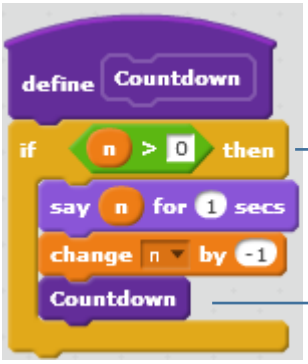
Đoạn chương trình chính.

5. Thủ tục gọi thủ tục khác và gọi chính nó

Như chúng ta đã thấy, thủ tục sau khi được khởi tạo sẽ hiện ra trong khung các mẫu lệnh và được sử dụng như một lệnh bình thường. Do vậy hoàn toàn có thể xảy ra là trong nội dung của 1 thủ tục này có lời gọi đến thủ tục khác, hoặc một thủ tục sẽ có lời gọi đến chính nó. Trong hoạt động này chúng ta sẽ làm quen với các hiện tượng này.

a. Thủ tục có thể gọi chính nó

Khi trong thủ tục có lời gọi đến chính nó, điểm cần đặc biệt chú ý nhất là phải có điều kiện đảm bảo không để thủ tục chạy vô hạn. Trong ví dụ sau, thủ tục Countdown có nhiệm vụ hiển thị giá trị biến nhớ n và đếm ngược cho đến 0. Quá trình này cần phải kết thúc với điều kiện n > 0.

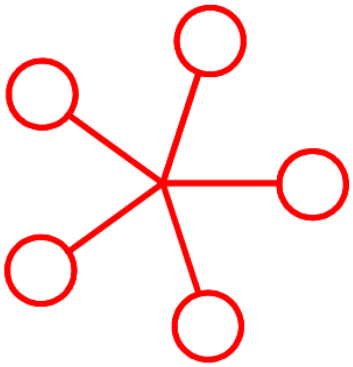


Điều kiện để dừng thủ tục.

Lời gọi chính mình.

b. Thủ tục gọi thủ tục

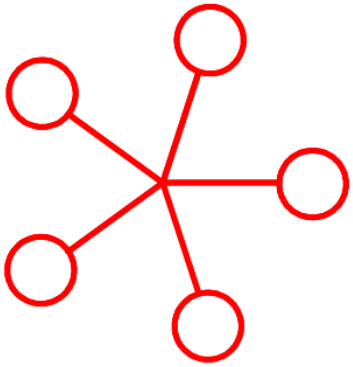
Chúng ta quay lại 1 ví dụ đã biết trong các bài trước, vẽ bông hoa 5 cánh như hình sau.

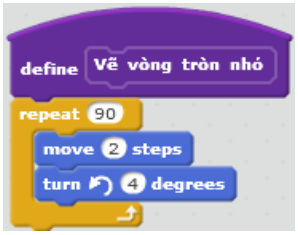


Chúng ta nhớ lại qui trình vẽ bông hoa này như sau:

- Ở vòng lặp ngoài sẽ lặp 5 lần và vẽ 5 đoạn thẳng từ tâm đến vị trí tiếp cận vòng tròn.
- Ở vòng lặp trong là vẽ vòng tròn nhỏ.

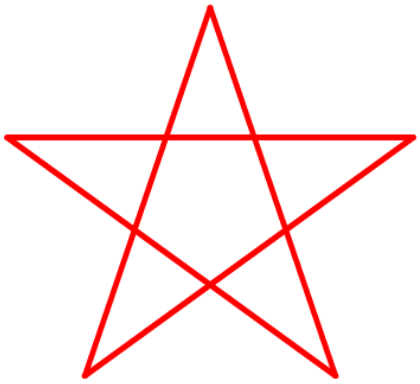
Dựa trên chương trình gốc này, chúng ta sẽ tạo 2 thủ tục: **Vẽ hoa** ở vòng lặp ngoài và **Vẽ vòng tròn nhỏ** ở vòng lặp trong. Thủ tục **Vẽ hoa** sẽ gọi **Vẽ vòng tròn nhỏ** bên trong các lệnh của mình. Nội dung cụ thể 2 thủ tục này trong hình dưới đây.

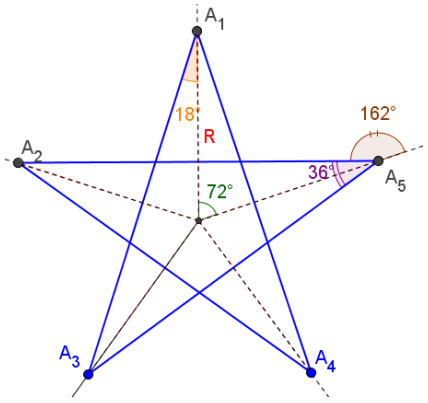




6. Chương trình vẽ ngôi sao 5 cánh

Trong hoạt động này chúng ta sẽ thiết lập chương trình vẽ ngôi sao 5 cánh như hình dưới đây với bán kính **R** cho trước. Để thuận tiện cho việc phân tích bài toán, chúng ta cần vẽ thêm hình 1 ngôi sao 5 cánh ở bên với các tham số mô tả kỹ hơn cấu trúc của ngôi sao này.





Phân tích bài toán.

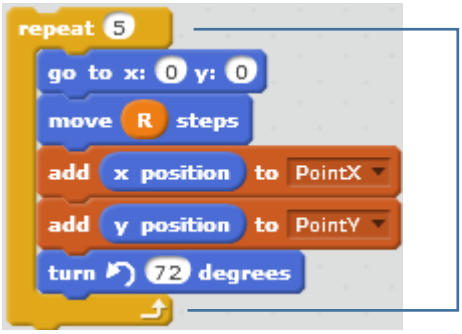
Quan sát hình trên bên phái cho cho chúng ta gợi ý cách thực hiện yêu cầu bài toán. Thiết lập 2 bảng, biến nhớ dạng List để lưu trữ tọa độ các điểm A_1, A_2, A_3, A_4, A_5. Các biến mảng này sẽ ký hiệu là **PointX** và **PointY**. Bảng **PointX** sẽ lưu trữ các tọa độ X, mảng **PointY** sẽ lưu trữ các tọa độ y.

Như vậy các bước để giải quyết bài toán này như sau:

1. Thiết lập tọa độ các điểm A_1, A_2, A_3, A_4, A_5 và đưa vào 2 mảng **PointX, PointY**.

2. Vẽ hình ngôi sao bằng cách nối vẽ các điểm A_1-A_3, A_2-A_4, A_3-A_5, A_4-A_1, A_5-A_2.

Đoạn chương trình sau mô tả cách thiết lập tọa độ dãy điểm A_1, A_2, A_3, A_4, A_5. Các tọa độ này lần lượt được bổ sung vào 2 danh sách **PointX, PointY**.

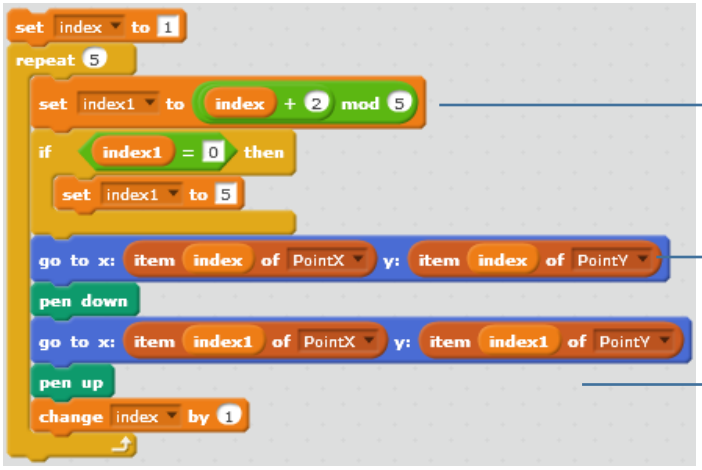


Vòng lặp 5 lần, mỗi lần thực hiện:

- Bắt đầu từ vị trí (0, 0), dịch chuyển 1 đoạn bằng R, nạp thông tin tọa độ X, Y vào 2 bảng PointX, PointY.

- Xoay 72 độ.

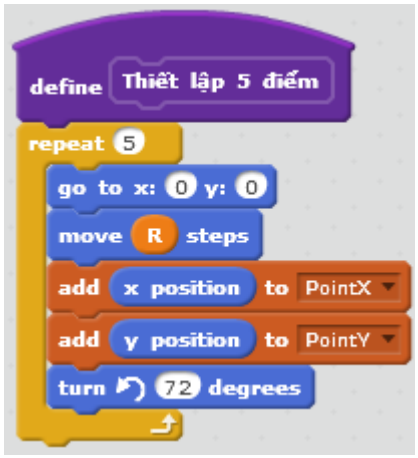
Đoạn chương trình sau thực hiện công việc vẽ các nét nối các cặp điểm A_1-A_3, A_2-A_4, A_3-A_5, A_4-A_1, A_5-A_2. Việc này cũng được thực hiện trong 1 vòng lặp 5 lần.



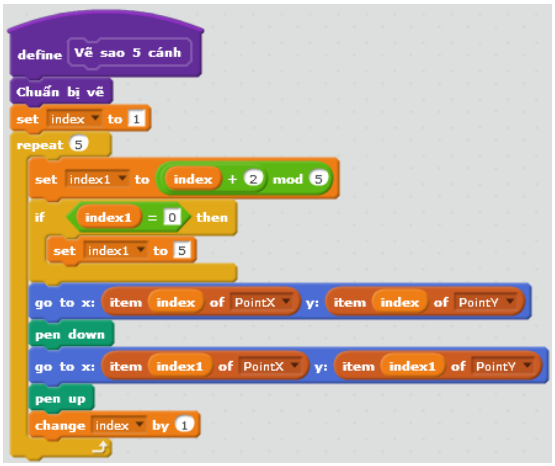
Xác định các chỉ số **index** và **index1**

Vẽ đoạn thẳng nối điểm từ điểm có chỉ số **index** đến điểm có chỉ số **index1**

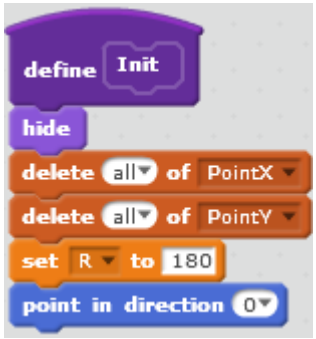
Toàn bộ các thủ tục của chương trình như sau:



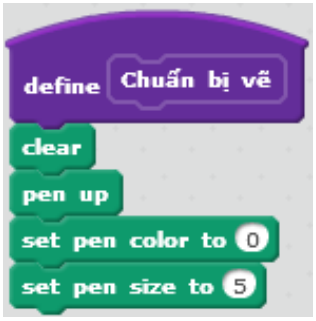
Thủ tục **Thiết lập 5 điểm** là đỉnh ngôi sao trên màn hình.



Thủ tục **Vẽ ngôi sao 5 cánh**. Thủ tục này gọi thủ tục **Chuẩn bị vẽ**.



Thủ tục **Init**.



Thủ tục **Chuẩn bị vẽ**

Chương trình chính bao gồm 3 lệnh tương ứng với 3 thủ tục.

Câu hỏi và bài tập

1. Cho trước dãy số **List**, hãy viết 1 thủ tục sinh ngẫu nhiên 1 hoán vị của dãy này, kết quả sẽ đưa vào mảng **Listout**.

2. Cho trước 1 xâu ký tự **Str**. Viết 1 thủ tục sinh ngẫu nhiên 1 hoán vị của xâu này, kết quả đưa vào xâu **Strout**.

3. Hoàn thiện chương trình Vẽ hoa.

4. Hãy thực hiện bài toán vẽ hình ngôi sao theo cách khác, ví dụ: tính độ dài cạnh ngôi sao (khoảng cách d = A_1A_3, thuật toán vẽ chính sẽ như sau: xuất phát từ vị trí A_1, thực hiện lặp 5 lần thao tác sau: xoay trái 36 độ, di chuyển theo độ dài d.

5. Cho trước 1 xâu ký tự **Str** và 1 số tự nhiên **m** (m < lenghth(Str)). Viết 1 thủ tục sinh ngẫu nhiên 1 xâu con độ dài **m** lấy từ Str và đã được hoán vị của xâu này, kết quả đưa vào xâu **Strout**.

6. Viết thủ tục vẽ lá cờ đỏ sao vàng:

- Chiều dọc = 2/3 chiều dài.

- Ngôi sao 5 cánh có chiều dài cánh = 1/3 chiều dài lá cờ.

7. Cho trước 1 dãy số nằm trong biến dãy List. Viết 1 thủ tục sắp xếp lại dãy số này theo thứ tự tăng dần.

8. Cho trước 1 dãy số nằm trong biến dãy List. Viết 1 thủ tục sắp xếp lại dãy số này theo cách sau: các số âm chuyển sang bên trái, các số dương chuyển sang bên phải, các số = 0 ở giữa.

9. Cho trước số tự nhiên n, viết thủ tục sinh và hiện trên màn hình n số hạng đầu tiên của dãy số Fibonaci. Nhớ lại dãy Fibonaci được định nghĩa như sau:

$F_1 = 1$, $F_2 = 1$, $F_3 = 2$, ……., $F_n = F_{n-1} + F_{n-2}$.

10. Cho trước vòng tròn bán kính R.

- Viết thủ tục tính chu vi đường tròn.

- Viết thủ tục tính diện tích đường tròn.

11. Viết chương trình nhập 1 số dương bất kỳ R, sau đó vẽ vòng tròn có tâm là trung điểm màn hình, bán kính R.

12. Viết thủ tục vẽ đường tròn biết tâm (Tx, Ty), bán kính R theo thuật toán sử dụng công thức tính tọa độ điểm trên vòng tròn.

Gợi ý: xem mục 6, bài học Xử lý số 2.

Tọa độ điểm A(x,y) trên vòng tròn được tính theo công thức sau:

x = Tx + R*cos(α).

y = Ty + R*.sin(α).

Mở rộng

Chương trình **Show Number**.

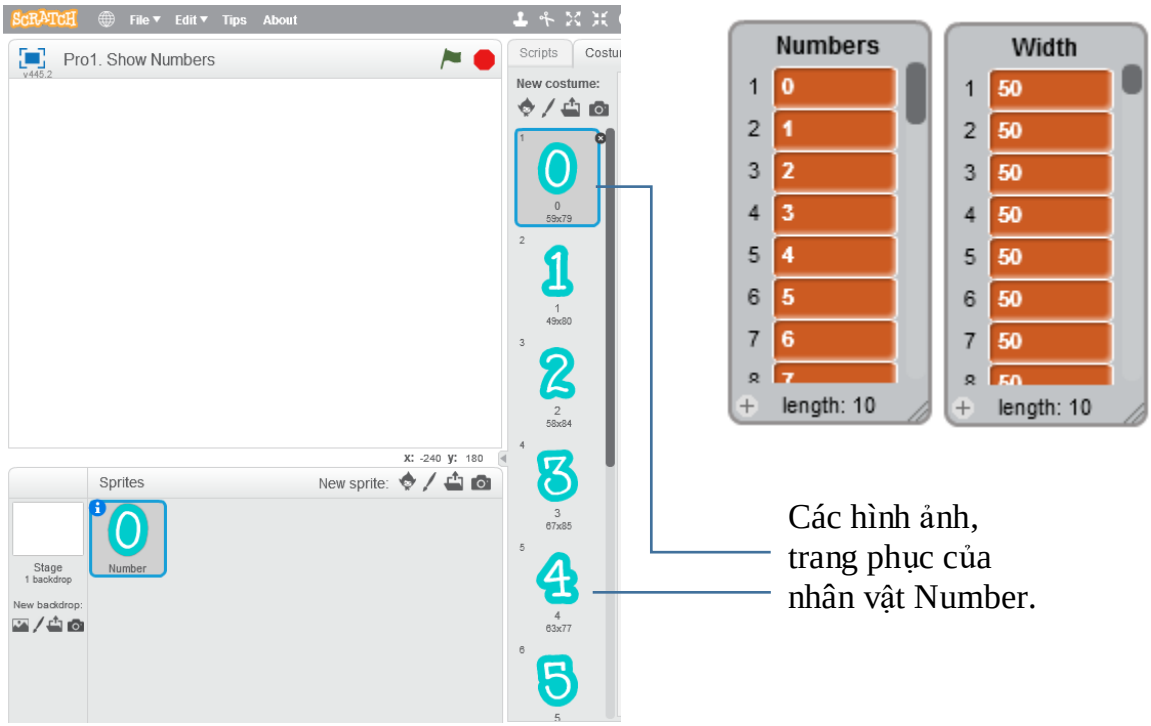
Viết chương trình thực hiện thể hiện số theo yêu cầu sau.

Show Numbers.sb2

Chương trình sẽ yêu cầu người dùng nhập 1 số tự nhiên từ bàn phím, sau đó thể hiện số này trên màn hình bằng các chữ số lớn, rõ ràng.

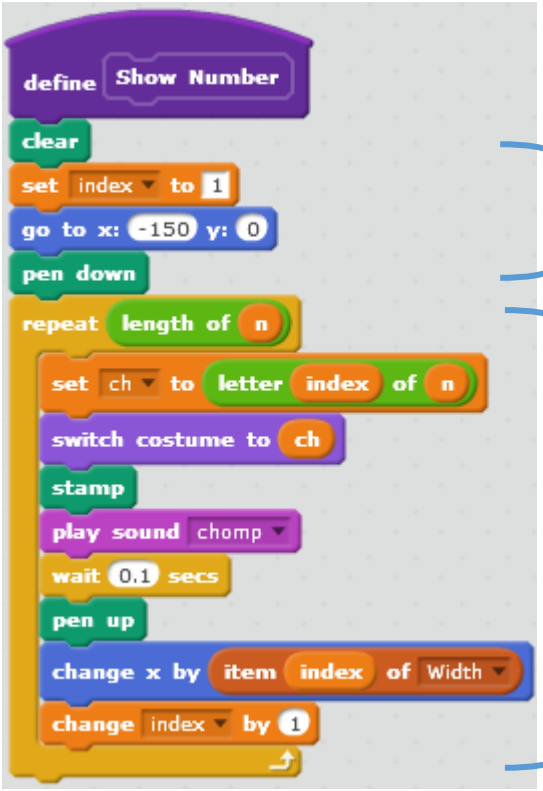
Ý nghĩa của chương trình này rất lớn và có nhiều ứng dụng trong các chương trình khác nhau, vì các giá trị số của biến nhớ được thể hiện trên màn hình rất nhỏ, khó quan sát.

Để thực hiện bài toán này chúng ta tạo ra 1 nhân vật có tên Number có đầy đủ toàn bộ 10 trang phục chính là các chữ số 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Các chữ số này và độ rộng của chúng trên màn hình được lưu trong 2 dãy **Numbers** và **Width** như hình dưới đây.



Các hình ảnh, trang phục của nhân vật Number.

Biến nhớ **n** dùng để lưu số do người dùng nhập. Thủ tục sau có nhiệm vụ thể hiện số n trên màn hình bằng nhân vật **Number**.



Thiết lập các giá trị ban đầu, vị trí bắt đầu viết số, xóa màn hình.

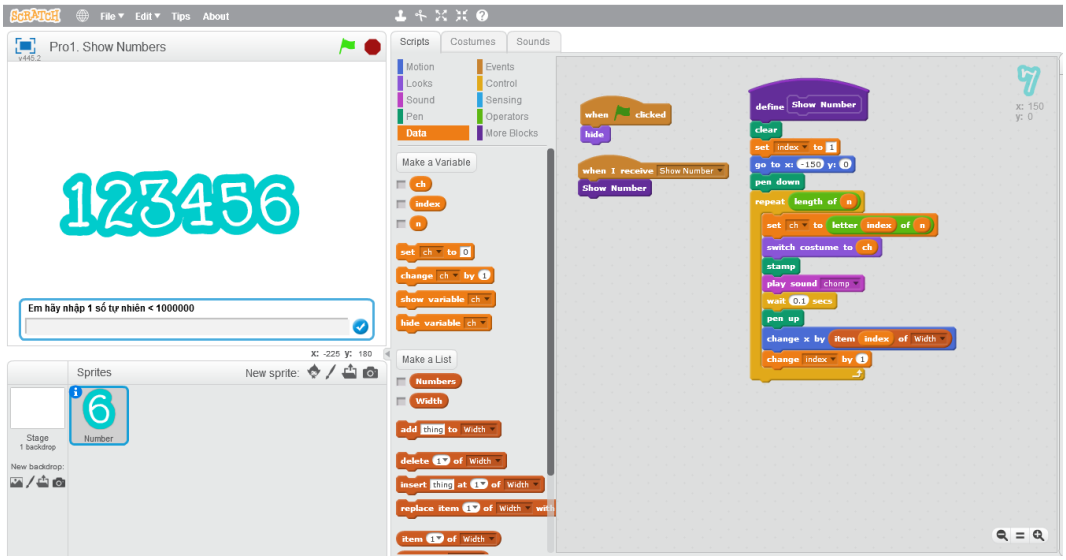
Lặp theo độ dài các chữ số của **n**.

Với mỗi chữ số, ví dụ **ch**, chuyển trang phục sang **ch**, thực hiện lệnh **stamp**, sau đó dịch chuyển tọa độ sang phải theo đúng độ rộng của ký tự **ch** vừa viết trên màn hình.

Quá trình này lặp lại cho đến khi viết xong các chữ số của **n**.

Yêu cầu nhập số n được tiến hành bởi sân khấu. Sau khi nhập xong dữ liệu, thông điệp **Show Number** được gửi đi và khi nhận thông điệp này, thủ tục **Show Number** sẽ được kích hoạt để thể hiện số đã nhập trên màn hình.

Show Numbers.sb2



Giao diện thể hiện của chương trình.

Bài 19. Thủ tục 2

Mục đích

Học xong bài này, bạn sẽ biết:

- Thiết lập thủ tục có tham số.

- Phân biệt biến nhớ riêng của thủ tục và biến nhớ của nhân vật.

- Các ứng dụng tiếp theo của thủ tục.

Bắt đầu

Trong bài học trước, em đã được làm quen với khái niệm thủ tục. Thủ tục sau khi được định nghĩa sẽ hiện ra trong khung điều khiển lệnh như 1 lệnh mẫu và em có thể sử dụng lệnh này như mọi lệnh khác. Ví dụ thủ tục **Vẽ đa giác đều** sau.

Chúng ta cùng nhau xem lại nội dung của thủ tục này, em có nhận xét gì khi quan sát nội dung này?

Em hãy quan sát các biến nhớ **n** và **d**.

Các biến nhớ **n** và **d** có màu cam sẫm là biến nhớ của nhân vật đang thực hiện thủ tục này.

Như vậy các biến nhớ **n**, **d** được tạo ra bởi nhân vật đang thực hiện thủ tục này và không phụ thuộc vào thủ tục. Do vậy khi gọi thủ tục này, các biến nhớ **n**, **d** cần được gán dữ liệu trước. Ví dụ:

Trong Scratch, em đã biết có rất nhiều lệnh có tham số, tức là chúng ta cho thể nhập trực tiếp dữ liệu trên dòng định nghĩa của lệnh này.

Ví dụ nếu như chúng ta có thủ tục vẽ 1 đa giác đều, trong đó tham số số cạnh và độ dài cạnh có thể nhập trực tiếp như lệnh sau thì việc áp dụng thủ tục này trên thực tế sẽ thuận tiện và đơn giản hơn rất nhiều.

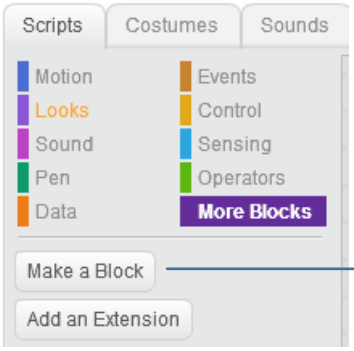
Trong bài học này chúng ta sẽ làm quen với loại thủ tục có tham số này.

Nội dung bài học

1. Thiết lập tham số cho thủ tục

Trong hoạt động này chúng ta sẽ cùng nhau thiết lập các thủ tục có tham số trong Scratch, bắt đầu từ bài toán vẽ đa giác đều trên.

(a) Để thiết lập thủ tục chúng ta chọn nhóm lệnh **More Blocks** và sau đó nháy lên nút **Make a Block**.

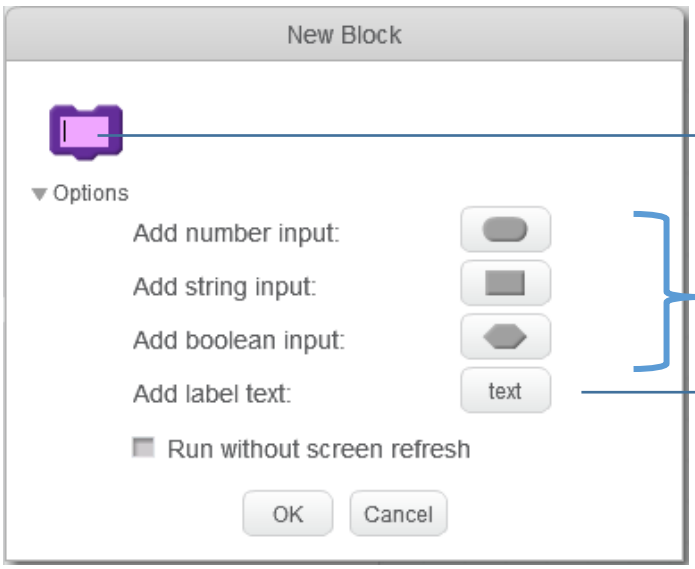


Nháy nút này để bắt đầu khởi tạo 1 thủ tục mới.

(b) Xuất hiện hộp hội thoại New Block như hình dưới đây. Nháy lên nút Options để mở rộng hộp hội thoại.

Nháy lên nút Options để mở rộng hộp hội thoại này đầy đủ sẵn sàng cho việc khởi tạo thủ tục có tham số.

(c) Hộp hội thoại tạo thủ tục mở rộng như sau.



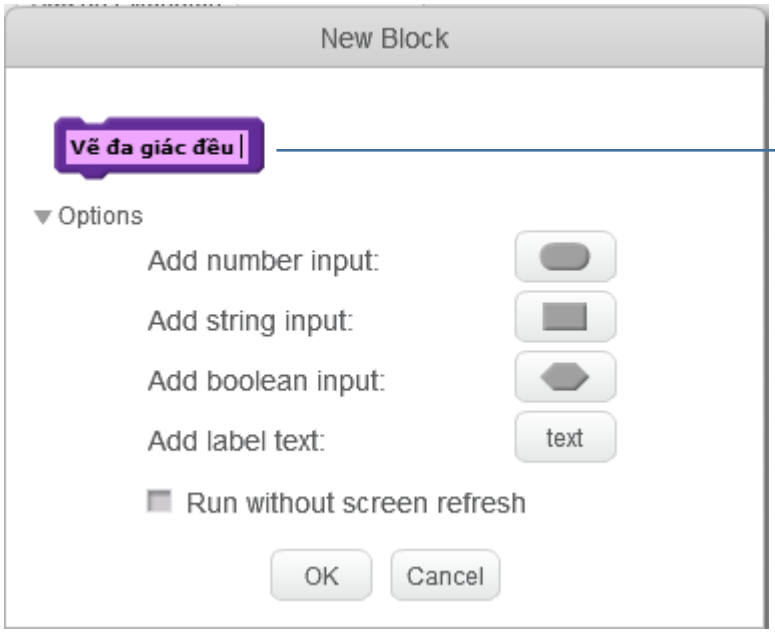
Nhập tên thủ tục và các tham số của thủ tục tại đây.

Các nút này dùng để đưa tham số vào thủ tục. Cho phép 3 loại tham số: số, xâu và logic.

Chèn giữa các tham biến là chữ được chèn bằng nút này.

Chú ý ý nghĩa các nút chèn tham số và chữ vào tên của thủ tục.

(d) Đầu tiên cần nhập tên của thủ tục trước. Ví dụ nhập tên **"Vẽ đa giác đều "**.

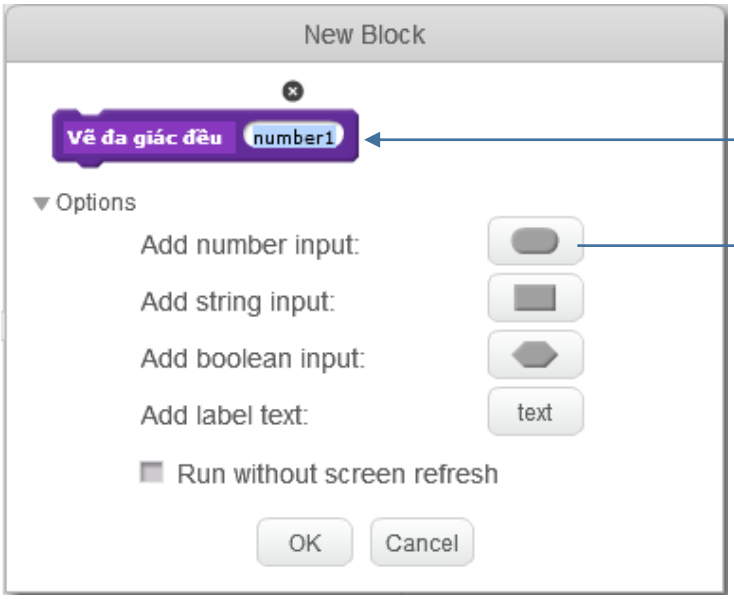


Nhập tên của thủ tục tại đây.

(e) Sau khi nhập tên chính của thủ tục thì bắt đầu có thể nhập tham số.

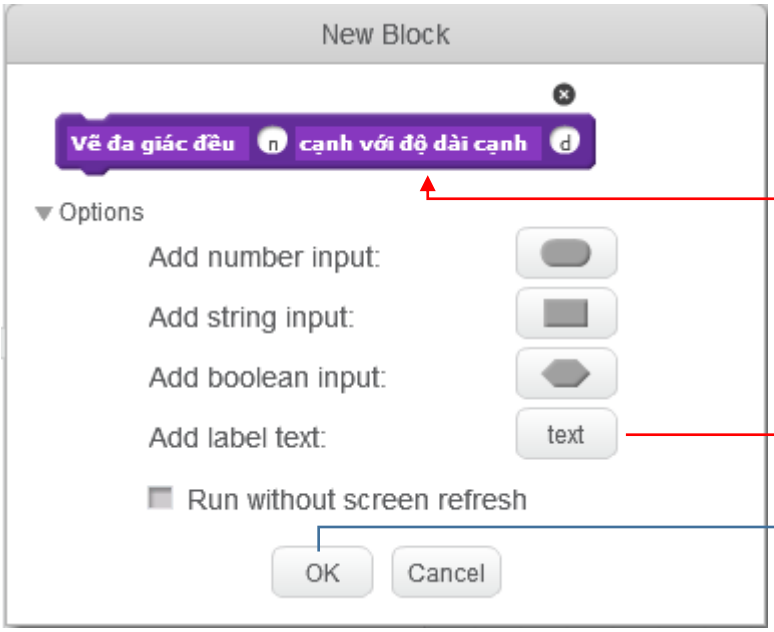
Trong ví dụ dưới đây, sau khi nhập tên thủ tục là Vẽ đa giác đều, em sẽ nháy nút  để tạo 1 tham số dạng số (number) đầu tiên.

Xuất hiện khung tham số với tên **"number1"**. Em hãy thay đổi thành **"n"**.



Nháy nút này để tạo ra tham số đầu tiên dạng số của thủ tục. Tên của tham số sẽ mặc định là **"number1"**, em hãy đổi thành **"n"**.

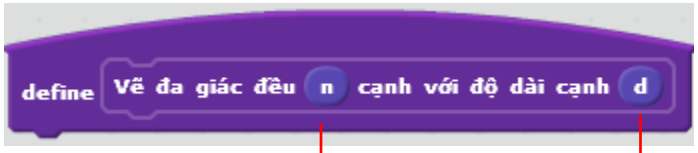
(f) Tiếp theo, em hãy chèn 1 đoạn chữ vào thủ tục ("**cạnh với độ dài cạnh**"), sau cùng bổ sung thêm 1 tham biến dạng số nữa, đặt tên là "**d**" như hình sau.



Nháy nút này để tạo ra thêm dòng chữ "**cạnh với độ dài cạnh**" trong tên

Nháy OK để kết thúc khởi tạo tên của thủ tục.

(g) Sau khi tạo xong tên của thủ tục, em sẽ thấy xuất hiện lệnh **define**, biểu tượng sau trên cửa sổ lệnh của nhân vật. Em hãy chú ý quan sát các tham biến đã được định nghĩa trong thủ tục.



Các tham biến trong dòng định nghĩa thủ tục có màu xanh.

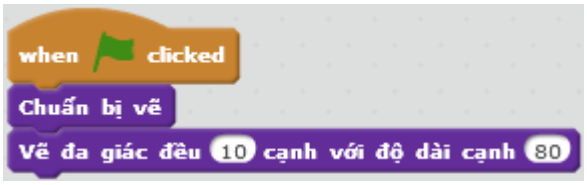
Các tham biến được định nghĩa trong thủ tục sẽ có màu xanh. Trong quá trình viết các lệnh mô tả thủ tục chúng ta có thể sử dụng các tham biến này như các biến nhớ bình thường, ngoại trừ các lệnh thao tác trực tiếp thay đổi giá trị của các biến nhớ này.

(h) Bây giờ chúng ta điền nội dung của thủ tục bằng các lệnh đã biết. Chú ý sử dụng các tham biến của thủ tục như các biến nhớ thông thường.

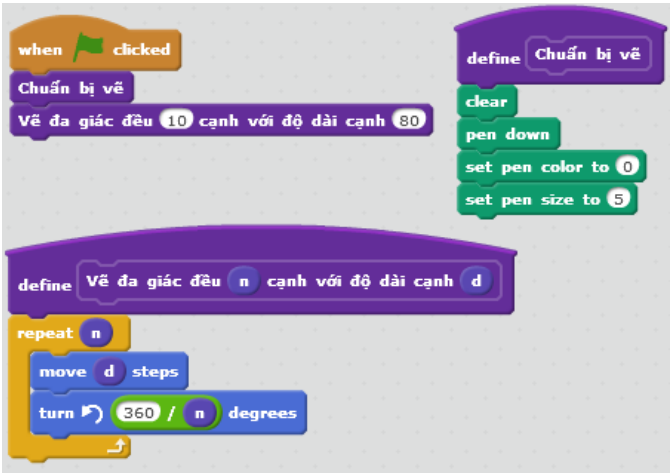
Trong nội dung của thủ tục cần sử dụng các tham biến như các biến nhớ bình

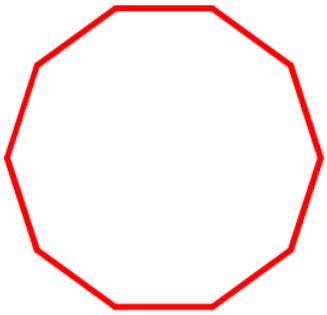
(i) Sau khi thủ tục được định nghĩa xong, tên của thủ tục sẽ xuất hiện tại khung mẫu lệnh như 1 lệnh có tham số bình thường của Scratch.

(j) Bây giờ chương trình chính của bài này sẽ được viết đơn giản bằng 2 lệnh như sau.



Toàn bộ cửa sổ lệnh và kết quả thực hiện chương trình như sau.

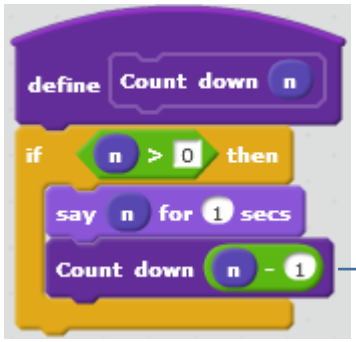




2. Thủ tục đếm ngược Count down

Thủ tục đếm ngược **Count down** từ bài trước bây giờ có thể viết dưới dạng có tham số như sau.

Chú ý lời gọi chính nó sẽ có tham số thay đổi.



Lời gọi chính thủ tục gốc **Count down** với giá trị tham số n - 1.

Như vậy chúng ta thấy các thủ tục có tham số vẫn có thể gọi chính nó, các thủ tục có tính chất như vậy được gọi là đệ qui.

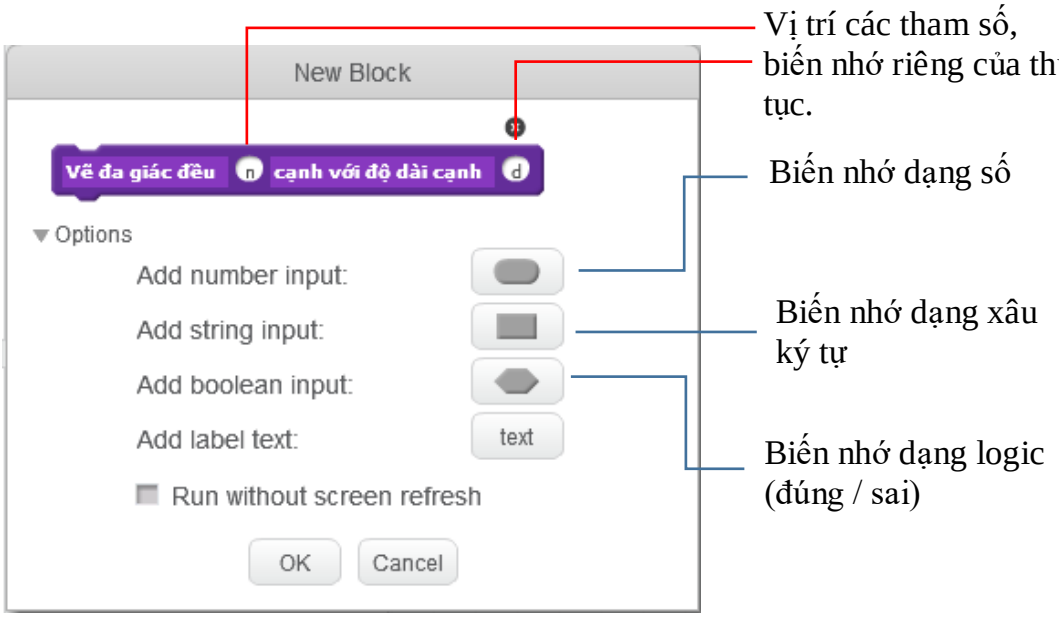
Một trong các vấn đề quan trọng nhất của các thủ tục có lời gọi đến chính nó (thủ tục đệ qui) là vấn đề kiểm soát không để chương trình bị rơi vào trạng thái "quẩn" tức là chạy vô định không bao giờ dừng. Các thủ tục này cần có lệnh kiểm soát khi nào thì dừng chương trình.

Trong ví dụ trên, chính lệnh If sẽ đảm bảo thủ tục sẽ dừng khi giá trị tham số n nhỏ hơn hoặc = 0.

Trong phần cuối của bài học này chúng ta sẽ được làm quen nhiều hơn với các thủ tục đệ qui này.

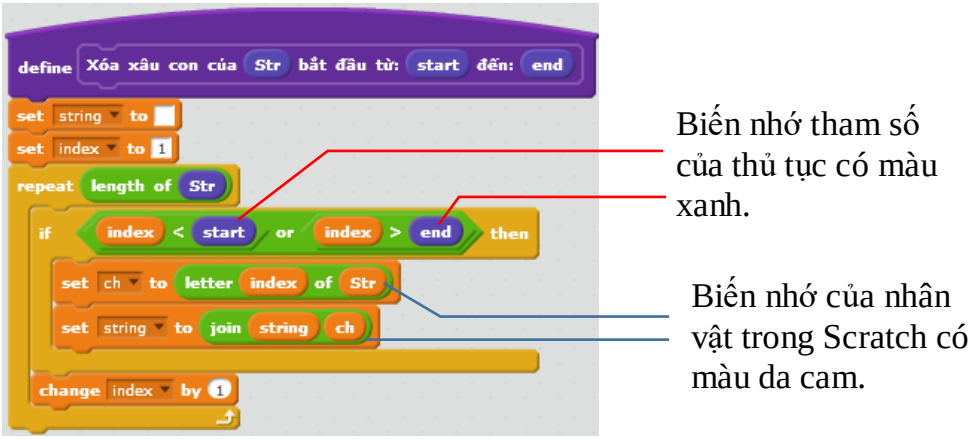
3. Biến nhớ của thủ tục

Các tham biến được định nghĩa trong thủ tục chính là các biến nhớ riêng của thủ tục này. Chúng ta có thể tạo ra nhiều biến nhớ như vậy trong quá trình định nghĩa thủ tục mới. Trong hoạt động này chúng ta cùng phân tích sâu hơn về các tham biến hay biến nhớ riêng này của thủ tục trong Scratch.



Các tham biến được định nghĩa trong thủ tục có thể hiểu như các biến nhớ riêng của thủ tục này. Vậy các biến nhớ này có đặc điểm gì khác biệt với khái niệm biến nhớ thông thường mà ta đã biết.

(1) Tất cả các tham biến, hay biến nhớ của thủ tục đều có màu xanh để phân biệt với biến nhớ của nhân vật trong Scratch có màu da cam. Do vậy 2 hệ thống biến nhớ này là độc lập hoàn toàn với nhau.



(2) Các giá trị được truyền vào tham số của thủ tục có thể là giá trị số cụ thể, có thể là biến nhớ.

Trong ví dụ sau cả 3 tham số của thủ tục đều được truyền giá trị thông qua các biến nhớ.

Trong ví dụ trên, giá trị của biến nhớ nhân vật **Str** được truyền vào thủ tục (tại thời điểm gọi thủ tục này) thông qua tham biến **Str**. Chú ý biến **Str** màu da cam khác biệt hoàn toàn với tham biến **Str** màu xanh.

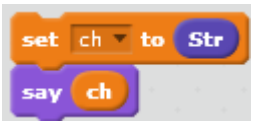
Ví dụ: lệnh sau đây nếu được thực hiện trong thủ tục trên sẽ gán giá trị của tham biến **Str** của thủ tục cho biến nhớ **Str**.

Còn trong ví dụ sau, sau khi gán trực tiếp biến nhớ Str thành "Hà Nội" thì giá trị tham biến Str của thủ tục vẫn không thay đổi.

(3) Các tham biến, biến nhớ riêng của thủ tục chỉ có ý nghĩa bên trong thủ tục này.

Các biến nhớ màu xanh chỉ có tác dụng nếu việc sử dụng chúng được đặt trong các lệnh bên trong thủ tục này.

Ví dụ: 2 lệnh sau nếu đặt ngay trong chương trình chính sẽ không có ý nghĩa vì biến nhớ màu xanh Str không có tác dụng.



(4) Không thể thay đổi giá trị của các tham biến hay biến nhớ của thủ tục.

Các tham biến hay biến nhớ bên trong thủ tục không thể thực hiện được các lệnh thay đổi giá trị như xóa, gán hoặc thay đổi thông tin. Khi thực hiện lời gọi thủ tục, các tham biến này được truyền 1 giá trị từ bên ngoài và giữ nguyên giá trị đó trong suốt thời gian thủ tục có hiệu lực.

4. Một số thủ tục với xâu ký tự

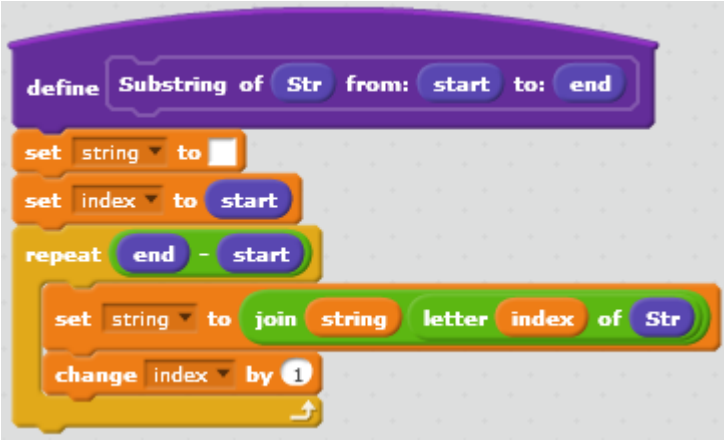
4.1. Thủ tục lấy ra 1 xâu con từ xâu gốc

Thủ tục này có các tham biến đầu vào:

Str: xâu gốc ban đầu.

start, end: vị trí các ký tự bắt đầu và kết thúc cần lấy ra xâu con từ xâu gốc **Str**.

Kết quả xâu con được gán cho biến nhớ **string**.



4.2. Thủ tục chèn 1 xâu vào một xâu khác

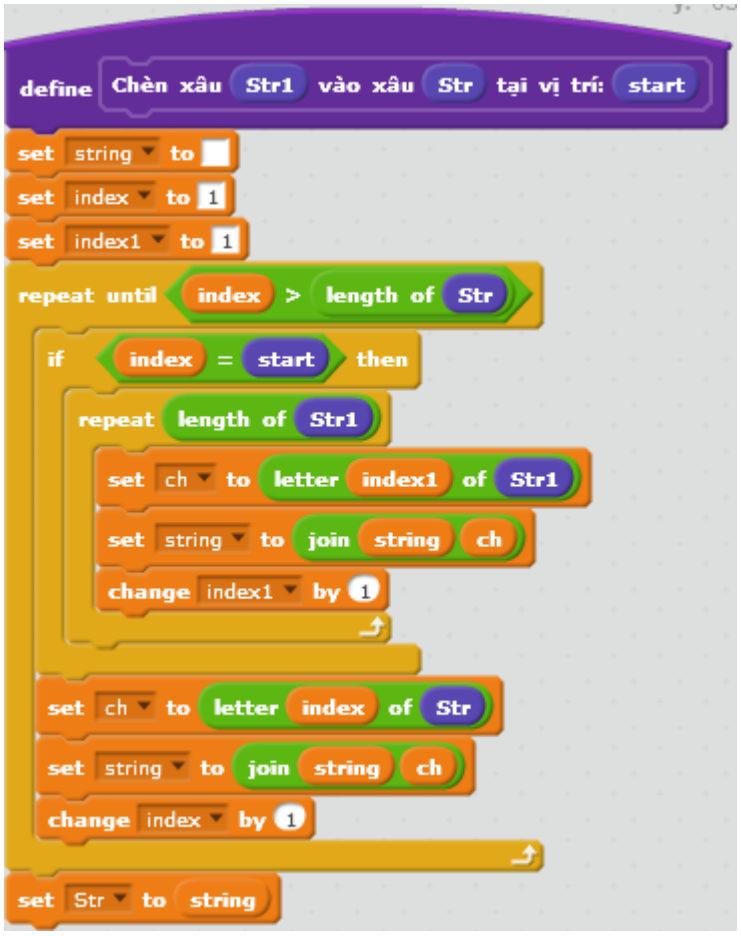
Các tham số của thủ tục:

Str1: xâu, nội dung cần chèn.

Str: xâu ký tự gốc ban đầu.

start: vị trí cần chèn trong xâu gốc **Str**.

Kết quả xâu gốc **Str** sau khi đã chèn **Str1** vào vị trí **start** sẽ được gán cho biến nhớ **string**.



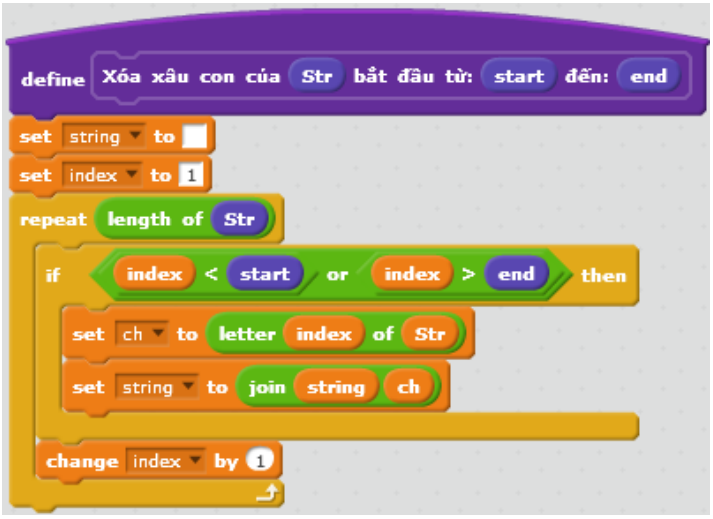
4.3. Thủ tục xóa 1 xâu con trong 1 xâu ký tự

Các tham biến của thủ tục này:

Str: xâu ký tự gốc.

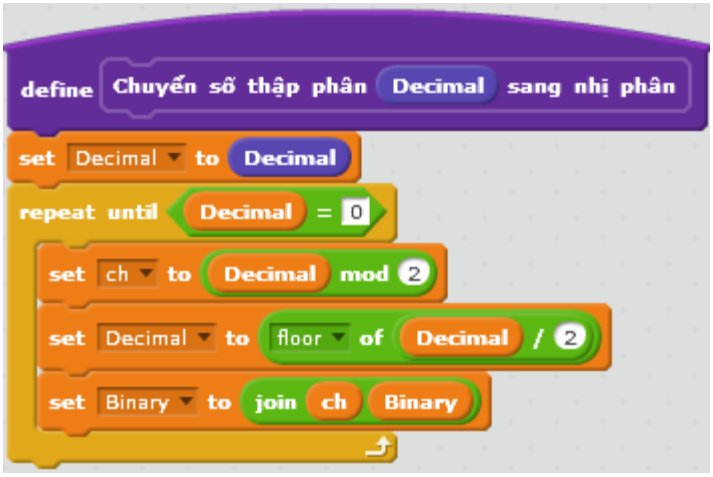
start, end: vị trí đầu và cuối trong xâu **Str** cần xóa.

Kết quả của việc xóa các ký tự trong xâu **Str** sẽ gán vào xâu **string**.



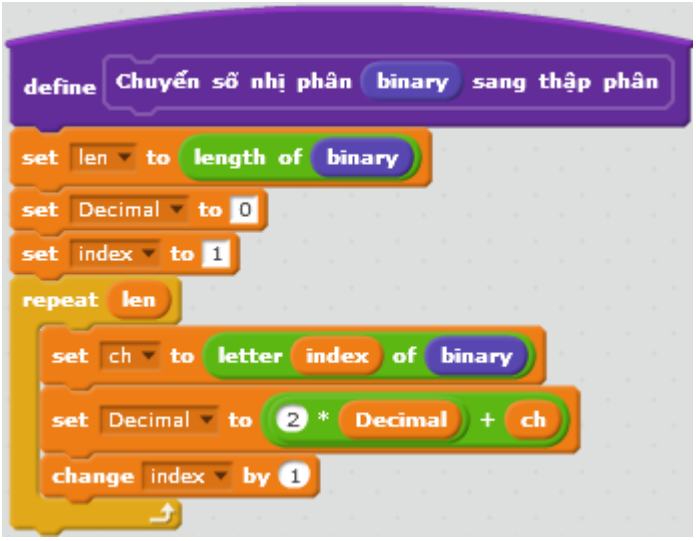
4.4. Thủ tục chuyển số thập phân sang nhị phân

Thủ tục này có 1 tham biến là **Decimal**. Kết quả việc chuyển đổi sang xâu nhị phân được lưu trong biến nhớ **Binary**.



4.5. Thủ tục chuyển số nhị phân sang thập phân

Thủ tục này có 1 tham biến là **Binary**. Kết quả chuyển đổi xâu nhị phân này sang số thập phân được lưu trong biến nhớ **Decimal**.



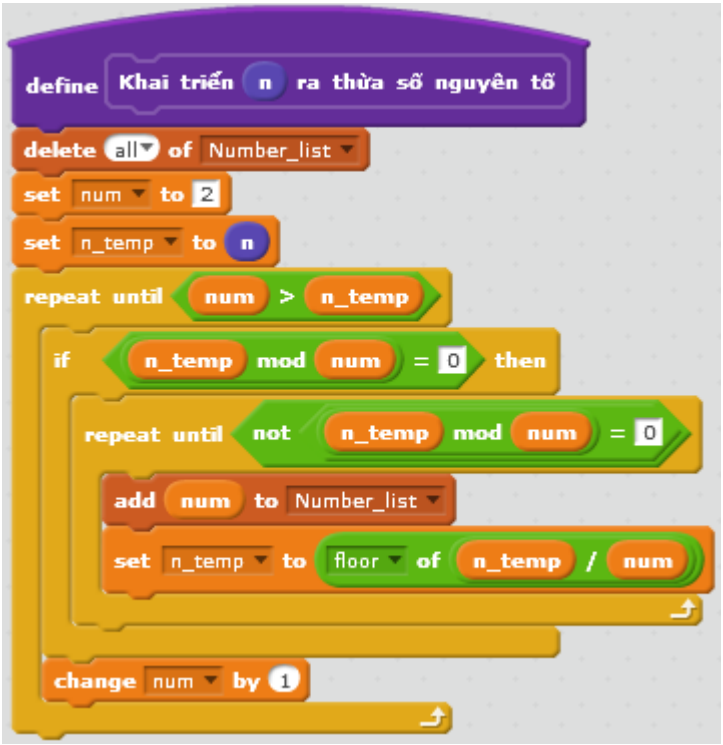
5. Một số thủ tục với số

5.1. Thủ tục kiểm tra 1 số có phải là nguyên tố hay không.

Thủ tục này có 1 tham số duy nhất là số tự nhiên **n**. Kết quả kiểm tra ghi trong biến nhớ **nguyento**. Nếu **nguyento = 1** thì **n** là số nguyên tố, ngược lại nếu **nguyento = 0** thì **n** là hợp số.

5.2. Thủ tục khai triển 1 số tự nhiên thành tích các thừa số nguyên tố

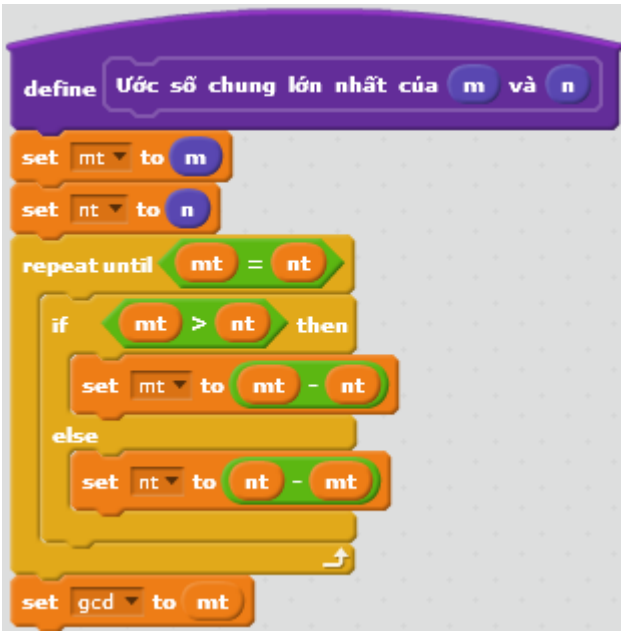
Thủ tục này chỉ có 1 tham số duy nhất là **n**. Kết quả khai triển **n** thành tích các thừa số nguyên tố được lưu trong dãy **Number_list**.



5.3. Thủ tục tính USCLN và BSCNN của 2 số tự nhiên

Cả 2 thủ tục này đều có các tham biến là các số **m** và **n**.

Kết quả ước số chung lớn nhất của 2 số trên sẽ được lưu trong biến nhớ **gcd**.



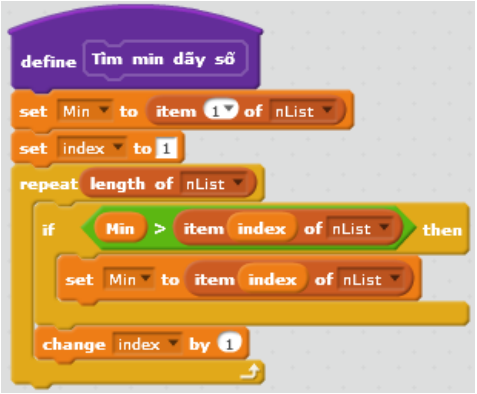
Kết quả của thủ tục tính bội số chung nhỏ nhất của 2 số **m** và **n** được lưu trong biến nhớ **lcm**.

6. Một số bài toán xử lý liên quan đến dãy số

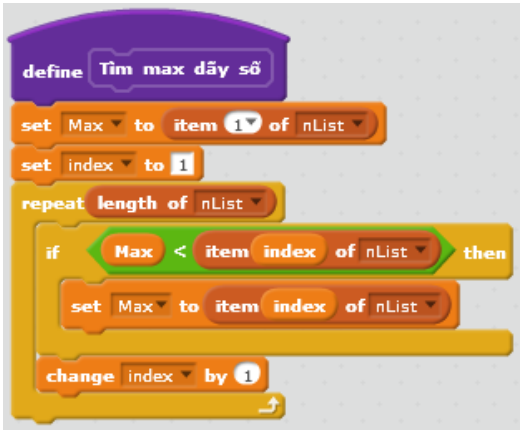
Trong hoạt động này chúng ta sẽ cùng nhau giải quyết một số bài toán cổ điển liên quan đến dãy số. Dãy số cho trước sẽ được ký hiệu là lưu trữ trong biến nhớ dạng danh sách **nList**.

6.1. Bài toán tìm phần tử nhỏ nhất và lớn nhất của 1 dãy số cho trước

Giá trị nhỏ nhất của dãy số **nList** được lưu trong biến nhớ **Min**.



Giá trị lớn nhất của dãy số **nList** được lưu trong biến nhớ **Max**.



6.2. Bài toán tính số các khoảng đơn điệu của 1 dãy số cho trước

Khoảng đơn điệu của 1 dãy số là 1 dãy con liên tục đơn điệu (tăng hoặc giảm thực sự). Để hiểu rõ hơn định nghĩa này, chúng ta quan sát 1 số ví dụ sau:

Dãy số gốc	**Phân tích**	**Số các khoảng đơn điệu**
1 1 1 1 1 1 1	dãy này chỉ bao gồm các số hạng bằng nhau nên không có bất kỳ khoảng đơn điệu nào.	0
1 2 3 3 3 -2 - 3	khoảng đơn điệu tăng đầu tiên (1 2 3), tiếp theo là 1 khoảng đi ngang, sự đi ngang không được tính là khoảng đơn điệu. Do vậy dãy này chỉ có 2 khoảng đơn điệu.	2
1 1 2 3 4 4 3 2 1 1 1 1 2	khoảng đi ngang đầu tiên (1 1) không được tính là khoảng đơn điệu. Dãy này chỉ có 3 khoảng đơn điệu.	3

Chúng ta sử dụng các biến nhớ sau:

nList: dãy số đã cho ban đầu.

count: biến đếm các khoảng đơn điệu tăng thực sự của dãy số. Đây chính là đáp án cần tìm.

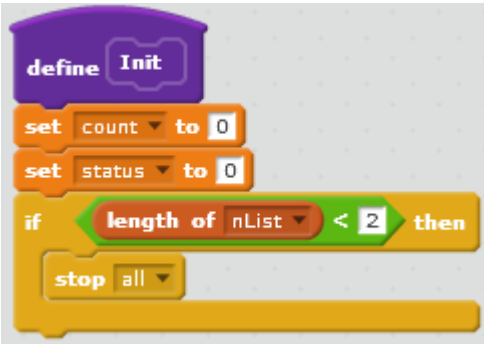
status: biến nhớ ghi lại trạng thái của phần tử hiện thời của dãy khi chúng ta đang phân tích. Ý nghĩa của biến này như sau:

status = 0, phần tử này chưa xét hoặc không cần xét.

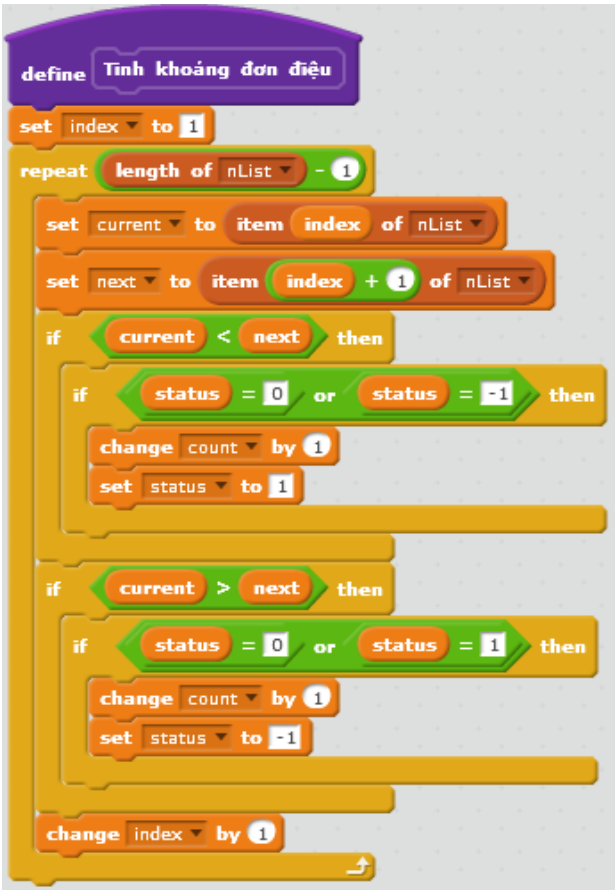
status = 1, phần tử hiện thời đang nằm trong 1 khoảng đơn điệu tăng thực sự.

status = -1, phần tử hiện thời không nằm trong 1 khoảng đơn điệu tăng thực sự.

Thủ tục **Init** sẽ đặt các giá trị ban đầu cho biến count và status. count = 0. status = 0. Kiểm tra dãy nList nếu có > 1 phần tử thì mới tiếp tục làm việc.



Thủ tục chính của bài toán.



6.3. Bài toán tìm 1 dãy con liên tục đơn điệu tăng có độ dài lớn nhất trong một dãy số cho trước

Ví dụ với dãy: 3 2 1 4 5 6 5 6 7 thì dãy con cần tìm là 1 4 5 6.

Phân tích bài toán.

Gọi dãy đã cho là $a_1, a_2, ..., a_N$. Vì dãy cần tìm là dãy liên tục nên chúng ta có thể dánh dấu và tìm được tất cả các dãy con liên tục đơn điệu tăng chỉ cần 1 lần duyệt từ đầu dãy. Nhận xét trên cho ta ý tưởng làm bài toán này chỉ cần 1 vòng duyệt tuyến tính từ 1 đến N.

Trong quá trình duyệt chúng ta sẽ sử dụng các biến nhớ sau:

Thông tin về dãy cần tìm được lưu trong các biến sau:

bmax – chỉ số phần tử đầu tiên của dãy con đơn điệu tăng cực đại.

lengthmax – độ dài của dãy con đơn điệu tăng cực đại.

Như vậy dãy này sẽ có các chỉ số là bmax, bmax+1, ..., bmax+lengthmax-1

Chúng ta cần lưu thông tin của dãy con liên tục đơn điệu tăng hiện thời, tức là dãy mà chúng ta đang khảo sát:

bcurr – chỉ số phần tử đầu tiên của dãy

lengthcurr – độ dài hiện thời của dãy.

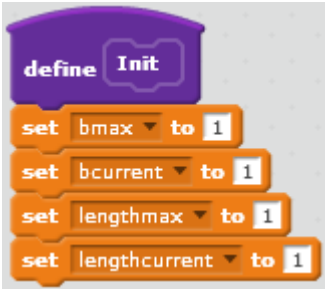
Phần khai báo ban đầu như sau:

```
bmax = 1; lengthmax = 1;
bcurr = 1; lengthcurr = 1;
```

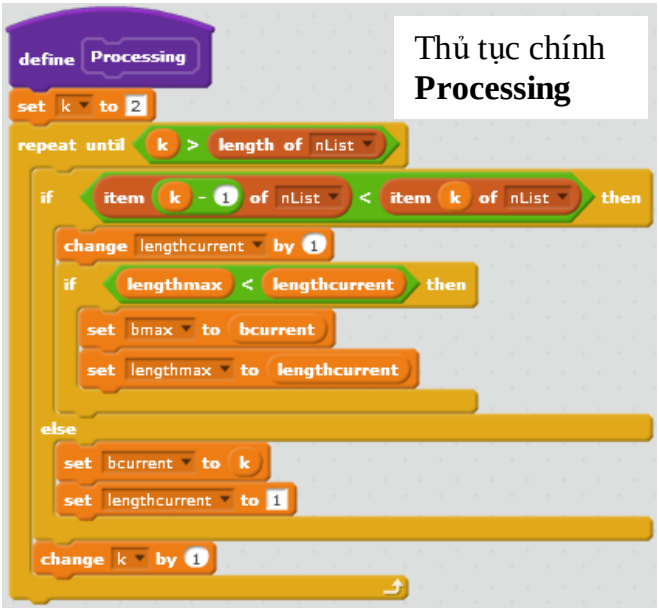
Phần chương trình duyệt chính như sau:

Ý tưởng thuật toán: Trong quá trình duyệt từ đầu đến cuối của dãy, chúng ta luôn ghi nhớ lại 2 tham số: dãy con đơn điệu tăng lớn nhất (bmax, lengthmax) và dãy con đơn điệu tăng đang xét tại vị trí hiện thời (bcurr, lengthcurr) và luôn so sánh 2 dãy này với nhau để cập nhật thông tin chính xác.

```
cho k chạy từ 2 đến N
     if a[k-1] < a[k] then
          lengthcurr:=lengthcurr + 1;
          if lengthmax < lengthcurr then
               bmax : = bcurr;
               lengthmax : = lengthcurr;
          else
               bcurr : = k; lengthcurr : = 1;
```



Thủ tục **Init** khai báo các giá trị ban đầu của các biến nhớ.



Thủ tục chính **Processing**

Toàn bộ chương trình hoàn chỉnh như sau.

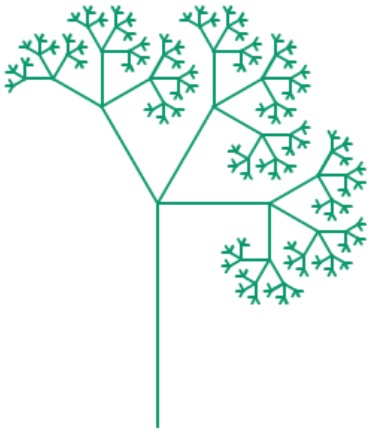
Day con lien tuc tang cuc dai.sb2

Em hãy hoàn thiện chương trình này.

7. Thủ tục đệ qui (gọi chính nó)

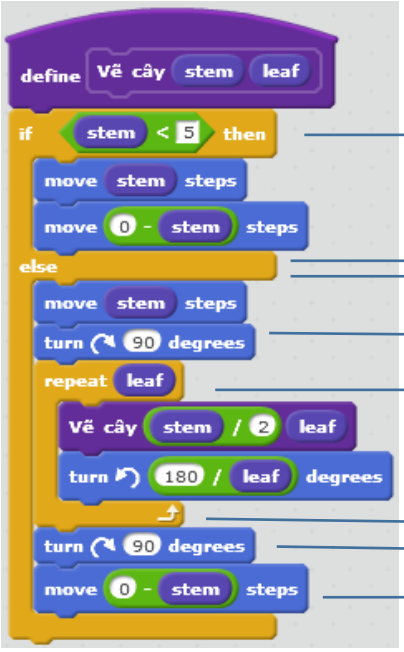
La va canh.sb2

Chúng ta hãy quan sát cây trong hình bên và thiết kế thủ tục để vẽ cây này.



Phân tích

Chúng ta cùng phân tích các qui luật vẽ của hình bên. Xuất phát từ gốc, thủ tục sẽ kẻ 1 cành gốc dài, sau đó xoay 90º để vẽ 3 cành (lá tiếp theo) có độ dài = 1/2 cành gốc ban đầu. Với mỗi cành lá đó, thủ tục lại được gọi tiếp tục đệ qui. Do vậy thủ tục cần thiết lập sẽ có 2 tham số là độ dài cành và số lượng lá.



Thủ tục Vẽ cây với 2 tham số: **stem** (độ dài cành) và **leaf** (số lượng lá)

Đoạn chương trình kiểm tra điều kiện kết thúc: nếu độ dài cành < 5 thì chỉ vẽ cành thôi.

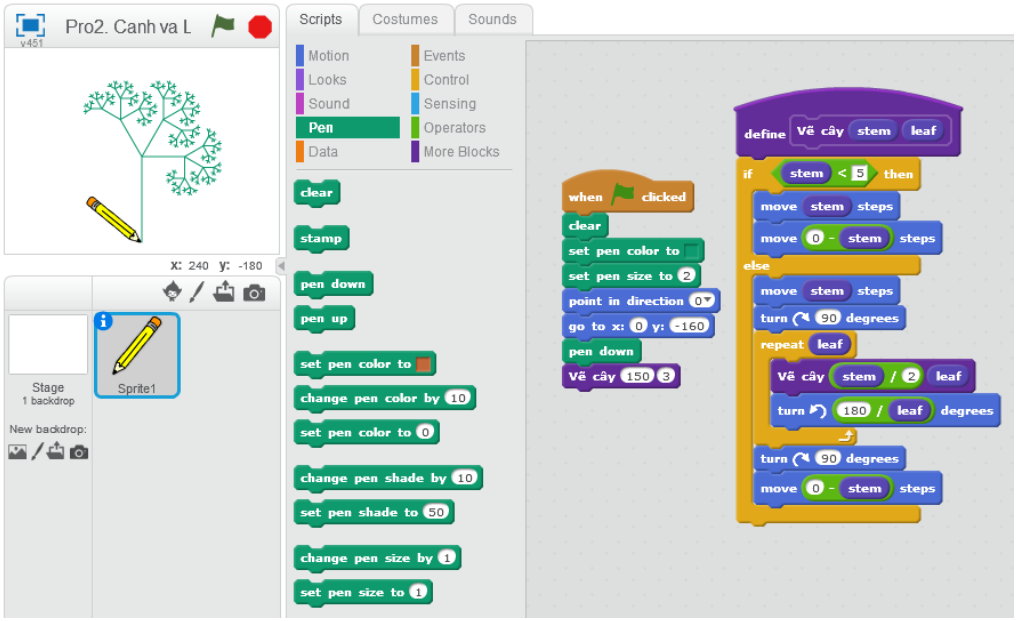
Bắt đầu phần chính của thủ tục vẽ cây.

vẽ cành chính, sau đó xoay phải 90º để bắt đầu vẽ tiếp

Đoạn chương trình lặp gọi đệ qui để vẽ tiếp lá bên trong. Vẽ xong quay lại vị trí cũ và xoay trái 60º

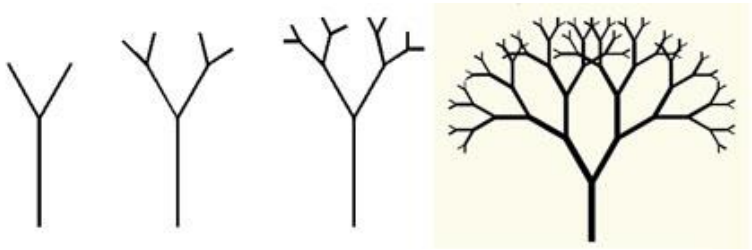
Khi ra khỏi lời gọi đệ qui thì xoay phải thêm 90º nữa để quay lại vị trí xuất phát.

Giao diện hoàn chỉnh của chương trình như hình dưới đây.



Viết lại toàn bộ chương trình trên.

Em hãy thay đổi thủ tục trên để có thể vẽ được các hình sau:



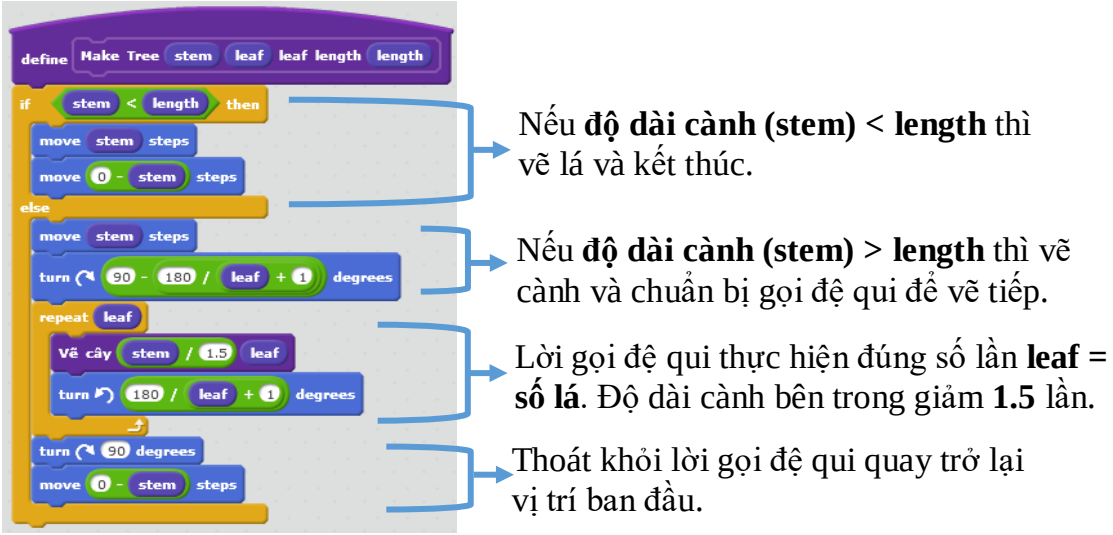
8. Bài toán vẽ cây, lá hoàn chỉnh

Trong hoạt động này chúng ta sẽ mở rộng thủ tục vẽ cây trong phần trên để có nhiều lựa chọn hơn trong việc thể hiện hình vẽ cây.

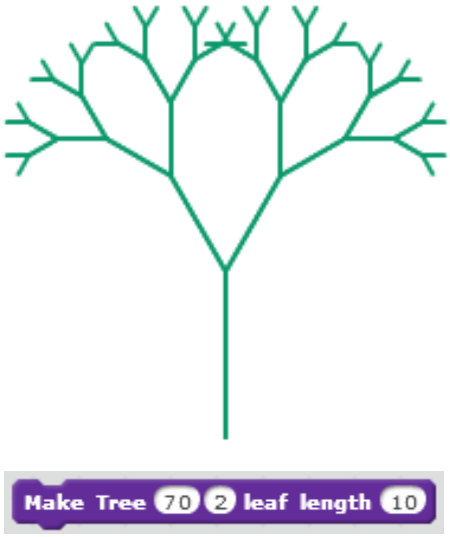
Thủ tục mới có tên **Make Tree** và có 3 tham số sau:

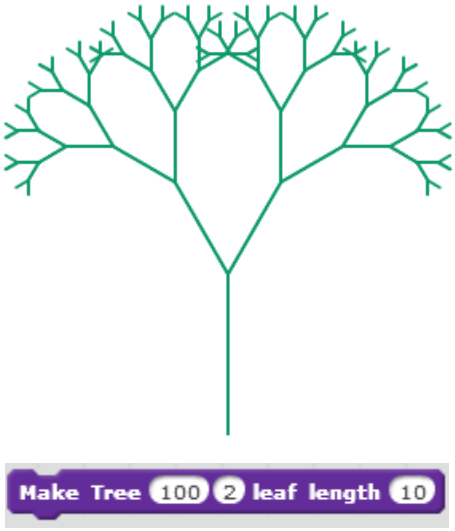
stem: độ dài cành gốc; **leaf:** số lượng lá và **length:** độ dài của lá cuối cùng.

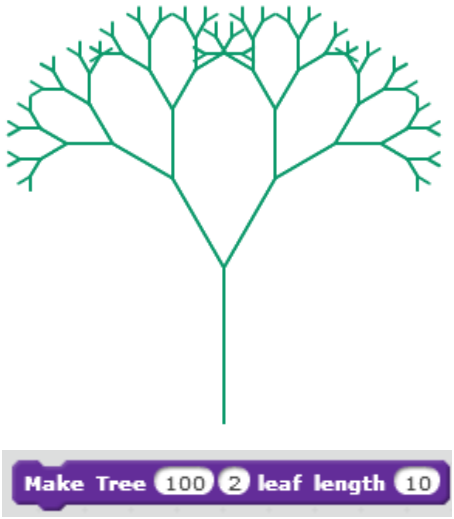
Thủ tục được thể hiện bằng lệnh Scratch như sau:

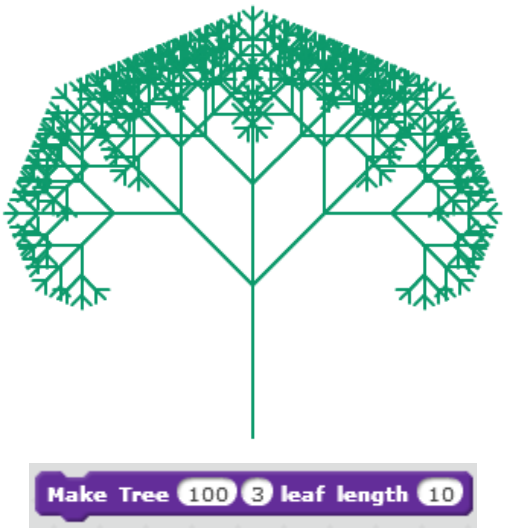


Kết quả thể hiện rất đa dạng trong bảng sau.









Em hãy mở rộng thủ tục trên, bổ sung thêm 1 tham số nữa để chỉ ra tỉ lệ **ratio** của độ dài cành khi gọi thủ tục đệ qui so với cành gốc ban đầu.

Ví dụ thủ tục mới sẽ có dạng:

Câu hỏi và bài tập

1. Có người nói "tạo một thủ tục tức là tạo thêm một lệnh mới", đúng hay sai?

2. Viết thủ tục với tham số n tính số hạng dãy Fibonaci thứ n.

3. Cho trước 1 dãy số. Viết chương trình tìm ra 1 dãy con liên tục có các số bằng 0 có độ dài cực đại trong dãy trên.

4. Cho trước 2 dãy số **List1**, **List2**. Viết chương trình gộp 2 dãy này vào thành 1 dãy và sắp xếp theo thứ tự tăng dần. Kết quả đưa vào dãy **Listout**.

5. Viết thủ tục với 2 tham số là tọa độ tâm vòng tròn (X, Y) và độ dài bán kính R.

6. Viết thủ tục vẽ hình sau:

7. Biết rằng đơn vị đo nhiệt độ theo Celcious C và theo Fahrenheit F có quan hệ với nhau bởi công thức:

F = (9/5) * C + 32.

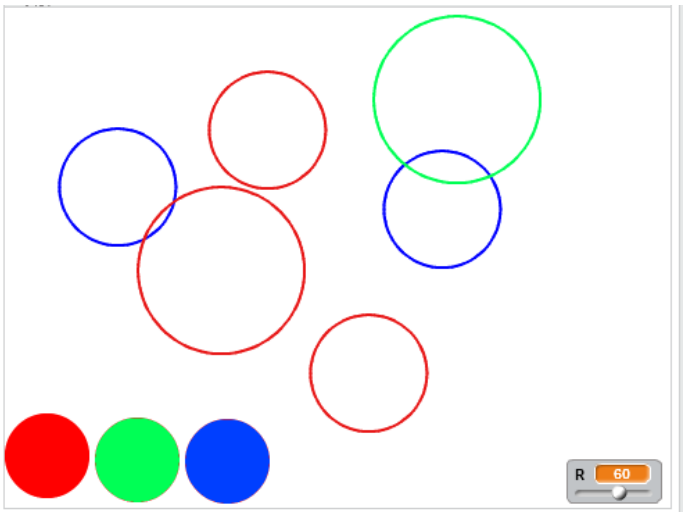
Viết 2 thủ tục, 1 thủ tục có tham số C biến đổi nhiệt độ từ C sang F và 1 thủ tục có tham số F biến đổi nhiệt độ từ F sang C.

8. Viết thủ tục với tham số là 1 xâu nhị phân, thủ tục biến đổi xâu nhị phân này thành số thập phân và hiện đáp số trên màn hình bằng lệnh **say**. Nếu tham số đầu vào không là xâu nhị phân thì thủ tục thông báo: đầu vào sai.

9. Viết 1 thủ tục với tham số là 1 tên người Việt Nam. Thủ tục có chức năng xóa đi các ký hiệu trống thừa trong tên này, ví dụ các ký tự trống thừa ở đầu, cuối và bên trong. Ví dụ với tên được nhập là " Nguyễn Quang Vinh " thì thủ tục cần sửa lại là "Nguyễn Quanh Vinh".

Ve vong tron.sb2

10. Viết chương trình **Vẽ vòng tròn** với yêu cầu như sau:



Các nhân vật của chương trình.

Biến nhớ R cho phép người chơi thay đổi giá trị ngay trên màn hình. Giá trị này phải > 0.

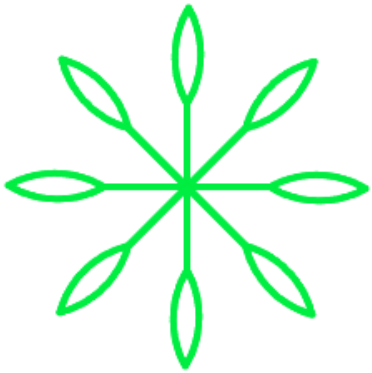
Chương trình hoạt động như sau:

Người dùng có thể nháy chuột lên 3 nút tròn để chọn màu sắc, dùng chuột tương tác để điều khiển giá trị của bán kính R. Khi nháy chuột lên 1 vị trí trống bất kỳ trên màn hình, chương trình sẽ vẽ 1 vòng tròn với tâm là điểm vừa nháy, bán kính R và màu sắc đang chọn.

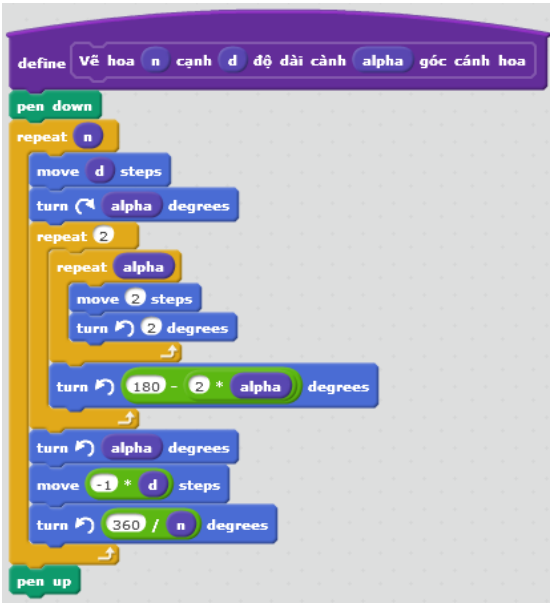
11. Em hãy viết lại tất cả các chương trình mẫu đã có trong các bài học trước đây, viết lại và có sử dụng thủ tục để chương trình trở nên sáng sủa, hợp lý hơn.

12. Viết chương trình với độ tổng quát cao nhất sử dụng thủ tục để vẽ hình các bông hoa sau:

Ve hoa tong hop.sb2

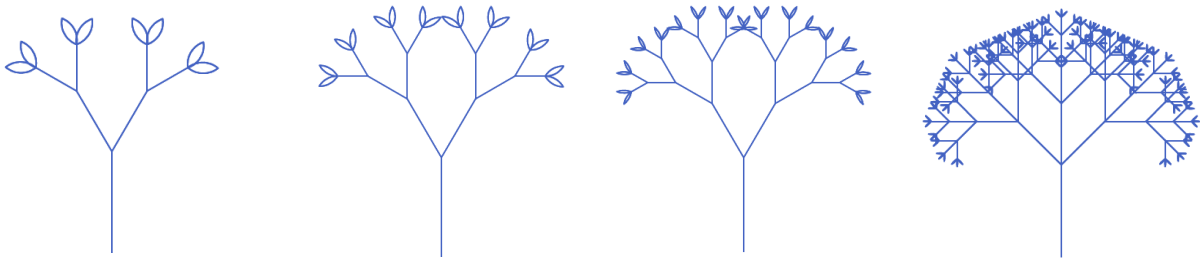


Thủ tục vẽ hoa chính sẽ như sau:

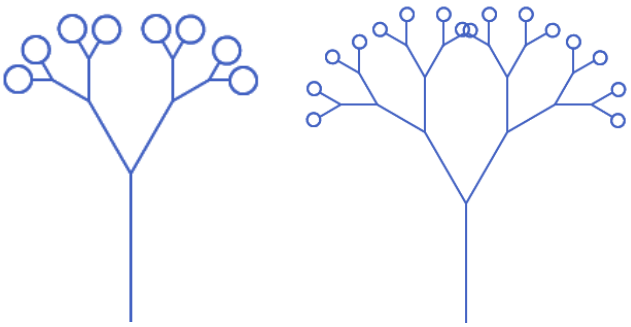


Hãy viết, thử nghiệm và giải thích chương trình trên.

13. Mở rộng chương trình của mục 8, vẽ cây lá hoàn chỉnh, yêu cầu lá có hình dạng như thật. Ví dụ 1 số hình ảnh cây, lá như vậy.



14. Sửa chương trình bài tập trên để vẽ cây lá có hoa tròn như sau:



Mở rộng

1. Math Board

Thiết kế chương trình mô phỏng bài học tính cộng, trừ đơn giản sau.

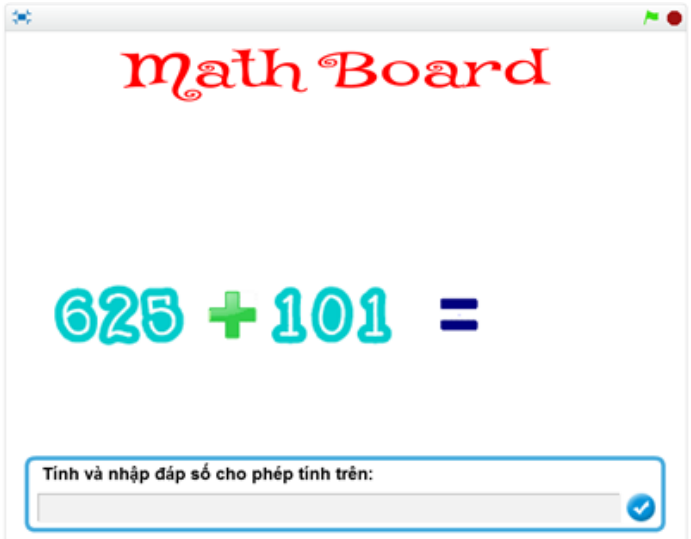
Chương trình **Math Board**

Math Board.sb2

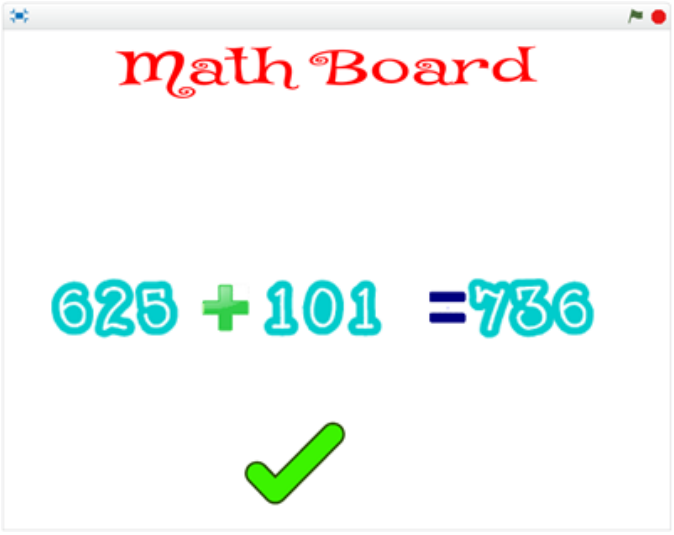
Chương trình sẽ tự động sinh các phép toán cộng, trừ số đơn giản. Người dùng nhập trực tiếp từ bàn phím đáp án. Nút Check sẽ kiểm tra đáp án đó là đúng hay ai và thông báo trên màn hình. Nếu làm sai, chương trình sẽ ghi đáp án đúng bên cạnh đáp án sai.

Sau khi làm xong, nháy nút Next để tự động sinh và chuyển bài tiếp theo.

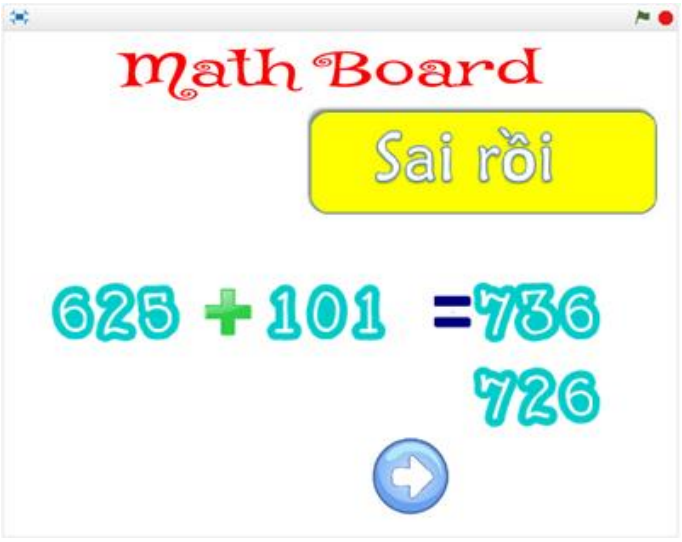
Cụ thể hơn, yêu cầu các bước thực hiện của chương trình như sau:



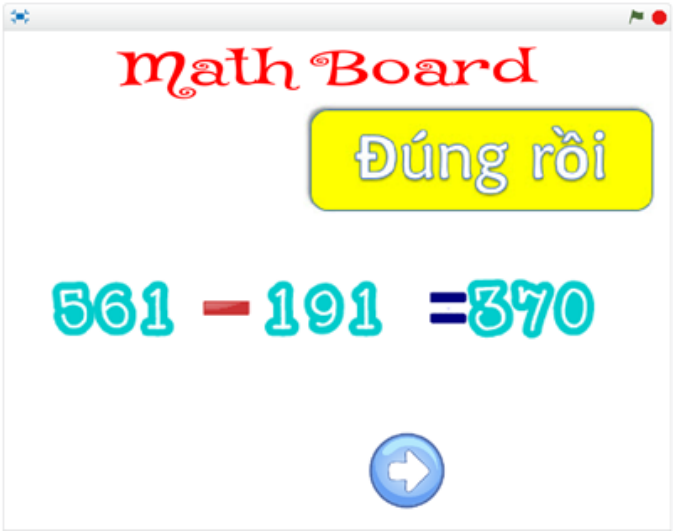
Phần mềm tự động sinh phép toán và dữ liệu bài toán, yêu cầu người chơi nhập đáp án từ bàn phím.



Khi người dùng nhập xong, đáp án do người dùng nhập sẽ hiện trên phép tính ở màn hình. Khi nó nút Check xuất hiện. Nháy nút này để kiểm tra đúng hay sai.



Nếu sai, chương trình sẽ hiện thông báo sai rồi và hiện đáp án đúng bên dưới. Đồng thời xuất hiện nút Next. Nháy lên nút này để bắt đầu 1 phép tính mới.



Nếu đúng, chương trình sẽ xuất hiện thông báo Đúng rồi. Đồng thời xuất hiện nút Next. Nháy lên nút này để bắt đầu 1 phép tính mới.

Em hãy thiết kế và hoàn thiện chương trình này.

2. Đường cong Koch

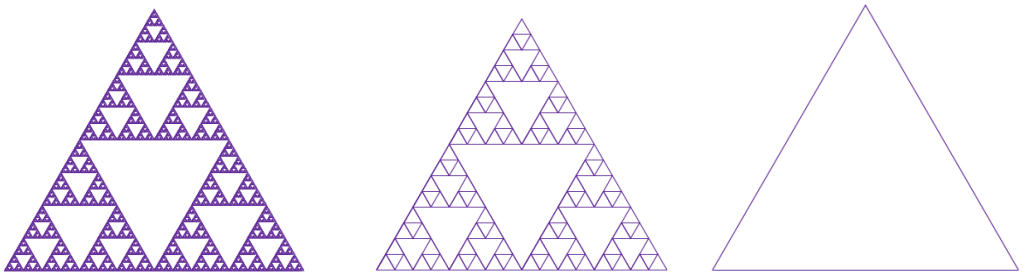
Em hãy thiết kế chương trình để vẽ đường cong Koch, một khái niệm Fractal rất quen thuộc trong toán học như sau:

Koch Curve.sb2

3. Tam giác Serpinskii

Tương tự bài trên, em hãy thiết kế thủ tục để có thể vẽ được các hình sau, được gọi là các Tam giác Serpinskii.

Serpinski-Triangle.sb2



Bài 20. Clone 1. Phân thân của nhân vật

Mục đích

Học xong bài học này, bạn sẽ hiểu được:

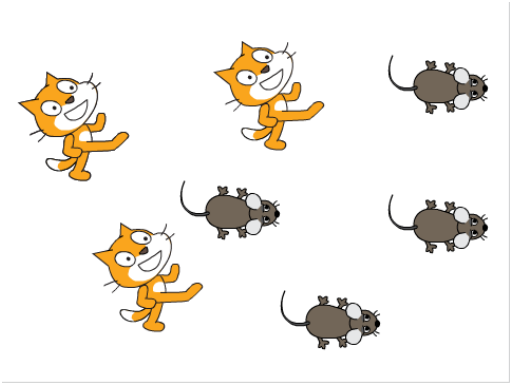
- Khái niệm phân thân (clone) của nhân vật và ý nghĩa của Clone.

- Tính chất và thuộc tính của Clone.

- Một vài ứng dụng Clone trong các bài toán thực tế.

Bắt đầu

Em hãy nhìn vào hình ảnh 1 ứng dụng của Scratch sau và có nhận xét gì?



Trên màn hình em sẽ thấy nhiều con mèo và chuột cùng chuyển động. Rõ ràng đây không phải là hình ảnh của lệnh **stamp**, mà phải là các nhân vật.

Để thiết kế chương trình trên em phải làm gì?

- Sử dụng lệnh **stamp** liên tục?

- Tạo ra nhiều nhân vật có hình dạng giống nhau và cho chúng chuyển động?

- Em có cách nào khác hay không?

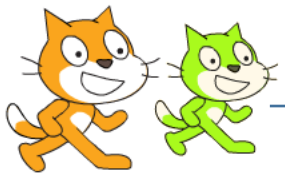
Trong bài học này, chúng ta sẽ làm quen với 1 khái niệm hoàn toàn mới trong Scratch có thể giải quyết dễ dàng bài toán trên.

Nội dung bài học

1. Khái niệm phân thân - clone trong Scratch

Tất cả các nhân vật trong Scratch đều có thể "phân thân", tức là tạo ra các nhân vật khác là song sinh với chính bản thân mình. Các nhân vật được phân thân đó gọi là Clone.

Chúng ta cùng xem 1 chương trình ngắn để bước đầu làm quen và phân biệt được 2 khái niệm: nhân vật chính (gốc) và phân thân (clone) của nhân vật này.



Nhân vật chính, gốc

Nhân vật được phân thân, là song sinh của nhân vật gốc. Nhân vật này được gọi là **clone** của nhân vật chính.

Chúng ta cùng xem đoạn chương trình của nhân vật chính con mèo.

Đoạn chương trình điều khiển **clone**.

Đoạn chương trình khởi tạo **clone**.

Nhân vật gốc và Clone

Nhân vật gốc là khái niệm Nhân vật (Sprite) mà chúng ta vẫn biết từ trước đến nay trong môi trường Scratch.

Mỗi nhân vật khi được tạo ra sẽ có các tính chất, thuộc tính riêng như hình ảnh, tọa độ x, y, hướng quay, trang phục, âm thanh, kích thước, các biến nhớ riêng. Mỗi nhân vật có 1 cửa sổ lệnh riêng của mình.

Mỗi nhân vật có thể tạo ra các phân thân (clone) của riêng mình.

Clone được khởi tạo từ 1 nhân vật gốc. Clone là 1 phân thân của nhân vật gốc và có mọi thuộc tính của nhân vật gốc.

Khi bắt đầu khởi tạo, clone kế thừa toàn bộ tính chất, thuộc tính của nhân vật gốc. Tuy nhiên sau khi ra đời, có thể điều khiển clone bằng tất cả các lệnh của Scratch như 1 nhân vật bình thường. Điểm khác biệt chỉ ở chỗ:

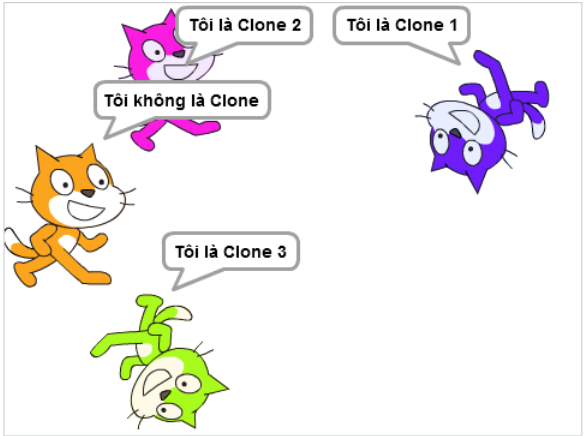
- Các chương trình điều khiển Clone bắt buộc nằm trong lệnh sự kiện **when I start as a clone**.

- Bản thân clone có thể xóa chính mình.

Bảng sau liệt kê các thông tin nhanh liên quan đến nhân vật gốc và clone.

	Nhân vật gốc	Clone
Mô tả nhanh	Là nhân vật hoạt động chính trên sân khấu cho người sử dụng khởi tạo, hoạt động vĩnh viễn.	Là nhân vật phân thân từ 1 nhân vật gốc, được khởi tạo bởi lệnh **create clone of**. Clone sau khi được tạo ra sẽ có đầy đủ tính chất như 1 nhân vật bình thường và kế thừa mọi thuộc tính từ nhân vật gốc của mình. Clone không hoạt động vĩnh viễn.
Khởi tạo Clone	Nhân vật chính dùng lệnh  để tạo Clone. Có thể tạo Clone của mình hoặc của các nhân vật khác. Chú ý: Sân khấu cũng có quyền tạo Clone cho mọi nhân vật.	
Điều khiển Clone		Clone sau khi được tạo ra sẽ chịu sự điều khiển của câu lệnh sự kiện  . Cho phép nhiều chương trình cùng điều khiển 1 Clone.
Xóa Clone		Clone tự xóa bản thân mình bằng lệnh  .
Thời gian sống	Vĩnh viễn	Chỉ hoạt động trong thời gian chạy chương trình.

Chúng ta cùng xét 1 ví dụ sau. Ví dụ này minh họa cho quan hệ giữa nhân vật chính và các phân thân - clone của chính mình.



Tạo 1 biến nhớ riêng có tên CloneID của nhân vật chính. Chương trình sẽ lần lượt tạo ra 3 clone, và trước mỗi lần tạo sẽ gán giá trị CloneID lần lượt là 1, 2, 3. Các clone này sẽ kế thừa biến nhớ CloneID cho riêng mình với các giá trị lần lượt là 1, 2, 3. Khi tạo ra, các clone sẽ di chuyển ngẫu nhiên, tự do trên màn hình và luôn hiện giá trị CloneID của riêng mình để phân biệt với các clone khác.

Chương trình cụ thể trên Scratch như sau:

Chương trình chính:

Sau mỗi giây, tăng CloneID lên 1 đơn vị và tạo 1 Clone mới của Mèo. Clone mới này sẽ mang giá trị biến CloneID tương ứng bằng 1, 2, 3. Giá trị này sẽ gắn với từng Clone và không thay đổi nữa. Sau khi tạo xong 3 Clone thì nói "tôi không là Clone".

Chương trình cho mỗi Clone:

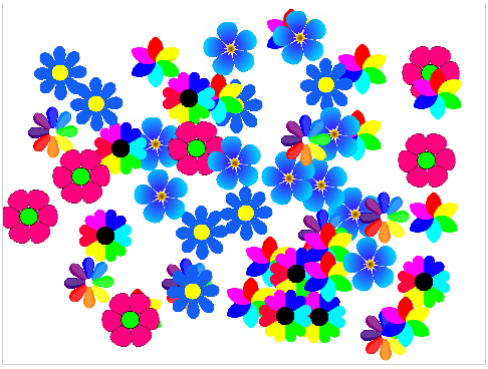
Khi được sinh ra, mỗi Clone sẽ đọc CloneID của riêng mình và sau đó chạy xung quanh sân khấu, sau mỗi 0.1 giây thì thay đổi màu áo của mình.

Clone có ý nghĩa gì trong các ứng dụng thực tế của Scratch, chúng ta cùng tìm hiểu các hoạt động tiếp theo.

2. Rừng hoa

Chương trình đơn giản sau cho em hiểu thêm hoạt động của clone.

Rung hoa.sb2



Chương trình đơn giản này chỉ có đúng 1 nhân vật là bông hoa, nhưng với nhiều hình ảnh đẹp mắt khác nhau.

Mục đích của chương trình là tạo ra 1 rừng hoa với nhiều màu sắc sặc sỡ.

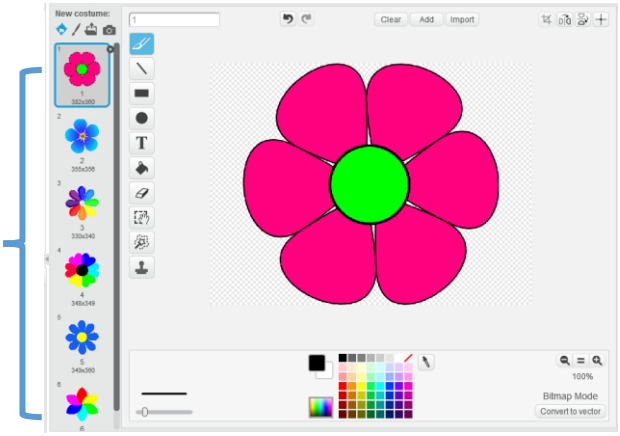
Để làm như vậy, em cần sưu tầm nhiều hình ảnh hoa với màu sắc đa dạng khác nhau và đưa vào thành các trang phục của nhân vật này.

Chương trình được xây dựng đơn giản như sau:

- Nhân vật chính bông hoa cần tạo ra nhiều hình ảnh khác nhau nhưng có kích thước gần giống nhau làm trang phục. Ví dụ:

Thiết lập các hình bông hoa có kích thước gần giống nhau và màu sắc đa dạng khác nhau.

Dãy các trang phục (costume) được đánh số từ 1



- Trong mạch chương trình chính, em cho bông hoa lần lượt tạo ra 50 clone của chính mình.

Lệnh này sẽ tạo ra 50 bản clone của nhân vật chính. Các phân thân clone này khi tạo ra có trang phục và khuôn dạng giống với nhân vật chính và ở tại đúng vị trí của nhân vật chính (nhưng nằm phía dưới).

- Mỗi phân thân, clone sẽ di chuyển nhanh tới 1 vị trí ngẫu nhiên trên màn hình với thể hiện trang phục ngẫu nhiên.

Em hãy hoàn thiện chương trình này.

3. Trò chơi mèo đuổi chuột

Trong hoạt động này, em hãy thiết kế trò chơi Mèo đuổi Chuột.

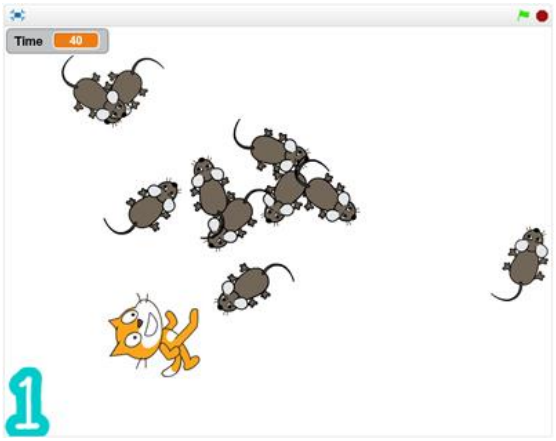
Nhân vật chính của chúng ta là Mèo và Chuột. Trò chơi được mô tả đơn giản như sau:

- Lũ chuột sẽ được phân thân và chạy lung tung (ngẫu nhiên) trên màn hình. Thấy mèo từ xa chuột đã quay đầu bỏ chạy.

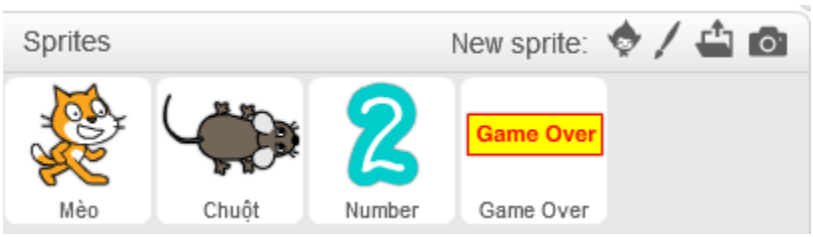
- Mèo được điều khiển bởi con người (dùng các phím điều khiển). Nhiệm vụ của người chơi là điều khiển mèo đuổi và bắt chuột càng nhiều càng tốt.

- Thời gian mỗi lần chơi chỉ là 1 phút. Trên màn hình luôn hiển thị số lượng chuột đã bị bắt. Khi kết thúc trò chơi, bạn nào bắt được số chuột nhiều hơn sẽ chiến thắng. Khi kết thúc, thông báo Game Over sẽ hiện trên màn hình.

Meo va chuot.sb2



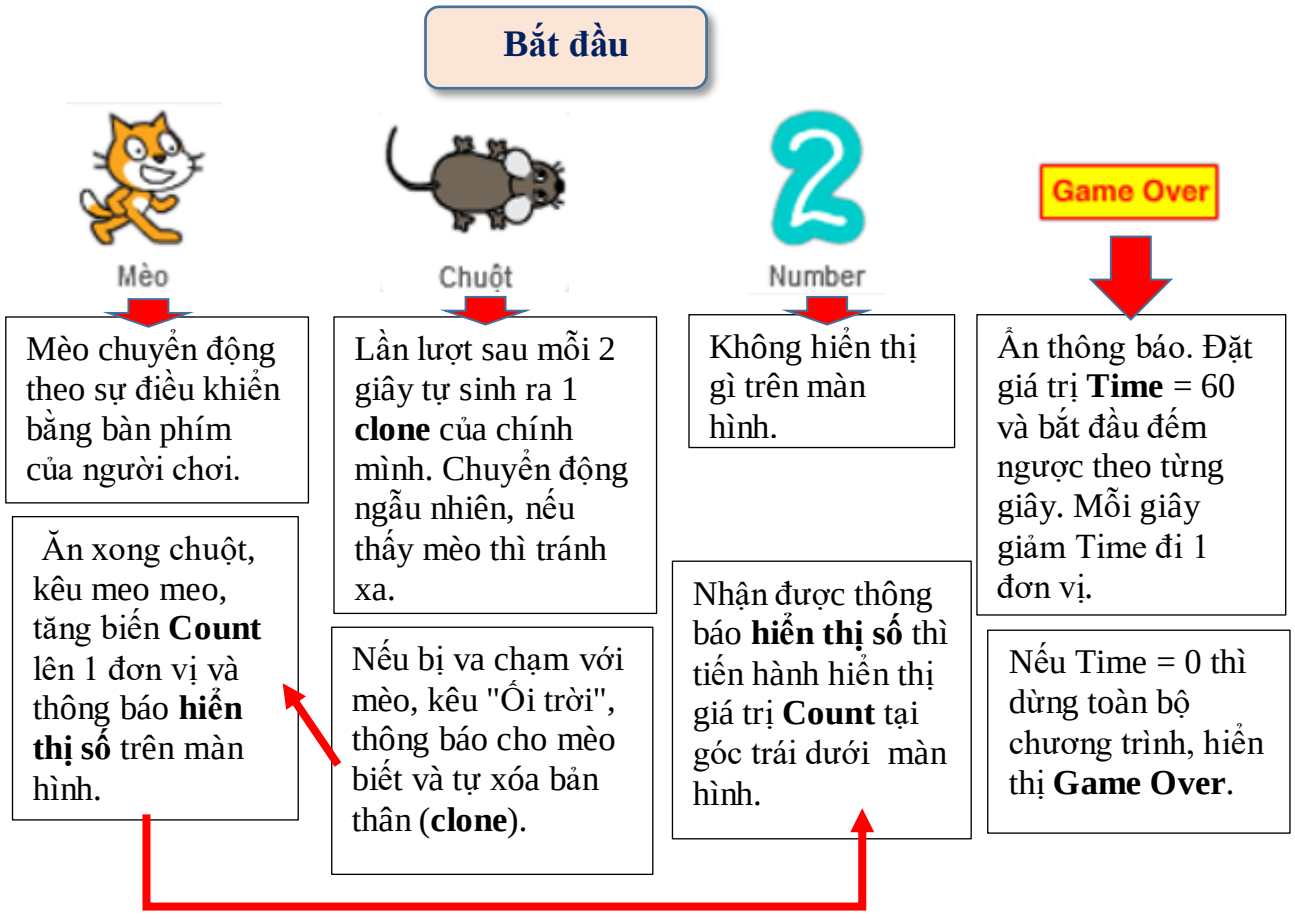
Để thiết kế chương trình chúng ta bắt đầu từ các nhân vật. Hệ thống nhân vật bao gồm: mèo, chuột, các chữ số (đủ từ 0 đến 9) và nút thông báo Game Over.



Sử dụng 2 biến nhớ tổng thể:

Time - thời gian chơi và **Count** - số lượng chuột bị bắt.

Sơ đồ hoạt động của trò chơi như sau:

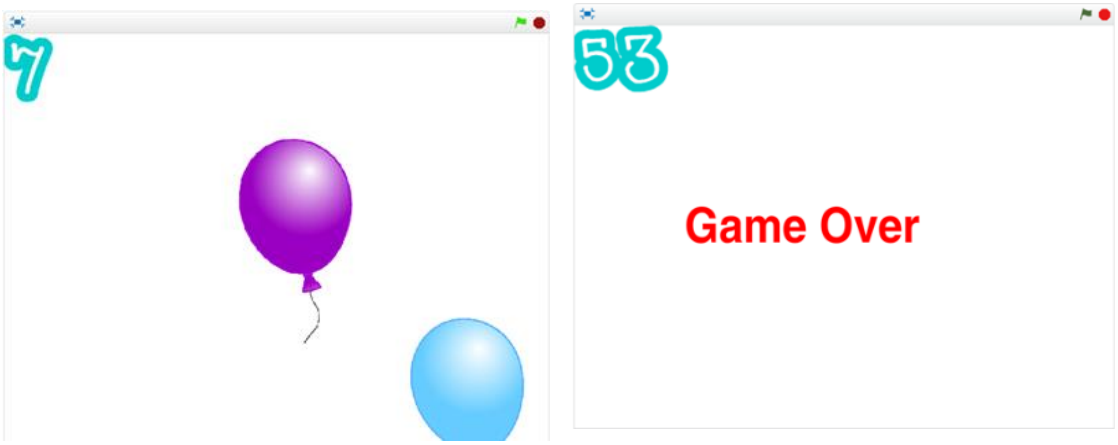


4. Trò chơi bóng bay

Trò chơi được mô tả như sau:

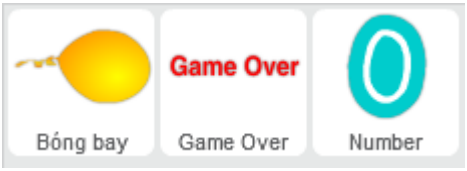
Từ phía dưới các quả bóng bay sẽ xuất hiện ngẫu nhiên và bay lên trên. Nhiệm vụ của người chơi là nháy chuột lên các quả bóng bay này. Mỗi khi nháy lên quả bóng bay sẽ nổ và biến mất. Thời gian chơi là 1 phút. Góc trái trên màn hình sẽ thể hiện số bóng bay đã được nháy trúng. Người thắng cuộc là người đã làm nổ nhiều số bóng bay nhất. Khi hết thời gian, màn hình xuất hiển dòng chữ và âm thanh "Game Over".

Bong bay.sb2



Gợi ý thiết kế chương trình.

Em hãy thiết lập 3 nhân vật sau: **Bóng bay**, **Chữ số** và biển thông báo **Game Over**. Ngoài ra cần có thêm 2 biến nhớ chung là **Time** - thời gian dùng để đếm ngược khi chơi và **Count** - biến nhớ ghi lại số lượng bóng bay đã được đánh trúng.



Nhiệm vụ mỗi nhân vật trên như sau:

1. Bóng bay

Bóng bay sẽ được tự động tạo các **clone** của mình và bay lên từ 1 vị trí ngẫu nhiên ở phía dưới. Bóng bay được thiết kế có 1 số hình ảnh với màu sắc khác nhau. Người chơi sẽ nháy chuột lên các quả bóng này. Nhiệm vụ của người chơi là nháy càng nhiều càng tốt lên các quả bóng bay. Bóng bay sẽ biến mất nếu vượt ra ngoài màn hình hoặc bị người chơi nháy đúng.

2. Chữ số

Góc trái trên màn hình sẽ luôn thể hiện số quả bóng bay đã bấm đúng. Nhân vật này sử dụng biến nhớ **Count** để đếm số bóng bay bị nháy chuột đúng và thể hiện con số này tại góc trái trên màn hình.

3. Biển thông báo Game Over

Trò chơi kết thúc sau 60 giây. Nhân vật Biển thông báo này có nhiệm vụ đọc thời gian và thông báo kết thúc trò chơi sau đúng 60 giây. Biến **Time** dùng làm nhiệm vụ đếm ngược thời gian.

Câu hỏi và bài tập

1. Viết chương trình cho con Mèo gốc phân thân thành 3 clone, 1 con chạy sang trái, 1 con chạy sang phải và 1 con chạy lên trên.

2. Em hãy làm cho trò chơi **Rừng hoa** đẹp lên như sau:

Sau khi xuất hiện các bông hoa sẽ tự động đổi màu và xoay tròn trên màn hình. Em sẽ thấy một bức tranh sắc màu sặc sỡ, lung linh đẹp mắt.

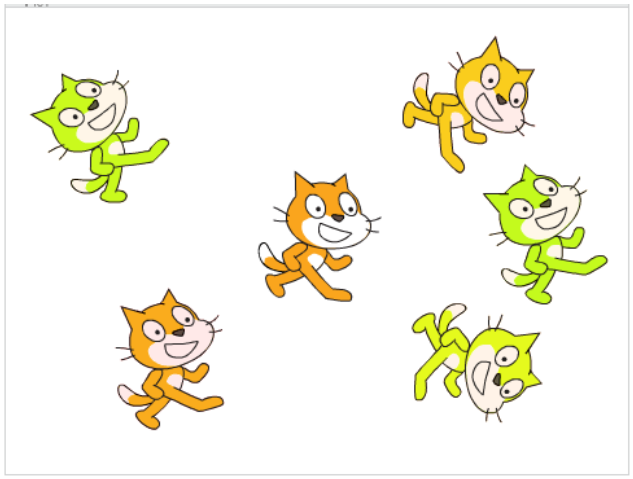
3. Các phân thân của nhân vật gốc có thể đặt tên được hay không? Vì sao?

4. Có thể lập trình điều khiển riêng cho từng Clone được hay không? Lấy ví dụ cụ thể.

5. Sử dụng kỹ thuật Clone, lập trình thực hiện bài toán sau:

Chương trình: **5 Mèo Clone**.

Từ 1 con Mèo gốc tạo ra 5 con Mèo Clone khác (xem hình), các chú mèo clone này thay đổi màu sắc và di chuyển đến các vị trí ngẫu nhiên trên màn hình.



Mèo gốc luôn ở giữa màn hình.

Nháy chuột lên các mèo clone thì các con mèo clone này sẽ biến mất.

Nhưng nháy lên mèo gốc thì không ảnh hưởng gì.

6. Thiết kế chương trình **5 Stars** đơn giản sau, sử dụng công cụ lập trình Clone.

Chương trình có 2 nhân vật chính: Mèo con và Ngôi sao.

Ở trạng thái ban đầu xuất hiện 5 ngôi sao là 5 clones của nhân vật ngôi sao này.

Chương trình hoạt động như sau.

- Khi khởi động, phần mềm tạo ra 5 Clone của nhân vật Ngôi sao (Stars), 5 phân thân này được xếp thẳng hàng phía trên, bên phải màn hình. Bản thân nhân vật gốc Stars sẽ được ẩn trên màn hình.

- Khi người dùng nháy chuột lên Mèo, thì Mèo sẽ phát tiếng kêu meo meo và lần lượt các ngôi sao (clone) trên màn hình sẽ biến mất, từ trái sang phải. Như vậy sau 5 lần nháy chuột lên Mèo thì cả 5 ngôi sao sẽ biến mất và chương trình kết thúc.

7. Viết chương trình thực hiện chức năng sau:

Nhân vật mèo ở tâm của sân khấu. Mèo sẽ tự động tạo ra 1 clone của mình. Nhân vật Mèo Clone này sẽ chạy theo hướng ngẫu nhiên về 1 hướng, khi gặp cạnh của màn hình thì biến mất. Ngay lúc đó Mèo lại tạo ra 1 Clone khác. Quá trình cứ như vậy lặp lại mãi.

Chương trình tự kết thúc sau thời gian 30 giây.

8. Mở rộng chương trình **Rừng hoa** (xem mục 2 của bài học) như sau:

- Phần đầu của chương trình giống như đã có: phần mềm tự động sinh 50 bông hoa Clone của nhân vật Bông hoa ban đầu, các bông hoa clone này xuất hiện ngẫu nhiên trên màn hình và có trang phục ngẫu nhiên.

- Phần bổ sung cần thực hiện. Sau đó phần mềm tiếp tục bổ sung thêm các bông hoa clone như trên sau mỗi 0.1 giây, nhưng cũng từ đó cứ sau 0.1 giây thì có 1 bông hoa clone sẽ mất đi (lấy ngẫu nhiên trong các hoa clone này).

Em hãy viết chương trình bổ sung như trên cho **Rừng hoa**.

9. Mở rộng bài tập 6, **5 Stars** thành **5 Stars 2 người chơi** như sau.

Giao diện ban đầu của phần mềm khi bắt đầu chạy.

Chú ý chỉ có 1 nhân vật là ngôi sao (Stars), 2 nhân vật là nút lệnh Player 1 và Player 2.

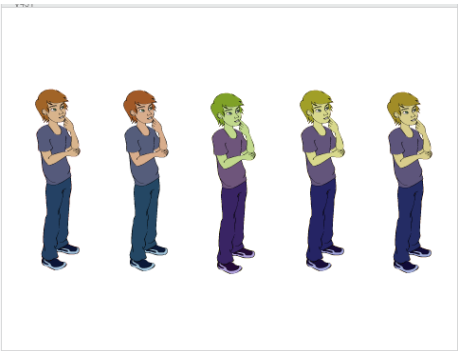
- Nhân vật chính của chương trình là ngôi sao Star (có 2 trang phục) và 2 nút Player 1 và Player 2.

- Khi bắt đầu chương trình ngôi sao sẽ phân thân thành 10 clone và thể hiện theo 5 clones phía trên, 5 clones phía dưới như trong hình.

- Nếu nháy chuột lên nút Player 1, các ngôi sao phía trên sẽ lần lượt mất đi từ trái qua phải.

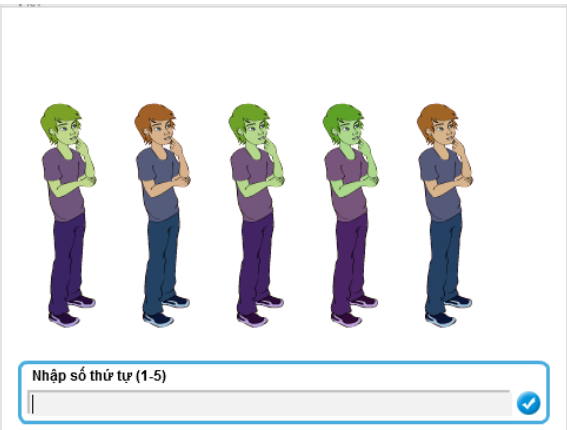
- Nếu nháy chuột lên nút Player 2, các ngôi sao phía dưới sẽ lần lượt mất đi từ trái qua phải.

10. Viết chương trình **5 anh em** như sau.

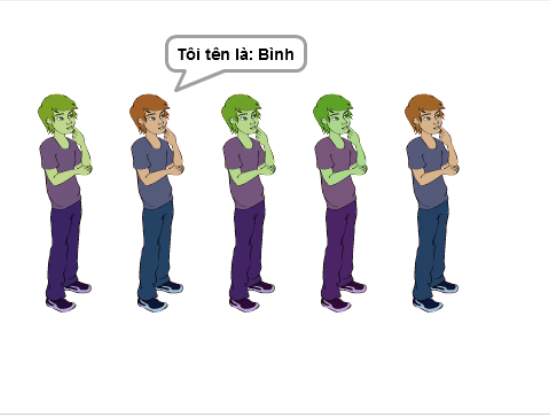


5 anh em tên là: Hà, Bình, Cường, Vinh, Quang là các clone xuất phát từ 1 nhân vật duy nhất.

Chương trình có 1 nhân vật duy nhất, khi bắt đầu sẽ được phân thân thành 5 anh em như trong hình. Mỗi bạn có 1 tên theo thứ tự từ trái qua phải là Hà, Bình, Cường, Vinh, Quang.



Phần mềm sẽ liên tục đưa ra câu hỏi để nhập 1 số thứ tự từ 1 đến 5.

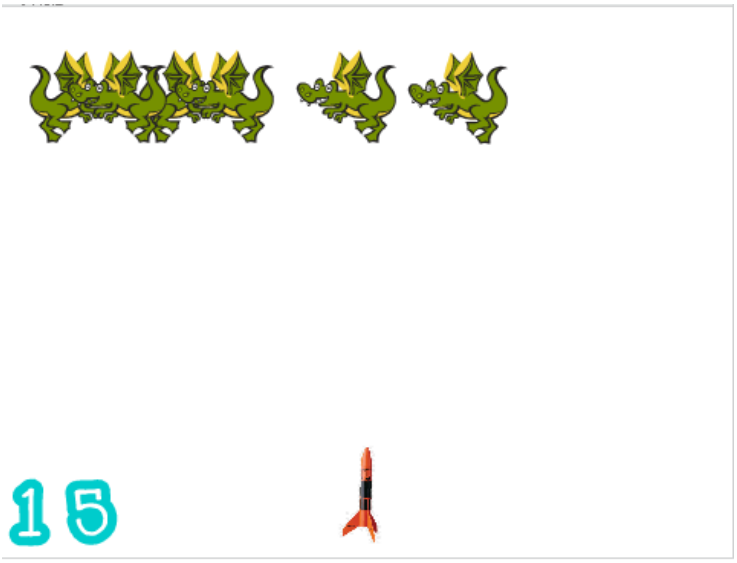


Khi nhập xong thì bạn có số thứ tự đó sẽ nói "Tôi tên là ….".

Mở rộng

Thiết kế trò chơi vui sau: **Tên lửa bắn rồng**.

Ten lua ban rong.sb2



Phía trên các con rồng sẽ xuất hiện ngẫu nhiên và bay ngang qua màn hình theo chiều ngang. Phía dưới có hình 1 tên lửa. Nhiệm vụ của người chơi là điều khiển tên lửa này đến bắn rồng. Người chơi thắng cuộc nếu bắn được > 100 con rồng. Số con rồng bắn được luôn hiện tại góc trái dưới. Người chơi sẽ thua nếu số lượng rồng xuất hiện quá đông > 10. Người chơi dùng phím Space để bắn, dùng phím trái, phải để điều khiển hướng của tên lửa.

Các nhân vật chính của trò chơi này.

Sơ đồ hoạt động của chương trình như sau.

Sau mỗi giây lại sinh ra 1 clone, tăng biến **gragon-count** lên 1 và chuyển động ngang

Nếu gặp tên lửa thì tăng biến **count**, tự biến mất và giảm biến **dragon-count** đi 1 đơn vị. Thông báo **Show Number**.

Chịu sự điều khiển của người chơi, hướng theo cách người dùng phím điều khiển.

Người dùng bấm phím Space - sinh ra 1 clone và bắn tên lửa theo hướng lên trên.

Nếu bắn trúng rồng, tự biến mất. Nếu gặp cạnh trên màn hình cũng biến mất.

Nếu nhận thông điệp **Show Number** thì hiển thị số **count** trên màn hình.

Kiểm soát 2 giá trị: **count** và **dragon-**

Nếu **count** > 100, dừng chương trình và hiển thị **Victory**.

Nếu **dragon-count** > 10 thì dừng chương trình và hiển thị **Game Over**.

Bài 21. Clone 2. Thuộc tính của phân thân nhân vật

Mục đích

Học xong bài học này, bạn sẽ hiểu được:

- Khái niệm thuộc tính của nhân vật và kế thừa cho Clone.

- Biến nhớ riêng là thuộc tính của Clone.

- Một số ứng dụng sâu hơn của Clone.

Bắt đầu

Trong bài học trước chúng ta đã biết cách thiết lập các phân thân (clone) của nhân vật để tạo ra các bài toán cần vẽ và xử lý rất nhiều nhân vật có hình ảnh giống nhau. Sử dụng phân thân (clone) sẽ làm giảm nhẹ công việc thiết kế bài toán.

Câu hỏi đặt ra cho bài toán clone: khi tạo ra các phân thân, chúng ta có cách nào để nhận biết, phân biệt và điều khiển riêng rẽ từng nhân vật đã được tạo ra không?

Giả sử có 1 nhân vật Mèo con phân thân thành 2 Mèo khác như hình sau. Làm thế nào để phân biệt và điều khiển độc lập từng chú mèo này.

Những câu hỏi như trên và nhiều vấn đề thú vị khác nữa sẽ được giải quyết khi em học bài học này.

Nội dung bài học

1. Hai chú mèo sinh đôi

Giả sử chúng ta có 1 chương trình mà Mèo mẹ sinh ra 1 cặp mèo sinh đôi. Hai chú mèo này, một có tên "Mèo vàng", một có tên "Mèo đen". Chương trình sử dụng công nghệ Clone để từ 1 nhân vật ban đầu tạo ra 2 Clone tương ứng với cặp mèo sinh đôi như hình dưới đây.

Hai chu meo sinh doi.sb2

Yêu cầu của chương trình: khi nháy chuột lên nhân vật thì nhân vật sẽ nói và kêu lên, ví dụ: Tôi là Mèo vàng / Mèo đen.

Thiết lập thêm 2 trang phục cho nhân vật chính của chương trình với tên Mèo vàng, Mèo đen.

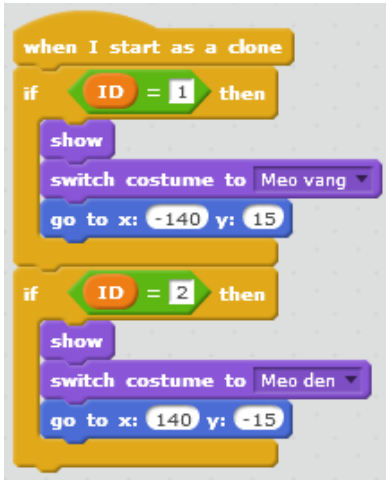
Giả sử chúng ta cần thiết lập 2 clone của nhân vật Mèo mẹ này, nhân vật 1 (ID=1) có tên là Mèo vàng, nhân vật 2 (ID=2) có tên là Mèo đen. Để thực hiện được những việc trên chúng ta sẽ thiết lập 2 biến riêng (local) của nhân vật chính là ID và Name. ID dùng để chỉ số thứ tự các clone, biến Name để lưu trữ tên của các Clone được khởi tạo. Khi bắt đầu chương trình, các biến nhớ riêng được gán giá trị ban đầu là ID = 0, Name = (empty). Đoạn chương trình sau sẽ tạo Clone đầu tiên.

Đoạn chương trình tạo Clone đầu tiên, đặt ID = 1 và Name = Mèo vàng.

Chú ý khi Clone đầu tiên tạo ra sẽ kế thừa và có giá trị ID = 1 và Name = "Mèo vàng".

Sau đây là các đoạn chương trình thiết lập ban đầu.

Khởi tạo và thiết lập 2 clones.



Đoạn chương trình thiết lập trạng thái ban đầu của 2 chú mèo sinh đôi - 2 clones.

Mỗi clone sẽ đổi trang phục của mình và dịch chuyển đến 1 vị trí của riêng mình.

Đoạn chương trình chính điều khiển clone nói tên của mình như sau:

Sự kiện thực hiện khi click chuột vẫn là lệnh **when this sprite clicked** nhưng bên trong mỗi clone sẽ được điều khiển riêng bởi câu lệnh **if <ID = ..> then**.

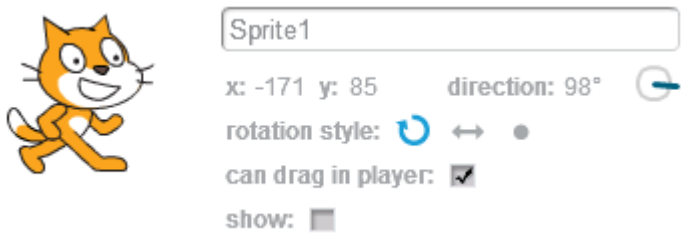
Đây là lệnh điều khiển cho clone đầu tiên.

Em hãy hoàn thiện chương trình này.

2. Thuộc tính của nhân vật

Thuộc tính của nhân vật là gì? Chúng ta đã được làm quen với rất nhiều thông tin thuộc tính (hoặc các thông tin có liên quan khác) nhưng chưa bao giờ trả lời một cách chính xác câu hỏi này.

Thuộc tính nhân vật thông qua hộp hội thoại thuộc tính.



Các thuộc tính bao gồm: Tên nhân vật; tọa độ hiện thời (x, y); hướng; kiểu xoay (rotation style); có thể dịch chuyển bằng chuột trong khi chạy chương trình; hiện/ẩn trên màn hình.

Trong Scratch, các thuộc tính quan trọng nhất của nhân vật được lưu cùng nhân vật thông qua các biến nhớ hệ thống và có thể truy cập bởi các nhân vật khác (hoặc sân khấu) thông qua 1 hàm thuộc tính nằm trong nhóm Cảm biến.

Danh sách thuộc tính của nhân vật có thể truy cập từ hàm số trên như sau:

Stt	Tên thuộc tính nhân vật	Mô tả ý nghĩa
1	**x position**	Tọa độ X hiện thời của nhân vật.
2	**y position**	Tọa độ Y hiện thời của nhân vật.
3	**direction**	Hướng hiện thời của nhân vật.
4	**costume #**	Số thứ tự trang phục của nhân vật (tính từ 1).
5	**costume name**	Tên trang phục hiện thời của nhân vật.
6	**size**	Tỉ lệ kích thước hiện thời của nhân vật (tính theo %, 100% = kích thước thật).
7	**volume**	Tỉ lệ âm lượng âm thanh hiện thời của nhân vật (tính theo %, 100% = âm thanh thực).

Tương tự đối với sân khấu, các thuộc tính của sân khấu cũng có thể lấy các thuộc tính từ hàm số.

Danh sách thuộc tính của sân khấu được ghi trong bảng sau:

Stt	Tên thuộc tính sân khấu	Mô tả ý nghĩa
1	backrop #	Số thứ tự nền sâu khấu hiện thời (tính từ 1 theo DS nền sân khấu đang có).

Stt	Tên thuộc tính sân khấu	Mô tả ý nghĩa
2	backdrop name	Tên nền sân khấu hiện thời.
3	volume	Tỉ lệ âm lượng âm thanh hiện thời của sân khấu (tính theo %, 100% = âm thanh thực).

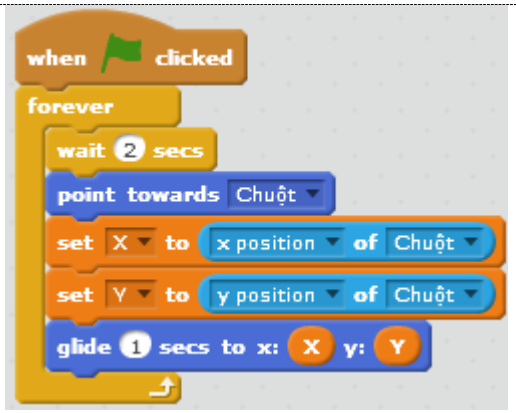
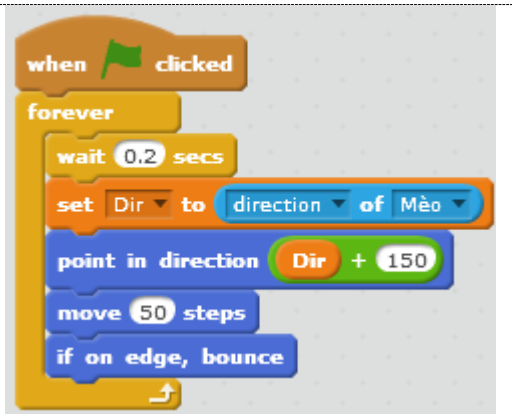
Sau đây là một số ví dụ ngắn sử dụng tính chất nêu trên.

1) Mèo đuổi chuột

Meo duoi chuot.sb2

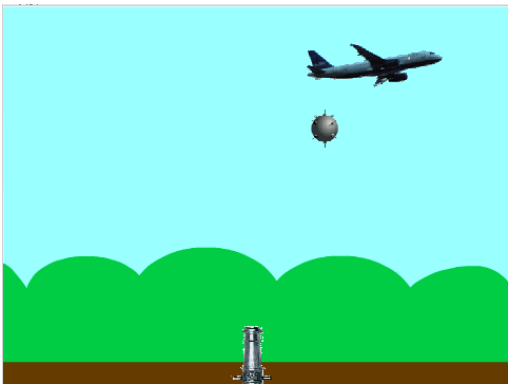

Mèo lười nên cứ sau mỗi 2 giây mới quan sát, định hướng chuột và chạy thẳng một mạch đến vị trí của chuột để bắt. Chuột nhanh nhẹn hơn, sau mỗi 0.2 giây, chuột đều cảnh giác quan sát vị trí của Mèo và thay đổi hướng để chạy ra xa nhất có thể. Tất nhiên khi gặp tường thì bị bật lại.

Các đoạn chương trình sau mô tả hoạt động đuổi bắt của Mèo và chạy của Chuột. Mèo và Chuột luôn phải xác định tọa độ và hướng đi của đối phương, do đó phải sử dụng hàm lấy thuộc tính.

Mèo	Chuột
	

2) Bắn máy bay bằng đạn thông minh

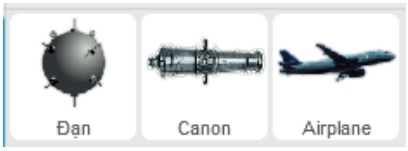
Ban may bay.sb2

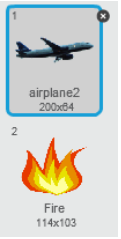


Chương trình mô phỏng bắn máy bay, tuy nhiên yêu cầu của chương trình là khi viên đạn được bắn ra khỏi nòng sẽ tự tìm hướng đến máy bay (viên đạn thông minh).

Chúng ta sẽ cùng thiết kế trò chơi đơn giản này như sau:

Nhân vật của chương trình.





Máy bay có 2 hình ảnh lúc bình thường và khi trúng đạn bị nổ.

Đạn cũng có 2 trang phục lúc bình thường và khi đạn nổ.

Thiết kế, điều khiển chương trình như sau:

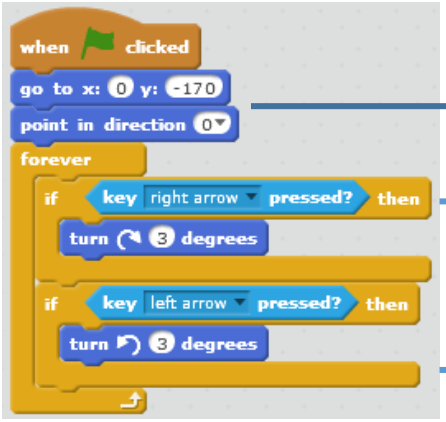
- **Súng canon:** Người dùng sẽ điều khiển nòng súng bằng các phím Phải, Trái để xoay nòng súng hướng lên máy bay. Nhấn phím Space để bắn ra 1 viên đạn theo hướng nòng súng.

- **Máy bay:** Máy bay sẽ luôn bay ngang trên bầu trời từ trái sang phải. Khi gặp bờ phải, máy bay sẽ xuất hiện lại từ bên trái.Nếu gặp đạn thì máy bay sẽ nổ tung, biến thành ngọn lửa và rơi xuống đất. Khi chạm đất chương trình kết thúc.

- **Đạn:** Khi bắn, viên đạ sẽ bay ra khỏi nòng theo hướng nòng súng, Nhưng ngay sau đó đạn sẽ điều chỉnh hướng bay, luôn hướng về máy bay. Khi trúng máy bay, viên đạn cũng sẽ nổ tung và rơi xuống đất cùng máy bay.

Lưu ý: với loại đạn thông minh này, mỗi viên đạn sẽ hạ một máy bay.

- Đoạn chương trình sau điều khiển súng Canon.



Thiết lập vị trí ban đầu và hướng nòng súng.

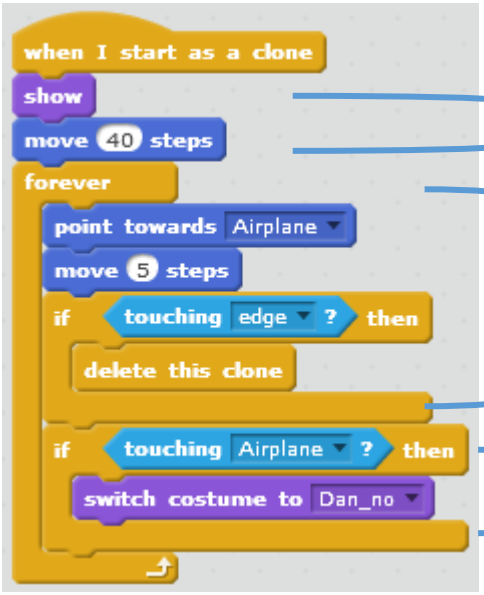
Điều khiển nòng súng bằng các phím Left, Right.

- Với nhân vật Đạn, chúng ta có 2 đoạn chương trình, một điều khiển việc tạo clone để viên đạn hình thành và một điều khiển viên đạn bay hướng đến máy bay.

Đoạn chương trình điều khiển tạo clone cho viên đạn.

Khi nhấn phím Space, đạn sẽ truy cập thông tin tọa độ của súng, di chuyển đến vị trí đó và tạo Clone sẵn sàng cho 1 viên đạn bay ra khỏi nòng súng. Chú ý nhân vật gốc này đang ở chế độ không hiển thị.

Đoạn chương trình điều khiển hướng bay của viên đạn.

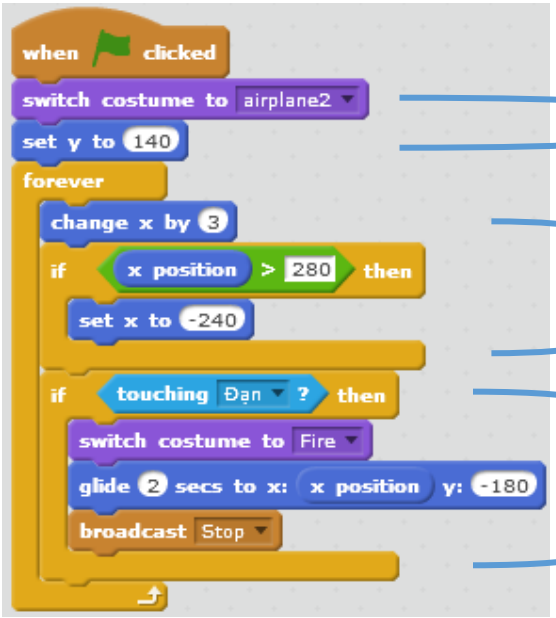


Clone - viên đạn sẽ hiện và bay theo hướng nòng súng 1 đoạn (40 bước).

Sau đó viên đạn sẽ luôn bay hướng về máy bay, nếu gặp cạnh sân khấu thì tự biến mất. (khả năng gặp cạnh ít xảy ra.

Nếu gặp máy bay, viên đạn sẽ nổ tung bằng cách thay đổi hình ảnh sang viên đạn nổ.

- Đoạn chương trình điều khiển máy bay.



Thiết lập thông số ban đầu của máy bay.

Máy bay sẽ bay ngang từ trái sang phải, nếu gặp cạnh phải thì sẽ xuất hiện lại từ bên trái.

Nếu gặp đạn, máy bay sẽ nổ tung (thay đổi hình ảnh), sau đó sẽ rơi thẳng xuống bằng lệnh glide, khi rơi xuống đất thì phát thông điệp Stop để kết thúc.

Em hãy hoàn thiện chương trình trên.

Các bài luyện tập khác có thể tham khảo trong phần Câu hỏi và bài tập.

3. Biến nhớ riêng là thuộc tính của nhân vật

Một tính chất rất quan trọng của Scratch cần nhớ là toàn bộ các biến nhớ riêng (local) của nhân vật sẽ được coi là thuộc tính nhân vật và có thể truy xuất tương tự như theo cách trên.

Trong hình sau chúng ta thấy danh sách các thuộc tính trong hàm truy xuất đã được mở rộng cho các biến nhớ riêng của nhân vật.

Các biến nhớ riêng của nhân vật sẽ được coi là thuộc tính và có thể được truy cập bằng **hàm lấy thuộc tính** từ nhóm **Cảm biến**.

Xét 1 ví dụ mô phỏng bài toán điểm danh học sinh trong lớp.

Diem danh lop.sb2

Thầy giáo lần lượt đọc tên từng bạn trong lớp. Khi đọc đến tên ai thì học sinh đó đáp "Có em", lần lượt cho đến khi kết thúc. Yêu cầu của chương trình là tên phải được khai báo là thuộc tính riêng của từng học sinh.

Chúng ta sẽ thiết lập chương trình với 6 nhân vật là Giáo viên và 5 học sinh như hình trên. 5 nhân vật học sinh có tên là HS1, HS2, HS3, HS4, HS5. Với mỗi nhân vật học sinh này, cần tạo 1 biến nhớ riêng với tên **Name** dùng để lưu Họ tên của từng nhân vật này.

Giáo viên sẽ lần lượt thực hiện 5 nhóm lệnh sau, mỗi nhóm lệnh có tác dụng gọi tên 1 học sinh và đợi HS này trả lời.

GV gọi tên học sinh thông qua hàm lấy thông số **Name**.

Sau đó GV gửi thông điệp thông báo cho HS này và chờ trả lời của HS để gọi tiếp.

Còn đây là đoạn chương trình của một học sinh (ví dụ HS1), với các học sinh khác lệnh hoàn toàn tương tự.

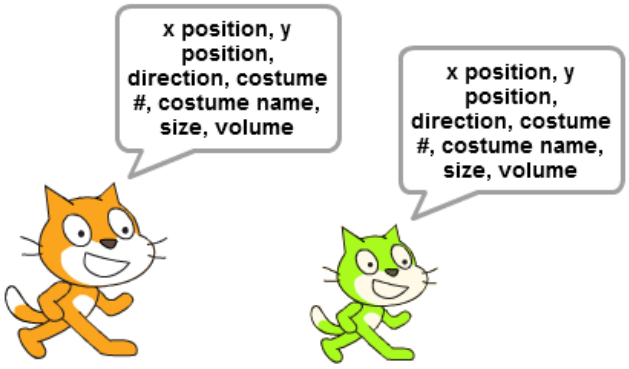
Lệnh gán trực tiếp họ tên HS vào biến Name khi bắt đầu chương trình.

HS trả lời "có em" khi nghe GV gọi tên của mình.

Em hãy hoàn thiện chương trình này.

4. Clone kế thừa thuộc tính nhân vật

Mỗi Clone được tạo ra sẽ được coi như 1 nhân vật hoàn chính và kế thừa toàn bộ các thuộc tính của nhân vật gốc của mình.



Nhân vật gốc.

Nhân vật clone - phân thân.

Khi một clone được tạo ra, clone này sẽ tiếp nhận và kế thừa tất cả các thuộc tính từ nhân vật gốc. Sau khi được tạo ra, chúng ta có thể thay đổi các thuộc tính này.

Trong bài học trước chúng ta đã được làm quen với các bài toán sinh và xử lý hàng loạt các clone của mình. Clone - phân thân của nhân vật có một đặc điểm là chỉ có 1 cửa sổ lệnh chia sẻ chung với nhân vật gốc. Do đó bài toán xử lý và điều khiển riêng rẽ các clone có lẽ là một bài toán khó. Các vấn đề cụ thể của bài toán này được ghi trong bảng sau.

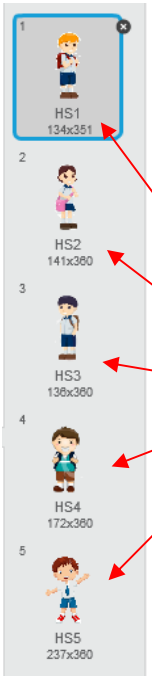
Stt	Câu hỏi, vấn đề	Các câu hỏi bổ sung
1	Làm cách nào để có thể thực hiện các lệnh điều khiển riêng rẽ từng Clone sau khi chúng được khởi tạo. Ví dụ sau khi tạo ra 2 clone của 1 con chim, làm thế nào để điều khiển chúng, 1 con bay lên, 1 con bay xuống?	
2	Các lệnh truyền thông điệp có cách nào có thể truyền thông tin đến từng phân thân riêng rẽ?	
3	Muốn thực hiện hội thoại giữa người dùng với một Clone cụ thể thì làm cách nào?	
4	Muốn truyền thông tin từ các chức năng cảm biến, ví dụ nhấn phím, cảm biến va chạm, màu sắc, … đến với từng Clone thì làm cách nào?	
5	Có thể hay không thực hiện hội thoại giữa các clone với nhau? Bằng cách nào?	

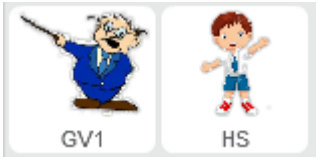
Chúng ta cùng xét lại ví dụ chương trình điểm danh lớp học. Chúng ta sẽ thiết kế lại chương trình này sử dụng clone để mô tả các học sinh của lớp học.

Yêu cầu của chương trình tương tự như ví dụ bài điểm danh lớp học ở trên, nhưng điểm khác cơ bản là bài tập này sẽ yêu cầu chỉ sử dụng 1 nhân vật HS gốc, các học sinh khác trong lớp phải được sinh ra bằng Clone.

Diem danh lop 2.sb2

Giao diện chương trình hoàn toàn tương tự: Giáo viên sẽ lần lượt điểm danh từng học sinh trong lớp. Khi thấy gọi đến tên ai thì học sinh đó nói "Có em". Cứ như vậy đến hết.



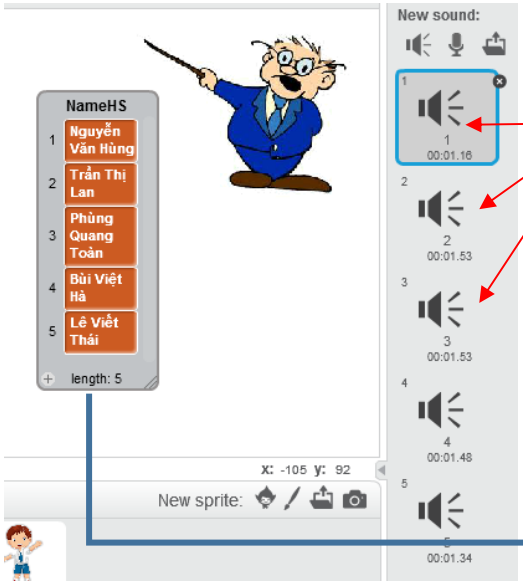


Yêu cầu của chương trình: chỉ có 2 nhân vật: giáo viên và học sinh.

Nhân vật học sinh sẽ có 5 trang phục tương ứng với hình ảnh các bạn HS trong lớp.

Thiết kế sơ bộ chương trình như sau:

(1) Giáo viên sẽ thiết lập một mảng tên HS trong lớp **NameHS** và thu âm trước gọi tên của từng HS trong lớp.

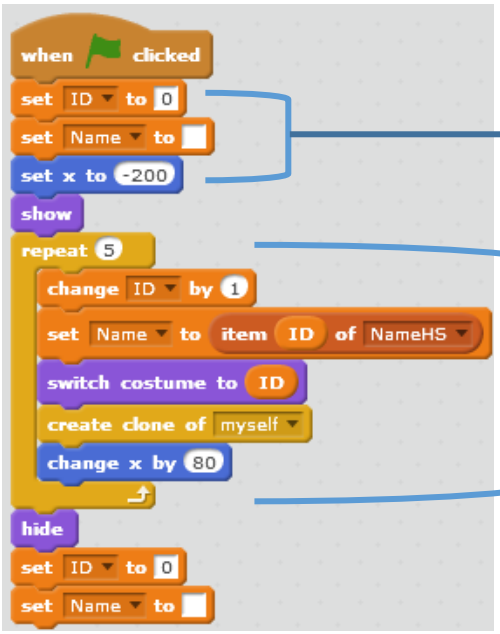


Thiết lập dãy các âm thanh gọi tên HS.

Thiết lập mảng **NameHS** lưu dãy các họ tên HS.

(2) Nhân vật HS sẽ có 2 biến nhớ riêng là **ID** (mã hóa Clone) và **Name** (tên của từng Clone - HS trong lớp).

Đoạn chương trình thực hiện khởi tạo 5 Clone của HS tương ứng với các học sinh trong lớp.



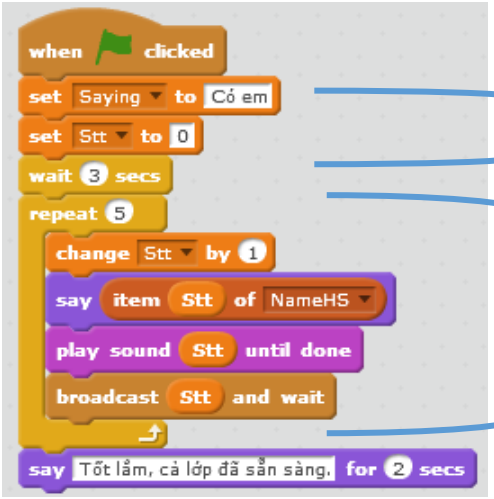
Thiết lập các thông số ban đầu cho nhân vật HS gốc.

Với mỗi vòng lặp, tăng ID lên 1, gán DS họ tên HS vào biến Name và khởi tạo Clone, dịch chuyển đến vị trí xếp hàng của lớp. Như vậy mỗi Clone sẽ có một ID và Name riêng đúng theo danh sách tên HS trong mảng NameHS.

Đây là đoạn chương trình GV lần lượt đọc tên HS và chờ để từng HS trả lời. Chú ý đặc biệt đến lệnh truyền thông điệp của GV. Sau mỗi lần đọc tên HS, GV sẽ thực hiện lệnh 

và đợi HS trả lời. Chú ý đến biến nhớ Stt có thể đưa được vào lệnh này. Khi nhận thông điệp, Clone tương ứng sẽ nhận và thực hiện bằng lệnh 

. Chú ý lệnh này không đưa biến Stt vào được mà phải nhập trực tiếp tên của thông điệp.



Thiết lập các thông số ban đầu, GV đợi 3 giây để lớp học thiết lập xếp hàng (HS tạo 5 clones).

Bắt đầu quá trình gọi tên từng HS, gửi thông điệp và đợi từng HS trả lời.

Khi điểm danh xong, GV khen và chương trình kết thúc.

Đoạn chương trình điều khiển HS trả lời. Chú ý các HS này là Clone của nhân vật gốc HS. Đoạn chương trình này có 1 số đặc biệt sau:

- Sự kiện nhận thông điệp cần gõ trực tiếp số thứ tự thông điệp tại ô thông số của lệnh.

- Sử dụng lệnh kiểm tra rẽ nhánh:

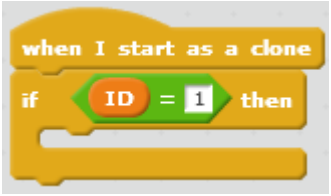
để điều khiển cho Clone với giá trị riêng ID cụ thể sẽ thực hiện khi nhận thông điệp này. Đoạn chương trình nhận thông điệp của Clone thứ nhất như sau:

Tương tự với các Clone khác.

Em hãy hoàn thiện chương trình trên.

5. Giao tiếp người dùng - clone, clone - clone

Tổng kết các câu hỏi, vấn đề liên quan đến điều khiển các Clone độc lập trong bảng sau.

Stt	Câu hỏi, vấn đề	Cách giải quyết
1	Làm cách nào để có thể thực hiện các lệnh điều khiển riêng rẽ từng Clone sau khi chúng được khởi tạo. Ví dụ sau khi tạo ra 2 clone của 1 con chim, làm thế nào để điều khiển chúng, 1 con bay lên, 1 con bay xuống?	Nhân vật gốc cần thiết lập các biến nhớ riêng dùng để phân biệt Clone, ví dụ ID dùng để mã hóa, Name dùng để đặt tên riêng. Trước khi khởi tạo Clone cần gán các tham số riêng này để Clone được kế thừa.
2	Các lệnh truyền thông điệp có cách nào có thể truyền thông tin đến từng phân thân riêng rẽ?	Lệnh truyền thông điệp được thực hiện như bình thường, điểm khác biệt chỉ ở lệnh nhận thông điệp. Câu lệnh rẽ nhánh **If …. then** cần đưa vào để xác định đúng Clone cần trả lời thông điệp.
3	Muốn thực hiện hội thoại giữa người dùng với một Clone cụ thể thì làm cách nào?	Bản thân Clone cũng có thể ra thông điệp hoặc thực hiện tất cả các lệnh giao tiếp như một nhân vật bình thường. Để lập trình cho một Clone cụ thể chúng ta có thể dùng nhóm lệnh sau: 

Stt	Câu hỏi, vấn đề	Cách giải quyết
4	Muốn truyền thông tin từ các chức năng cảm biến, ví dụ nhấn phím, cảm biến va chạm, màu sắc, … đến với từng Clone thì làm cách nào?	Tất cả các lệnh cảm biến đều có thể thực hiện đối với riêng rẽ các Clone độc lập bên trong lệnh **If …. then ….** như sau. 
5	Có thể hay không thực hiện hội thoại giữa các clone với nhau? Bằng cách nào?	Hoàn toàn có thể. Các Clone thực hiện lệnh truyền thông điệp bằng cách sử dụng nhóm lệnh:  Cách Clone độc lập nhận thông điệp đã trình bày trong mục 2.

Câu hỏi và bài tập

1. Giải thích vì sao trong chương trình mẫu ở trên, sau khi khởi tạo xong Clone, các biến nhớ như **ID** hay **Name** phải được gán lại các giá trị ban đầu?

2. Hoàn thiện chương trình Hai chú mèo sinh đôi trong mục 1.

3. Hoàn thiện chương trình Mèo đuổi chuột trong mục 2 và bổ sung thêm tính năng: mỗi khi bắt được chuột, mèo sẽ kêu lên "meo meo".

4. Hoàn thiện chương trình **Bắn máy bay thông minh** trong mục 2.

5. Bổ sung thêm các âm thanh tương ứng để làm cho chương trình **Bắn máy bay thông minh** trở nên hay hơn. Âm thanh có thể bổ sung tại các thời điểm sau:

- Khi bắn viên đạn ra khỏi nòng súng.

- Khi máy bay trúng đạn bị nổ tung.

6. Hoàn thiện chương trình **Điểm danh lớp** sau khi bổ sung thêm các âm thanh tương ứng của giáo viên và học sinh.

7. Hoàn thiện chương trình **Điểm danh lớp 2** sử dụng Clone để sinh hình ảnh các học sinh trong lớp học.

8. Thay đổi chương trình Điểm danh lớp 2 với yêu cầu sau: điểm danh bằng cách nháy chuột lên từng học sinh. Nháy lên HS nào thì HS đó trả lời "Tôi là …." như ảnh thể hiện bên dưới. Ghi chương trình này với tên Diem danh lop 3.sb2.

Diem danh lop 3.sb2

9. Thay đổi chương trình "Điểm danh lớp 2" với yêu cầu sau: người dùng sẽ lần lượt nhập tên học sinh từ bàn phím, nếu nhập đúng thì HS với tên đó sẽ hô lên "Có em", ngược lại GV sẽ nói "Không có tên như vậy".

10. Viết một chương trình đơn giản mô phỏng 1 cuộc nói chuyện, trao đổi của các bạn học sinh trong lớp học. Mỗi HS sẽ có các thông tin riêng của mình như:

- Tên.
- Quê quán.
- Chiều cao.
- Môn học yêu thích.
- Món ăn yêu thích.
- Địa chỉ nhà ở.

Chương trình sẽ tổ chức 1 cuộc trao đổi ngắn giữa các bạn trong lớp học này với các câu hỏi, trả lời xung quanh các thông tin trên. Yêu cầu các câu hỏi và trả lời phải rõ ràng từ phía người hỏi và người trả lời. Kịch bản của chương trình được thể hiện rõ theo từng bước của câu chuyện.

Hoc sinh noi chuyen.sb2

Giả sử 5 bạn trong lớp là Hùng, Lan, Toàn, Hà, Thái.

Ví dụ có thể thiết kế kịch bản đối thoại như sau.

1. Hùng (to Hà): Nhà bạn ở đâu?

Hà (to Hùng): Nhà tôi ở …..

2. Lan (to Thái): Cậu thích học nhất môn gì?

Thái (to Lan): Tớ thích nhất môn …… Còn cậu?

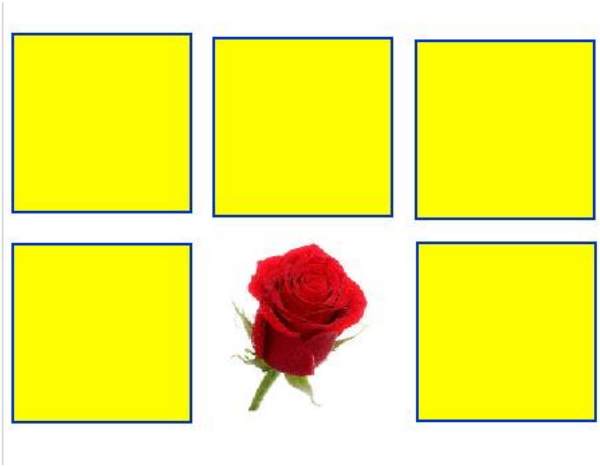
Lan (to Thái): Tớ thích nhất môn ……

Mở rộng

1. Thiết kế trò chơi đơn giản **Tìm hoa** sau đây.

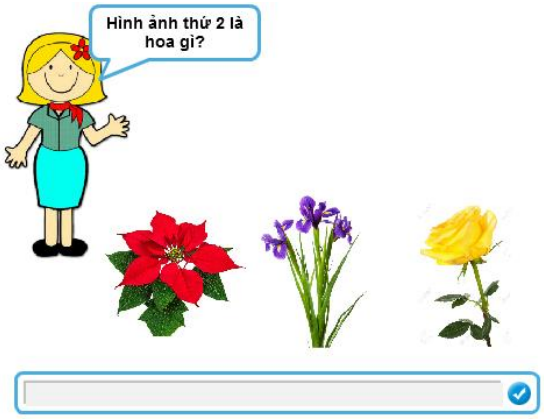
Tim hoa.sb2

Trên màn hình sẽ xuất hiện ngẫu nhiên các bông hoa với màu sắc khác nhau, bên trên các bông hoa có 1 hình vuông che phủ. Chương trình sẽ yêu cầu liên tục các câu hỏi, ví dụ: *Hãy tìm bông hoa hồng đỏ*. Người chơi sẽ nháy lên các hình vuông, nếu đúng, hình vuông sẽ mở ra và bông hoa xuất hiện. Ngược lại hình vuông mở ra 1 giây, có thông báo "sai rồi", hình vuông sẽ đóng lại như cũ và bạn tiếp tục tìm. Nếu nháy sai bị trừ điểm, nếu nháy đúng được thưởng điểm. Chương trình kết thúc khi tìm hết được tất cả các bông hoa.



2. Thiết kế trò chơi **Tìm hiểu các loài hoa** sau đây.

Tim hieu cac loai hoa.sb2



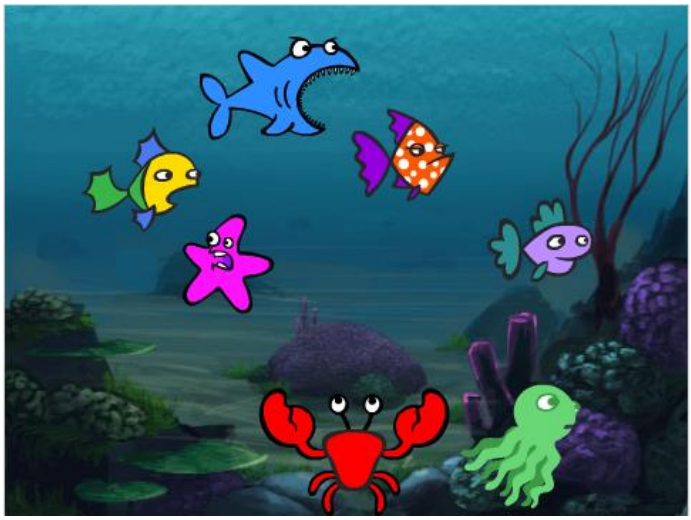
Phần mềm sẽ đưa ra liên tục các câu hỏi dạng "Đây là hoa gì?" với 3 hình ảnh hoa được sinh ngẫu nhiên từ 1 danh sách hoa cho trước. Người chơi cần nhập trực tiếp tên hoa từ bàn phím, nếu nhập đúng máy thông báo và tặng điểm, nếu sai chương trình thông báo sai, trừ điểm và tự động chuyển sang câu hỏi tiếp theo.

Chú ý: Yêu cầu của cả 2 chương trình trên là phải sử dụng kỹ thuật lập trình Clone.

3. Thiết kế chương trình **Cá bơi dưới nước** như sau:

Thiết lập chương trình mô phỏng cá bơi dưới nước với nền sân khấu và các nhân vật như hình dưới đây.

Ca boi duoi nuoc.sb2



Các nhân vật sẽ được điều khiển chuyển động như sau:

- 3 con cá con chuyển động ngẫu nhiên và di chuyển khắp toàn màn hình. Các con cá này không bao giờ bơi lộn ngược (lúc nào mắt cũng ở phía trên).

- Cá mập bơi nhanh và hung hăng toàn màn hình. Cá mập thường xuyên có những cú xoay người 180 độ. Miệng cá mập luôn há to để đớp mồi.

- Cá sứa (hình sao) chuyển động chậm, nhẹ nhàng ở nửa phía dưới đáy biển.

- Bạch tuộc luôn chuyển động chậm từ trái sang phải, vừa chuyển động vừa quẫy các đuôi của mình.

Cua luôn chuyển động ngang xung quanh màn hình, từ trái sang phải và từ phải sang trái. Thỉnh thoảng Cua sẽ thở và phun bọt khí lên trên. Khi bọt khí xuất hiện, các con cá và cá mập sẽ lao vào để đớp khí.

Video là hình ảnh mô phỏng 1 chương trình như vậy. Các bạn cố gắng viết chương trình mô phỏng gần như vậy.

https://www.facebook.com/habv2012/videos/pcb.736764283171001/1015537867948485/?type=3&theater

CHƯƠNG 5. Thiết kế trò chơi và phần mềm giáo dục

Chương 5 là chương đặc thù và khó nhất của cuốn sách. Trong chương này chúng tôi trình bày các ý tưởng, một số kỹ thuật chính và hướng dẫn chi tiết cách thiết kế một số chương trình, trò chơi, phần mềm giáo dục cụ thể. Đây có lẽ là nội dung dạy thiết kế phần mềm chi tiết đầu tiên của Việt Nam. Cái hay, cái đặc biệt của Scratch là chỉ cần học sau 1 thời gian ngắn, mỗi học sinh đều có thể tự thiết kế cho mình các trò chơi, phần mềm hoàn chỉnh theo đúng nghĩa đen và nghĩa bóng của từ này. Tóm tắt nội dung của chương 5.

- Bài đầu tiên của chương sẽ trình bày, hướng dẫn 1 số kỹ thuật thường gặp trên thực tế khi thiết kế phần mềm, trò chơi. Ví dụ như kỹ thuật tính điểm số, kỹ thuật đếm ngược, kỹ thuật thể hiện lượt chơi bằng hình ảnh các ngôi sao, kỹ thuật hiện số và chữ bằng hình ảnh. Tất cả các kỹ thuật này được mô tả chi tiết thông qua 1 trò chơi hoàn chỉnh, trò chơi **đánh bóng**.

- Bài tiếp theo sẽ trình bày chi tiết quá trình thiết kế từ ý tưởng, thiết lập nhân vật, thiết lập sơ đồ hoạt động của 1 số phần mềm, trò chơi tiêu biểu. Các trò chơi được thiết kế trong bài học này ít nhiều các em đã biết, nhưng lần đầu tiên được cùng tham gia thiết kế chi tiết, do vậy sẽ rất hấp dẫn đối với người học. Đặc biệt có những phần mềm rất nổi tiếng như Flappy Bird, Bắn súng, Xem bảo tàng, Trình diễn múa hát, Vẽ tự do, Hiệu ứng chim bay sẽ được mô tả chi tiết trong bài học này.

- 2 bài học sau cùng là phần chốt, khó nhất và phức tạp nhất của toàn bộ cuốn sách. Trong 2 bài học này, học sinh sẽ được cùng học và thiết kế các phần mềm giáo dục cụ thể nhưng khá phức tạp và cùng được tham gia thiết kế từ đầu đến cuối hoàn chỉnh. Bài 24 bao gồm 5 trò chơi, phần mềm liên quan đến số, bài 25 bao gồm 5 trò chơi, phần mềm liên quan đến chữ. Hầu hết các trò chơi này đều khá nổi tiếng trên thực tế và bây giờ học sinh có điều kiện cùng thiết kế từ đầu.

Danh sách các bài học có trong chương này.

Bài 22. Một số kỹ thuật thiết kế Games.

Bài 23. Thiết kế một số phần mềm giáo dục.

Bài 24. Các trò chơi với số.

Bài 25. Các trò chơi với chữ.


Không đâu! Lập trình Scratch hay hơn nhiều. Lập trình hấp dẫn hơn cả chơi Game.
Chơi Pokemon Go đi, hay lắm!

Bài 22. Một số kỹ thuật thiết kế Games

Mục đích

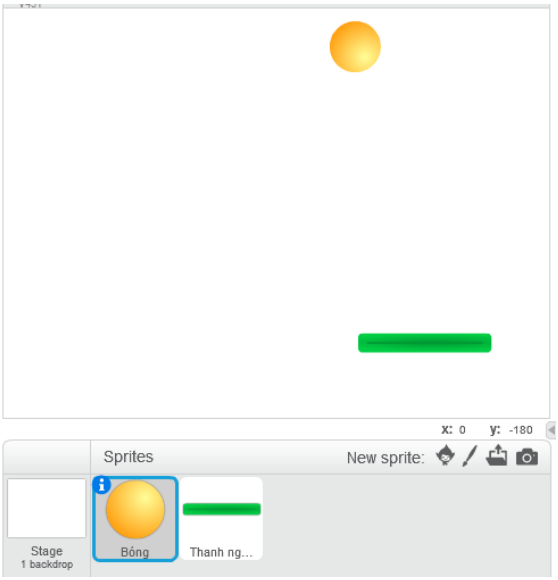
- Học sinh hiểu được 1 số kỹ thuật chính khi thiết kế phần mềm thực sự so sánh với việc chỉ viết chương trình theo yêu cầu của người khác.

- Học sinh biết được 1 vài kỹ thuật và đặc thù chính của một phần mềm hay trò chơi.

Bắt đầu

Chúng ta cùng nhau xem lại chương trình **Thả bóng** đã có trong Bài học **Chuyển động 2**. Chương trình đơn giản này được mô tả trong hình sau. Chỉ có 2 nhân vật chính là Bóng và Thanh ngang. Bóng sẽ tự động rơi xuống, nếu gặp Thanh ngang thì bật lên. Người chơi phải điều khiển Thanh ngang sao cho Bóng không rơi xuống đất. Nếu Bóng rơi xuống phía dưới Thanh ngang thì dừng chương trình.

Tha bong.sb2



Giao diện ban đầu của chương trình Thả bóng với 2 nhân vật chính là **Bóng** và **Thanh ngang**.

Chương trình điều khiển **Bóng** và **Thanh ngang**.

Em có thể thay đổi những gì trong chương trình này để biến nó thành 1 trò chơi đơn giản nhưng hấp dẫn? Hãy cùng bạn bên cạnh trao đổi các mở rộng sau đây.

- Bổ sung thêm đồng hồ đếm ngược để hạn chế thời gian chơi, ví dụ mỗi lượt chơi không quá 5 phút.

- Bổ sung đồng hồ tính thời gian chơi của người chơi, thời gian chơi càng lâu sẽ càng cao thủ.

- Bổ sung khái niệm lượt chơi. Ví dụ mỗi người chơi cho phép được 6 lượt, mỗi khi để bóng rơi xuống dưới thì mất 1 lượt. Hết lượt thì thua.

- Bổ sung thêm dấu hiệu khi thua thì xuất hiện dòng chữ Game over trên màn hình.

- Có thể bổ sung thêm vào trò chơi này những gì nữa để làm cho người chơi thêm hấp dẫn?

Muốn thực hiện những điều trên em phải làm gì?

Tất cả những vấn đề và câu hỏi trên, em sẽ được làm quen trong bài học này.

Nội dung bài học

1. Một số đặc điểm của phần mềm trò chơi

Sau đây là 1 số đặc điểm, hay thuộc tính thường có của các trò chơi hay phần mềm.

(a) Có đồng hồ tính thời gian thực.

Đồng hồ tính thời gian thực thường dùng với các mục đích sau:

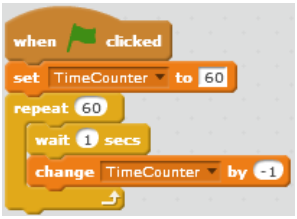
- Dùng để đếm ngược thời gian, ví dụ đếm từ 100 về 0 thì kết thúc trò chơi.

- Dùng đểm đếm xuôi thời gian, tính toán thời gian đã chơi của người chơi.

Có nhiều cách tạo các "đồng hồ" thời gian như vậy.

1. Ví dụ sử dụng 1 biến nhớ có tên TimeCounter dùng để đếm ngược thời gian. Giả sử, giá trị của biến này được gán 60 ngay từ lúc bắt đầu chương trình và nó sẽ được giảm đi sau mỗi giây, do vậy sau đúng 60 giây thì giá trị này sẽ bằng 0, hay đồng hồ sẽ được đếm ngược về 0.

Đoạn chươnng trình ngắn với biến nhớ này có thể như sau:



2. Một cách khác là dùng hàm thời gian hệ thống **timer** 

.

Đoạn chương trình sau cho phép chương trình chờ đúng 60 giây để kết thúc chương trình hoặc chuẩn bị thực hiện 1 sự kiện tiếp theo. Chương trình sẽ đặt giá trị timer = 0 và sau đó chờ cho đến khi giá trị này tăng trở lại = 60.

(b) Tương tác người chơi và máy tính chặt chẽ.

Tương tác với người chơi là đặc điểm quan trọng nhất của mọi phần mềm hay trò chơi. Các tương tác này rất đa dạng, tạo ra sự hấp dẫn của phần mềm. Có thể chỉ ra các loại tương tác sau:

1. Tương tác bằng bàn phím. Khi người dùng nhấn 1 phím sẽ tạo ra 1 sự kiện tác động lên chương trình, tạo ra 1 tương tác với phần mềm. Câu lệnh tạo sự kiện tương tác bàn phím trong Scratch là:

2. Tương tác bằng chuột.

Tương tác bằng chuột cũng đóng vai trò rất quan trọng trong mọi chương trình phần mềm. Ví dụ đối với Scratch thì tương tác bằng chuột có thể như sau:

- Sự kiện khi người chơi nháy chuột lên 1 nhân vật.

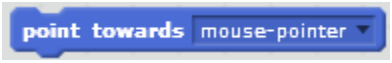
- Sự kiện khi người chơi nháy chuột lên 1 vị trí bất kỳ của màn hình, sân khấu.

- Hàm cảm biến khi nhận ra người dùng nháy chuột, và có thể tính toán được vị trí chính xác của con trỏ chuột tại thời điểm nháy chuột đó.

- Hàm tính toán khoảng cách từ nhân vật đến vị trí con trỏ chuột.

- Lệnh cho phép nhân vật luôn dính chặt với con trỏ chuột.

- Lệnh cho phép nhân vật luôn xoay về hướng con trỏ chuột.



3. Tương tác giữa các nhân vật của phần mềm.

Tương tác giữa các nhân vật được thể hiện trong Scratch có 2 loại:

- Tương tác gián tiếp thông qua thông điệp.

- Tương tác trực tiếp thông qua lệnh cảm ứng va chạm.

4. Tương tác thông qua các thông tin cảm biến khác

(c) Tính điểm để đánh giá người chơi.

Dùng điểm số để kích thích người chơi, xếp hạng người chơi. Cũng có thể dùng điểm để làm phần thưởng như tăng lượt chơi, tặng quà, …

Cách làm đơn giản nhất là dùng 1 biến nhớ để lưu trữ điểm số. Biến nhớ này sẽ gia tăng hoặc giảm bớt giá trị tùy theo các điều kiện cụ thể của trò chơi.

Thông thường điểm số sẽ luôn được thể hiện trên màn hình để người chơi quan sát được điểm của mình.

(d) Khái niệm lượt chơi thay thế cho thời gian đếm ngược.

Đây là một kỹ thuật rất hay được sử dụng trong các trò chơi, phần mềm, làm tăng độ hấp dẫn khi chơi.

Ví dụ trò chơi **thả bóng**, có thể qui định người chơi được 6 lượt chơi, mỗi lần để bóng rơi xuống phía dưới sẽ mất 1 lượt. Nếu mất hết toàn bộ 6 lượt thì thua.

(e) Mức khó, dễ của cách chơi

Hầu hết các trò chơi đều có khái niệm mức độ khó dễ. Theo thời gian chơi, phần mềm sẽ tự động tăng các mức khó, dễ này để người chơi sẽ cảm thấy khó chơi hơn. Thông thường cần tạo 1 biến nhớ chung để lưu mức độ này, ví dụ **gamelevel**. Tham số này sẽ bắt đầu từ 0 hoặc 1 và sẽ được tăng dần theo thời gian hoặc điểm số, đánh giá người chơi.

(f) Tương tác đối kháng các người chơi.

Tương tác đối kháng là việc mô phỏng đối kháng trực tiếp giữa 2 nhân vật trong trò chơi, ví dụ chơi cờ, đấu súng, … Có 2 loại đối kháng sau:

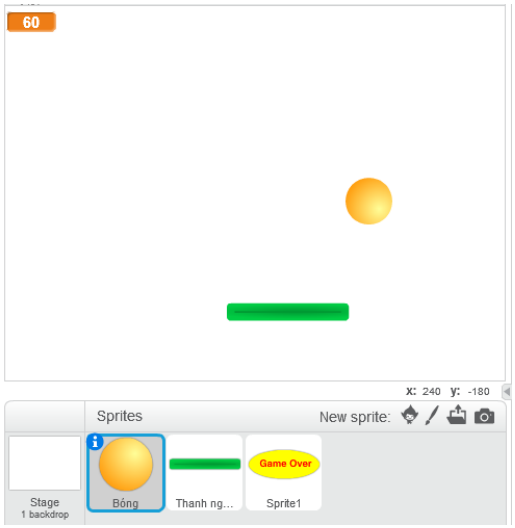
1) Đối kháng giữa 1 người chơi và 1 nhân vật do phần mềm tự mô phỏng. Tương tác đối kháng này thực chất là người đấu với máy tính.

2) Tương tác giữa 2 người chơi. Kỹ thuật này xử lý khó hơn trường hợp trên.

2. Kỹ thuật đếm ngược

Chúng ta cùng thực hiện kỹ thuật này thông qua trò chơi **Thả bóng**.

Tha bong extended.sb2

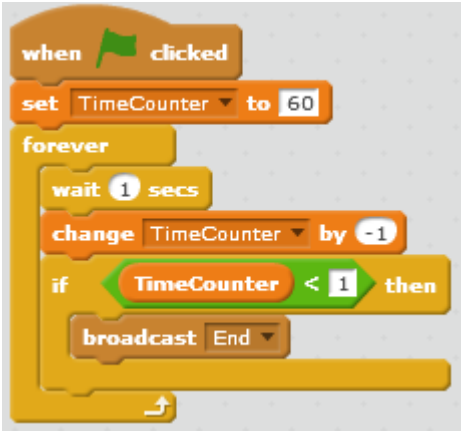


Cách chơi vẫn không có gì thay đổi. Bổ sung 1 đồng hồ thời gian để đếm ngược. Nếu sau 60 giây hoặc nếu người chơi để bóng rơi xuống phía dưới thì thua.

Khi thua, màn hình sẽ hiện dòng chữ "Game Over". Để làm việc này sẽ bổ sung thêm 1 nhân vật Game Over nữa như hình bên.

Cách xử lý kỹ thuật như sau:

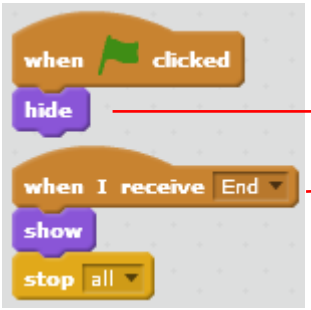
- Thiết lập 1 biến nhớ chung là **TimeCounter**, biến nhớ này sẽ đóng vai trò đồng hồ ngược. Giá trị ban đầu là 60, sau mỗi giây giá trị biến này giảm đi 1. Khi giảm về 0, từ sân khấu sẽ gửi đi thông điệp End để thông báo kết thúc.



Chương trình chạy trên nền sân khấu để điều khiển biến nhớ hệ thống **TimeCounter**.

Luôn kiểm tra, nếu biến nhớ TimeCounter < 1 thì ra thông điệp **End**.

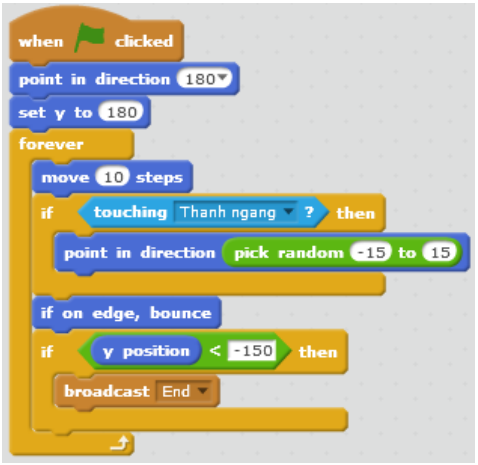
- Nhân vật **Game Over** sẽ xử lý việc này. Khi nhận được thông điệp End, nút Game Over sẽ hiển thị (nút này được ẩn đi khi bắt đầu chương trình) và ra lệnh dừng toàn bộ bằng lệnh **Stop All**.



Khi bắt đầu chương trình thì ẩn đi.

Khi nhận thông điệp **End** thì hiện và ra lệnh dừng toàn bộ chương trình.

- Chương trình của Bóng như sau, chú ý cách xử lý khi bóng rơi xuống phía dưới thì cũng gửi đi thông điệp End.



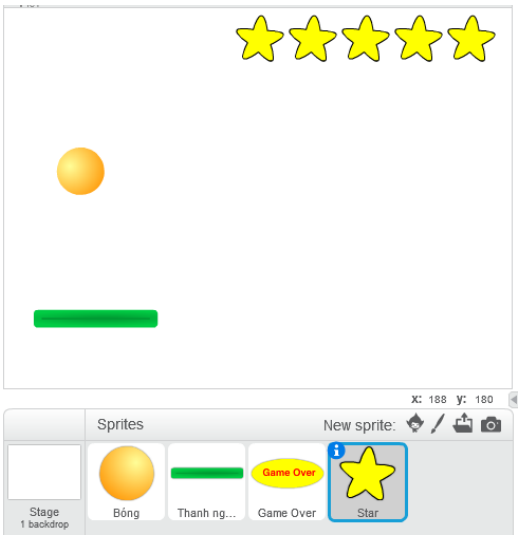
Nếu bóng rơi xuống phía dưới thì lập tức gửi thông điệp **End** để dừng chương trình.

3. Mô phỏng lượt chơi cho các nhân vật

Tha bong extended 2.sb2

Kỹ thuật mô phỏng lượt chơi có rất nhiều trong các phần mềm dạng trò chơi. Chúng ta sẽ mô phỏng chúng trong chương trình thả bóng sau đây.

Để mô phỏng lượt chơi, ngoài dùng 1 biến nhớ, chúng ta sẽ sử dụng công cụ Clone để thể hiện lượt chơi này. Chú ý đến nhân vật Star (ngôi sao) của chương trình.



Chương trình thả bóng quen thuộc. Mỗi người chơi có 5 lượt chơi. Mỗi lần bóng bị rơi xuống đất thì mất 1 lượt, khi hết tất cả các lượt thì sẽ bị thua.

Chương trình sẽ bổ sung thêm nhân vật Star (ngôi sao) để thể hiện lượt chơi còn lại ở phía trên màn hình. Trên màn hình luôn thể hiện số ngôi sao = số lượt chơi còn lại.

Chúng ta bắt đầu phân tích nhân vật ngôi sao của chương trình này. Việc thể hiện các ngôi sao từ đầu chương trình sẽ dùng tính năng phân thân, Clone của Scratch.

Như vậy khi bắt đầu chương trình, nhân vật ngôi sao sẽ phải phân thân thành 5 ngôi sao khác, được đánh số từ 1 đến 5 tính từ phải sang trái như hình sau.

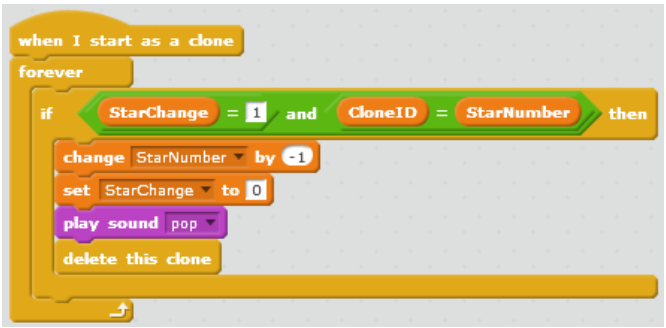
Để đánh số các clone, chúng ta dùng biến nhớ **CloneID - là biến riêng** của nhân vật này.

Biến nhớ dùng chung StarNumber dùng để tính số ngôi sao còn lại trên màn hình, đây chính là số lượt chơi còn lại của người chơi. Biến nhớ này sẽ được gán giá trị ban đầu = 5.

Đoạn chương trình sinh các phân thân và thể hiện các clone của ngôi sao trên màn hình, Mỗi ngôi sao mang 1 giá trị CloneID riêng của mình, đánh số từ 1 đến 5.

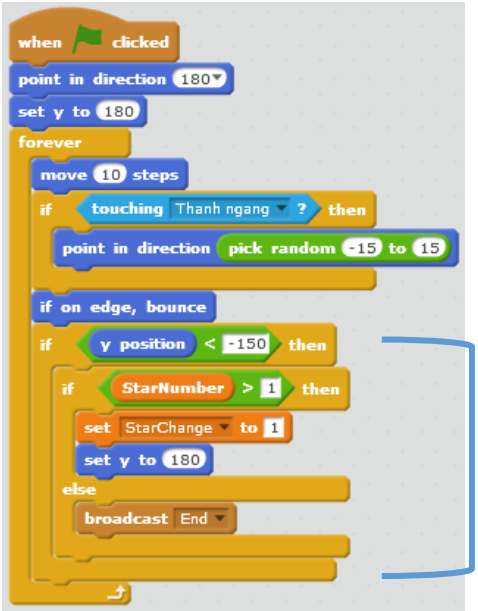
Đoạn chương trình sau mô tả hoạt động của bản thân các clone của ngôi sao. Đây là đoạn chương trình quan trọng nhất và là "key" của toàn bộ kỹ thuật dùng nhân vật thể hiện lượt chơi trên màn hình.

Chương trình có dùng 1 biến nhớ chung là **StarChange** có ý nghĩa như sau: biến nhớ này mô tả trạng thát chuyển số lượng của nhân vật ngôi sao. StarChange bình thường có giá trị = 0, nhưng tại thời điểm có biến động, giá trị của nó được thiết lập = 1 để thông báo đến thời điểm giá trị **StarNumber** sẽ giảm đi 1 đơn vị. Đoạn chương trình điều khiển clone này như sau:



Hàm kiểm tra **CloneID = StarNumber** để đảm bảo clone phía trái ngoài cùng sẽ bị xóa trước.

Còn đây là chương trình chính điều khiển bóng.



Đoạn chương trình kiểm tra khi bóng rơi xuống dưới. Nếu số lượt còn lại > 0 thì chuyển StarChage = 1 để giảm số lượt, ngược lại nếu đã hết lượt thì thông báo End để kết thúc.

4. Kỹ thuật tính điểm số cho người chơi

Tính điểm số cho người chơi là một trong những đặc tính phổ biết nhất cho tất cả các phần mềm trò chơi vì nó sẽ tạo ra sự kích thích, hứng thú của người chơi. Trong hoạt động này chúng ta sẽ minh họa kỹ thuật này thông qua chương trình **thả bóng** quen thuộc.

Tha bong extended 4.sb2

Thiết lập 1 biến nhớ **Diem_so** để lưu trữ điểm số cho người chơi. Khi bắt đầu chương trình, giá trị này sẽ được gán = 0. Biến nhớ sẽ được hiển thị online ngay trên màn hình.

Để cho chương trình hấp dẫn sẽ bổ sung thêm tính năng thưởng lượt chơi.

Cách tính điểm số qui định như sau:

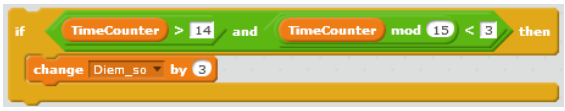
- Mỗi khi người dùng làm bật quả bóng lên trên bởi thanh ngang thì được cộng thêm 5 điểm.

- Điểm thưởng thêm: cứ sau mỗi 15 giây chơi, người dùng được tặng thêm 3 điểm.

Các đoạn chương trình tương ứng như sau:

Khi bóng chạm được thanh ngang thì bật lên và tăng điểm số thêm 5.

Mỗi khi thời gian chơi của người chơi vượt qua khoảng 15 giây thì điểm sẽ được tăng lên 3.

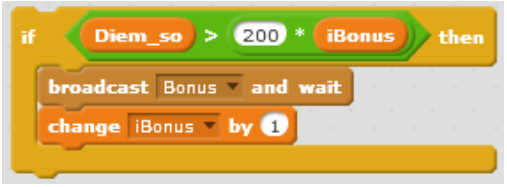


Thưởng lượt chơi. Qui định như sau: mỗi khi điểm số của người chơi vượt quá 200, 400, … (bội của 200) thì được thưởng thêm 1 lượt chơi. Trên màn hình sẽ xuất hiện thêm 1 ngôi sao.

Để thực hiện được yêu cầu trên chúng ta thiết lập thêm 1 biến nhớ để điều khiển việc thưởng lượt chơi, biến có tên **iBonus** và được gán giá trị ban đầu = 0. Thuật toán như sau:

Mỗi khi kiểm tra thấy Diem_so > 200*iBonus thì gửi thông điệp **Bonus** để tăng lượt và tăng **iBonus** lên 1 đơn vị. Nhân vật Ngôi sao sẽ nhận thông điệp **Bonus** và xử lý để bổ sung thêm lượt chơi.

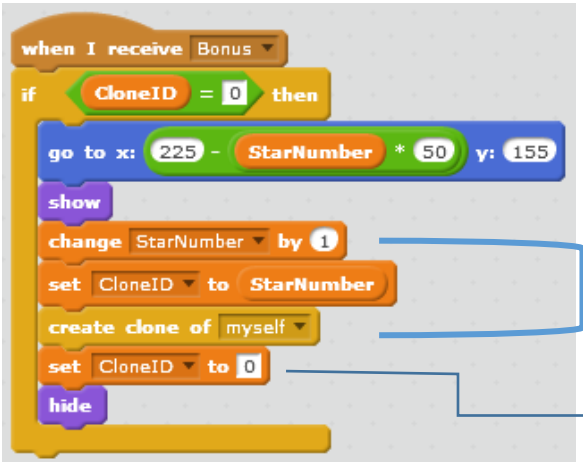
Đoạn chương trình của sân khấu điều khiển điểm số như sau:



Kiểm tra nếu Diem_so > 200*iBonus thì gửi thông điệp Bonus, sau đó tăng iBonus lên 1 đơn vị để chờ lần gửi Bonus tiếp theo.

Việc xử lý thông điệp Bonus của nhân vật Ngôi sao sẽ phức tạp hơn vì hiện tại trên màn hình có cả nhân vật Ngôi sao gốc và các Clone của mình. Các Clone cũng nhận được thông điệp Bonus. Do vậy cần lập trình để phân biệt được nhân vật gốc và các Clone. Chỉ có nhân vật gốc mới nhận Bonus và xử lý. Chúng ta phải sử dụng lại biến riêng CloneID để phân biệt nhân vật gốc và Clone của ngôi sao. Nhân vật gốc sẽ có CloneID = 0.

Đoạn chương trình xử lý thông điệp Bonus như sau:



Kiểm tra nhân vật gốc mới được xử lý thông điệp này.

Chuyển đến vị trí cần thiết.

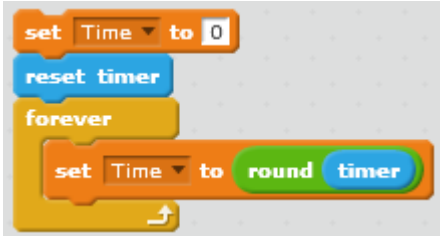
Tăng số lượng sao, đặt lại CloneID và khởi tạo Clone mới (tăng lượt).

Đặt lại ClodeID = 0 cho nhân vật gốc.

5. Thể hiện số và đếm số

Hiện số, đếm số theo thời gian là các đặc thù thường xuyên của các trò chơi hay phần mềm nói chung. Cách đơn giản nhất là thiết lập 1 biến nhớ và cho biến nhớ hiển thị trên màn hình.

Ví dụ sử dụng biến nhớ **Time**, đặt chế độ hiển thị biến nhớ này trên màn hình. Chúng ta có 2 cách sau để cho biến nhớ tự động đếm theo thời gian là giây.

Cách 1. Sử dụng hàm hệ thống **timer**, luôn gán **Time** cho **timer**.	**Cách 2.** Dùng vòng lặp vô hạn, sau mỗi 1 giây nghỉ bằng lệnh **wait** thì tăng **Time** lên 1 đơn vị.
	

Biến nhớ hệ thống **timer** cũng có chế độ hiển thị online  . Em hãy so sánh cách hiển thị của biến nhớ này với hiển thị biến nhớ **Time** do người dùng tạo ra.

Cách thể hiện trên có 1 nhược điểm là chữ số quá bé, không gây ấn tượng. Trong các phần mềm hoặc trò chơi cần các kỹ thuật để thể hiện số to hơn, rõ ràng hơn, chúng ta sẽ cùng nhau tìm hiểu các bài toán này.

Kỹ thuật thể hiện số bằng hình ảnh trên màn hình.

Show Numbers.sb2

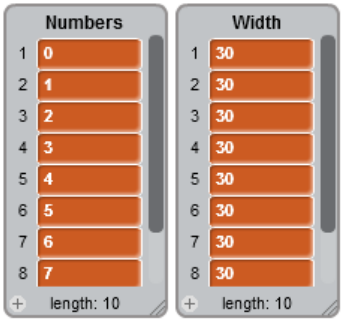
(a) Thể hiện số bằng hình ảnh nhân vật

Chúng ta sẽ cùng xây dựng chương trình và thuật toán thể hiện số bằng hình ảnh lớn trên sân khấu. Kỹ thuật này rất hay được sử dụng trong các trò chơi và phần mềm.

Ý tưởng của kỹ thuật này là sử dụng lệnh **stamp** của Scratch để hiển thị chữ số lớn bằng hình ảnh.

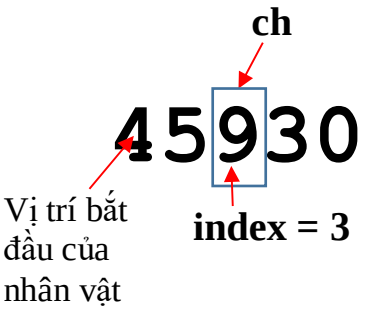
Thiết lập 1 nhân vật có tên Number bao gồm đúng 10 trang phục được đánh số từ 1 đến 10, trang phục 1 là số 0, trang phục 2 là số 1, …., trang phục 10 là số 9. Thuật toán sẽ dựa trên các trang phục này để hiển thị chữ số trên màn hình.

Giá trị 2 biến mảng chính liên quan đến thuật toán này (Numbers và Width) như sau:



- Bảng **Numbers** chứa 10 phần tử là 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Như vậy phần tử thứ **k** chính là chữ số **k-1**, tương ứng trang phục thứ **k** của nhân vật.

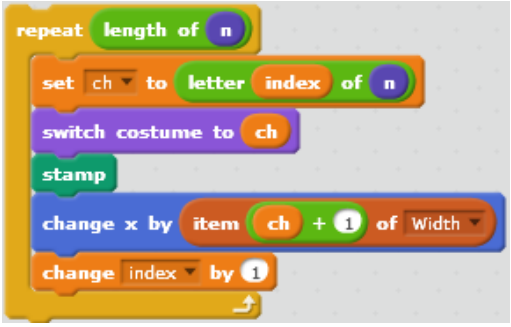
- Bảng **Width** lưu độ rộng của các trang phục tương ứng. Phần tử thứ **k** chính là độ rộng của trang phục số **k-1**.



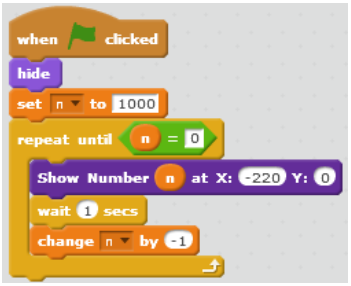
Thuật toán hiển thị số bằng hình ảnh:

Trong 1 vòng lặp độ dài length(n), bắt đầu từ vị trí index=1, lấy ra chữ số ch. Chuyển trang phục cho nhân vật, thực hiện lệnh **stamp**, sau đó chuyển sang phải theo độ dài bằng độ rộng của số ch+1 lấy từ bảng **Width**.

Thuật toán trên được mô tả bằng lệnh Scratch như sau:



Ví dụ đoạn chương trình hiển thị số trên màn hình đếm lùi từ 1000 như sau:



Em hãy hoàn thiện thủ tục **Show Number** thể hiện 1 số bắt đầu từ 1 vị trí cho trước trên màn hình.

(b) Kỹ thuật thể hiện đếm số tự động bằng hình ảnh

Kỹ thuật thể hiện việc đếm bằng hình ảnh trên màn hình.

Thuật toán sử dụng trong phần trên có 1 nhược điểm là mỗi lần thể hiện phải xóa tất cả để vẽ lại từ chữ số của số cần thể hiện. Chúng ta sẽ làm tốt hơn kỹ thuật trên đối với bài toán thể hiện đếm (tăng dần hoặc giảm dần) trên màn hình.

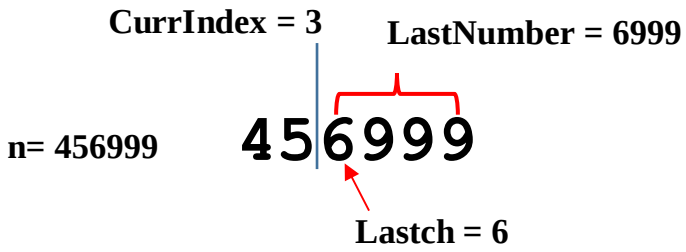
Ý tưởng chính của thuật toán này là sau mỗi lần tăng hoặc giảm số, không cần vẽ lại toàn bộ các chữ số, mà chỉ cần vẽ lại những chữ số cần vẽ lại.

Chúng ta cùng xét 1 ví dụ thể hiện phép đếm tăng số, bắt đầu từ 1 giá trị nào đó. Quá trình này được xét kỹ hơn khi chuyển thể hiện số **n** sang **n+1**.

Ví dụ với số n = 3587 thì số tiếp theo chỉ cần thay 7 → 8.

Ví dụ với số n = 456999 thì số cần thể hiện tiếp theo là 457000.

Rõ ràng cần tìm ra chữ số đầu tiên < 9 tính từ bên phải của số n.



Thuật toán được mô tả như sau:

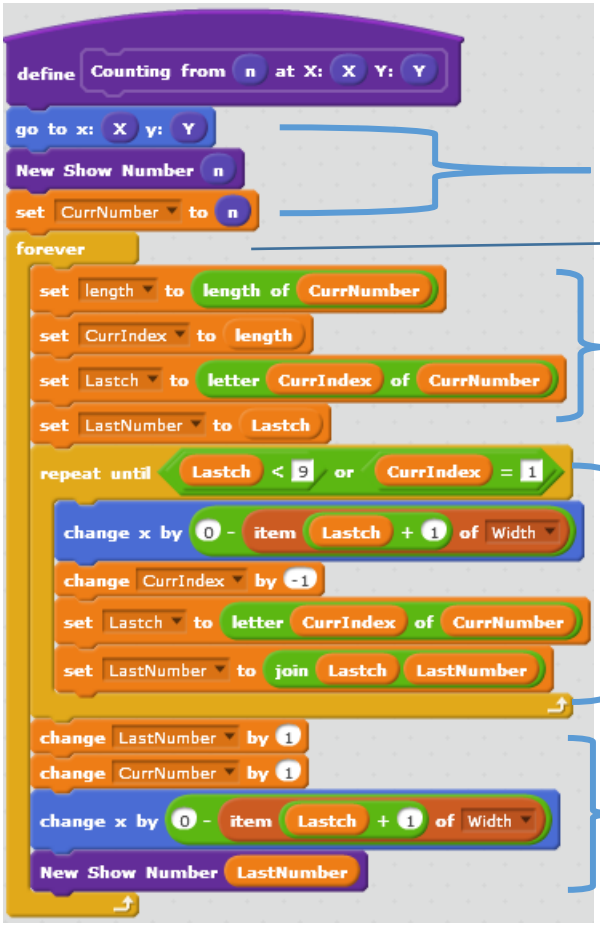
Thiết lập 3 biến nhớ: **CurrIndex**, **LastNumber** và **Lastch**.

> Bắt đầu từ vị trí cuối cùng của số n, thiết lập CurrIndex = length(n), Lastch = LastNumber = chữ số cuối cùng của **n**.
>
> Đi lùi về bên trái cho đến khi gặp chữ số đầu tiên < 9 hoặc CurrIndex = 1.
>
> Tính toán lại giá trị LastNumber.
>
> LastNumber có độ dài = Length(n) - CurrIndex + 1. (Nếu CurrIndex = 1 thì LastNumber = n).
>
> Tăng LastNumber và n lên 1 đơn vị.
>
> Thực hiện việc thể hiện lại số LastNumber bắt đầu từ vị trí hiện thời.

Thủ tục chính của thuật toán trên là **Counting from <n> at <X> <Y>** sẽ hiện máy đếm số tăng liên tục bắt đầu từ n.

Trong thủ tục này có sử dụng 1 thủ tục khác là **New Show Number <n>**. Thủ tục này có chức năng thể hiện 1 số <n> tính từ vị trí hiện thời của nhân vật, nhưng không xóa màn hình trước khi thể hiện chữ. Thủ tục này dùng để thể hiện liên tục số **LastNumber** trong thuật toán trên.

Chúng ta cùng phân tích thủ tục đếm số tăng này.



Thể hiện số gốc n trên màn hình, thiết lập CurrNumber = n.

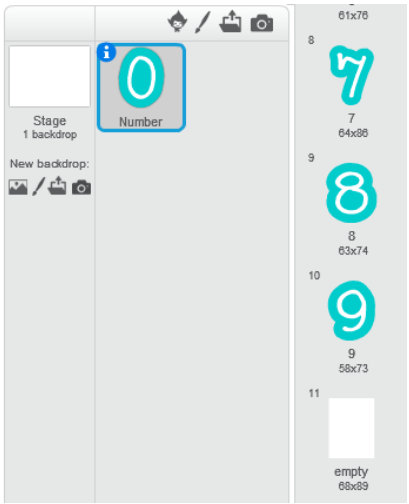
Bắt đầu vòng lặp chính tăng n để đếm

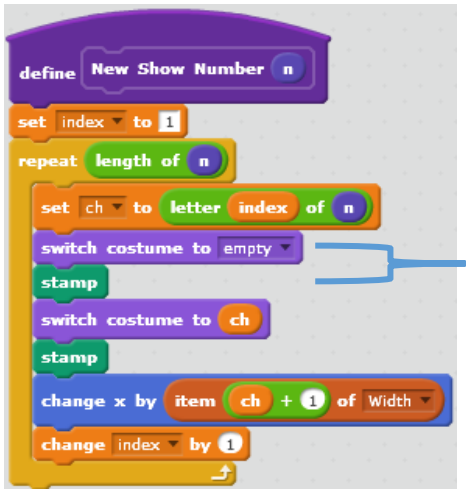
Thiết lập các biến chính vào đầu mỗi vòng lặp:
lengh, Lastch, CurrIndex, LastNumber.

Vòng lặp lùi CurrIndex về trái để tìm chữ số đầu tiên < 9. Mỗi lần lùi sẽ đồng thời lùi vị trí nhân vật số sang trái theo độ dài 1 chữ số.

Lùi vị trí nhân vật sang trái thêm 1 chữ số nữa, tăng LastNumber và CurrNumber lên 1 đơn vị. Thể hiện số LastNumber từ vị trí hiện thời.

Bây giờ chúng ta quay lại thủ tục **New Show Number**, chỉ ra 1 vị trí thay đổi duy nhất nhưng quan trọng của thủ tục này so với **Show Number**. Bổ sung vào các trang phục số 1 trang phục **empty** nữa để xóa 1 chữ số đã có trên màn hình.





Thủ tục **New Show Number** chỉ có 1 sự khác biệt duy nhất là trước khi vẽ 1 chữ số thì xóa vết của chữ số trước đó. Việc xóa này được thực hiện bởi trang phục **empty**.

Trên đây là bài toán thể hiện phép đếm số tăng dần. Thuật toán thể hiện phép đếm giảm dần hoàn toàn tương tự.

Bài tập: thiết lập thuật toán tương tự để xử lý bài toán đếm số giảm dần. Thủ tục tương ứng sẽ có tên **Counting down <n> at <X>, <Y>**.

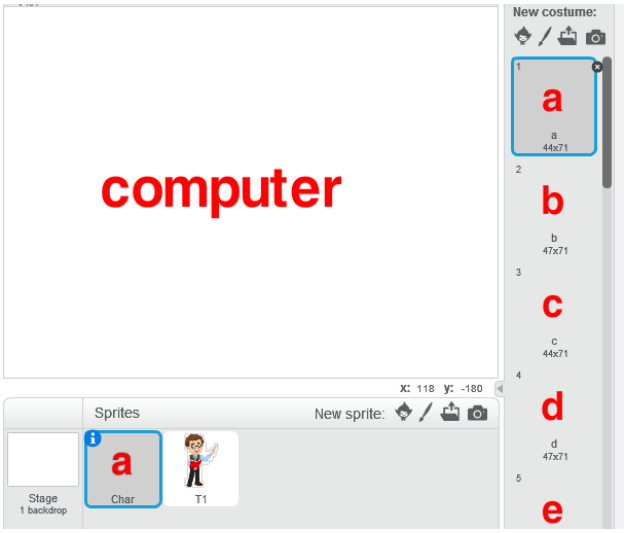
Kỹ thuật thể hiện chữ bằng hình ảnh.

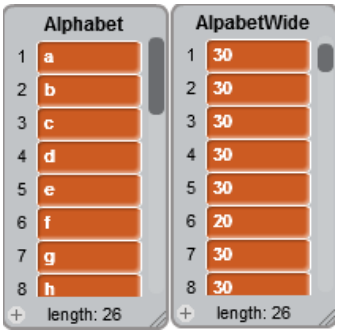
6. Thể hiện chữ, tên người, nhân vật

Bài toán thể hiện chữ trong các phần mềm, trò chơi cũng tương tự như việc thể hiện số. Chúng ta có thể dùng biến nhớ thể hiện ngay trên màn hình.

Để thể hiện các chữ bằng hình ảnh trên màn hình, chúng ta sẽ thiết lập nhân vật **Char** có các trang phục chính là dãy các chữ cái của bảng chữ cái ngôn ngữ đang sử dụng. Về lý thuyết thủ tục thể hiện Show Word có thể làm tương tự như Show Number trong ví dụ trên. Nhưng chúng ta sẽ đi theo cách khác trong hoạt động này.

Minh họa kỹ thuật hiển thị từ, tên người, tên sự vật được thể hiện trong chương trình sau:





Thiết lập nhân vật có tên **Char** với 26 trang phục tương đương bảng chữ cái tiếng Anh.

- Nhân vật chính là bộ chữ cái đầy đủ, ví dụ bộ 26 chữ cái tiếng Anh. Nhân vật trong chương trình của chúng ta tên là Char và có 26 bộ trang phục theo bảng chữ cái.

- Bộ dữ liệu chính bao gồm 2 bảng **Alphabet** và **AlphabetWide**. **Alphabet** là dãy 26 chữ cái tương ứng đúng với dãy các trang phục của Char. **AlphabetWide** là dãy các chiều rộng của các trang phục của Char trên sân khấu.

- Để thiết lập thuật toán theo yêu cầu chúng ta cần thêm 2 bảng nữa, **WList** và **Wkc** để lưu kết quả của từ, chữ cần thể hiện trên màn hình.

Hai bảng **Wlist** và **Wkc** sẽ lưu kết quả trung gian liên quan đến từ cần thể hiện trên màn hình. **WList** lưu dãy các chữ cái của từ và **Wkc** là thông tin độ rộng tương ứng của chữ cái này.

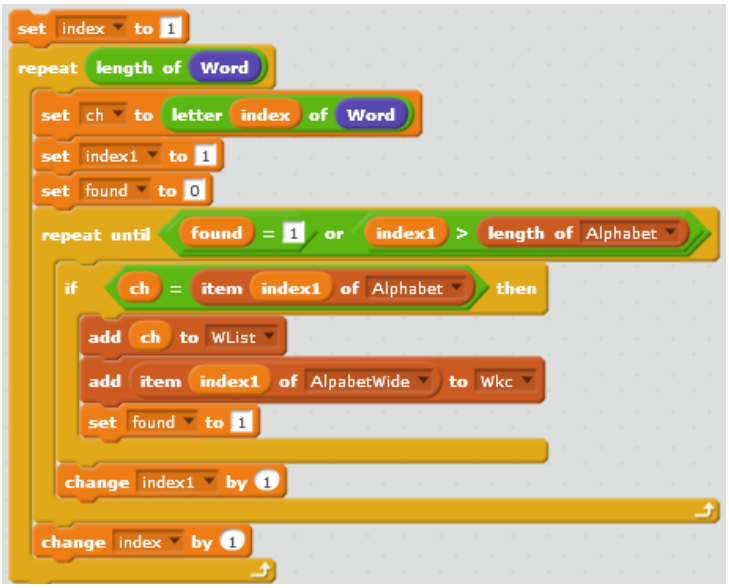
Giả sử từ cần thể hiện bằng hình ảnh trên màn hình là Word, thuật toán chính được mô tả như sau.

Bước 1 (Init). Phân tích và tách từng chữ cái của từ Word, đưa vào bảng WList, chỉ lấy các chữ cái có trong bảng Alphabet tương ứng. Bảng Wkc lưu thông tin tương ứng lấy từ bảng AlphabetWide.

Bước 2 (Show Word). Lần lượt lấy các chữ cái từ bảng WList, chuyển trang phục cho chữ cái này, dùng lệnh stamp để vẽ chữ cái trên màn hình, sau đó chuyển vị trí nhân vật sang phải theo 1 khoảng cách bằng đúng độ rộng của chữ cái này. Quá trình này lặp lại cho đến khi đi hết bảng WList.

Bước 1 được gọi là quá trình Init (chuẩn bị), bước 2 là quá trình thực sự thể hiện từ trên màn hình.

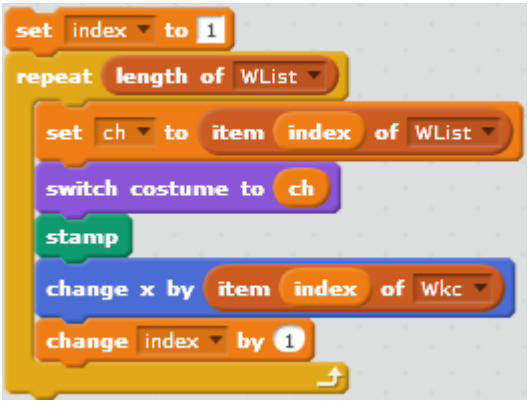
Phần lõi của bước 1, Init, được thể hiện bằng lệnh Scratch như sau.



Vòng lặp ngoài lấy từng chữ **ch** của Word

Kiểm tra xem ch có nằm trong bảng chữ cái **Alphabet** không, nếu có bổ sung chữ này vào **WList** và lấy độ rộng tương ứng đưa vào **Wkc**.

Phần lõi của bước 2 được thể hiện như sau:



Lấy từng ký tự trong bảng **WList**, chuyển trang phục sang ký tự này, thực hiện lệnh **stamp**, sau đó chuyển tọa độ X sang phải 1 giá trị bằng độ dài trong **Wkc** tương ứng.

2 thủ tục cần thiết kế cho kỹ thuật trên là:

Em hãy hoàn thiện 2 thủ tục Init và Show Word cho thuật toán trên.

random algorithms

7. Kỹ thuật sinh ngẫu nhiên trong các trò chơi và phần mềm

Kỹ thuật sinh ngẫu nhiên rất hay được sử dụng trong tất cả các phần mềm, trò chơi, ví dụ cần sinh 1 bộ câu hỏi trắc nghiệm từ 1 ngân hàng câu hỏi cho trước. Có rất nhiều bài toán cụ thể liên quan đến kỹ thuật này, chúng ta cùng tìm hiểu 1 số thuật toán như vậy.

(a) Thuật toán sinh ngẫu nhiên 4 số trong 1 dãy số dạng List.

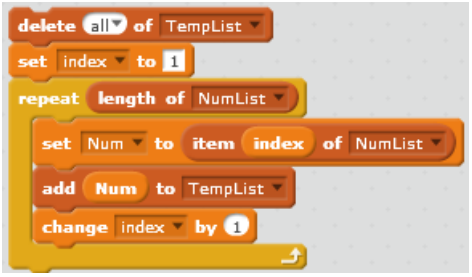
Xét bài toán sau: cho trước 1 dãy số **NumList**. Dãy này được minh họa bằng biểu diễn: a_1, a_2, ….., a_N.

Cần lấy ra ngẫu nhiên 4 phần tử khác nhau từ dãy này, kết quả được đưa vào 4 biến nhớ pA, pB, pC, pD và thể hiện trên màn hình. Chú ý dãy ban đầu NumList cần được bảo toàn sau thuật toán.

Phân tích.

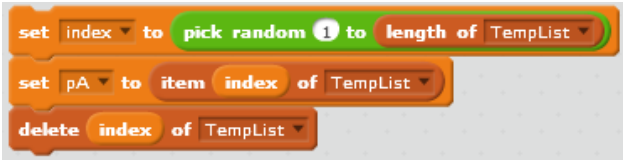
Ý tưởng thực hiện bài toán này rất đơn giản, chỉ cần lần lượt lấy ra 4 phần tử ngẫu nhiên của dãy NumList, sau mỗi lần lấy ra thì xóa phần tử gốc khỏi dãy ban đầu. Nhưng bài toán yêu cầu bảo toàn dãy gốc nên cần thực hiện trên 1 dãy tạm. Như vậy cần thiết lập thêm 1 biến nhớ dạng list nữa là TempList, cần thực hiện việc chuyển toàn bộ dữ liệu từ NumList sang TempList và tiến hành thuật toán sinh dữ liệu từ TempList này.

Đây là đoạn chương trình tạo dữ liệu ban đầu cho TempList.



Thuật toán sao chép toàn bộ dữ liệu từ dãy **NumList** sang dãy **TempList**.

Đoạn chương trình sau thực hiện việc sinh ngẫu nhiên phần tử từ dãy TempList, đưa vào pA và xóa phần tử này từ dãy TempList.



Tương tự thực hiện như vậy với 3 phương án còn lại pB, pC, pD.

Bài tập: em hãy viết hoàn chỉnh thủ tục mô tả thuật toán trên.

(b) Thuật toán sinh ngẫu nhiên 1 xâu con 4 ký tự của 1 xâu bất kỳ

Bài toán: cho trước 1 xâu ký tự bất kỳ **Str**. Cần lấy ra ngẫu nhiên 4 ký tự khác nhau từ xâu này, kết quả được đưa vào 4 biến nhớ pA, pB, pC, pD và thể hiện trên màn hình.

Phân tích.

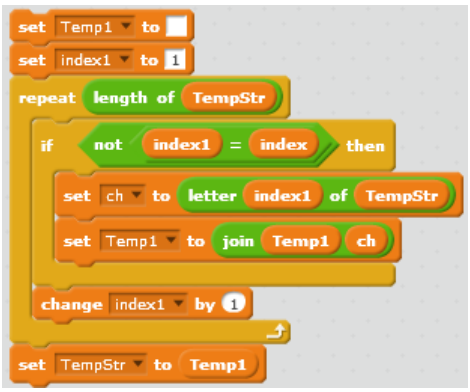
Bài toán này gần tương tự hoàn toàn bài toán trên, điểm khác duy nhất là dữ liệu gốc không là dãy, mà là 1 xâu ký tự. Có 2 cách thực hiện như sau:

Cách 1: tạo 1 biến xâu tạm thời TempStr, sao chép Str sang TempStr và thực hiện thuật toán lấy 4 phần tử ngẫu nhiên từ TempStr, sau mỗi lần lấy ra thì xóa ký tự ở xâu gốc.

Cách 2: tạo 1 dãy TempList, lấy từng ký tự từ xâu Str chuyển vào TempList, sau đó lấy ra 4 phần tử ngẫu nhiên tương tự thuật toán (a) đã nêu ở trên.

Với cách 1 của thuật toán này, sau đây là đoạn chương trình sao chép Str sang TempStr và lấy đi 1 phần tử ngẫu nhiên đầu tiên gán vào pA.

Tiếp theo cần xóa đi ký tự với chỉ số index của xâu TempStr. Vì Scratch không có lệnh xóa 1 ký tự trong xâu, nên chúng ta phải thực hiện "mẹo" bằng cách chuyển toàn bộ TempStr sang 1 xâu trung gian nữa, Temp1, sau đó cần gán ngược lại vào xâu gốc TempStr. Chú ý khi chuyển từ TempStr → Temp1 thì bỏ qua ký tự với chỉ số index. Đoạn chương trình trên như sau:



Thuật toán xóa 1 ký tự chỉ số index trong 1 xâu cho trước TempStr.

Bài tập: hoàn thiện nốt thuật toán này.

(c) Thuật toán sinh hoán vị ngẫu nhiên của 1 xâu ký tự

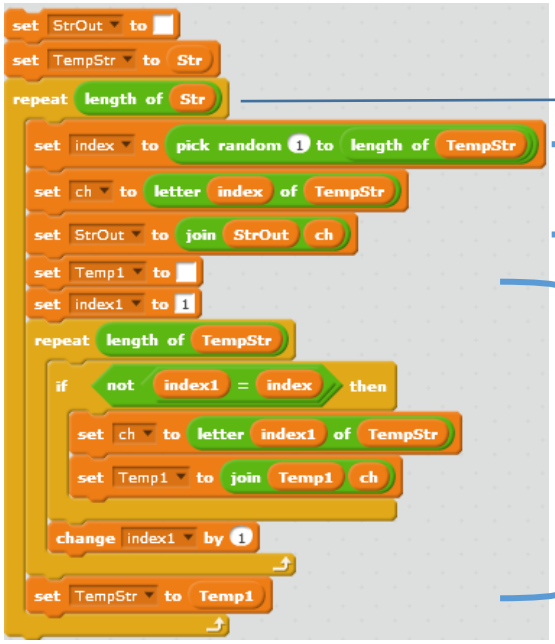
Bài toán: cho trước 1 xâu ký tự bất kỳ **Str**. Cần sinh ra 1 hoán vị ngẫu nhiên của xâu này, kết quả đưa vào biến **StrOut**.

Phân tích.

Nếu để ý 1 chút, chúng ta sẽ thấy bài toán này suy từ bài toán sinh 4 phần tử ngẫu nhiên ở trên. Chúng ta sử dụng 1 xâu đệm có tên TempStr. Thuật toán được mô tả như sau:

```
Gán TempStr = Str. Gán StrOut = rỗng.
Thực hiện vòng lặp độ dài = length(Str)
     Lấy ra 1 phần tử ngẫu nhiên của TempStr.
     Đưa phần tử này vào cuối của StrOut.
     Xóa phần tử này khỏi TempStr.
     (cách xóa: sao chép tất cả các ký tự từ TempStr
     sang Temp1 trừ ra phần tử trên, sau đó gán
     ngược lại vào TempStr).
```

Đoạn chương trình trong Scratch như sau:



Vòng lặp chính.

Sinh giá trị ngẫu nhiên index, lấy ra phần tử ch và đưa vào cuối của StrOut.

Xóa phần tử của TempStr tại vị trí index. Cách thực hiện: sao chép tất cả các ký tự từ TempStr sang Temp1 trừ ra phần từ có chỉ số index. Sau đó gán lại TempStr = Temp1.

Em hãy viết 1 thủ tục hoàn thiện thuật toán trên.

(d) Thuật toán sinh hoán vị ngẫu nhiên của 1 dãy số

Bài toán: cho trước 1 dãy số (hoặc dãy xâu ký tự) NList (WList). Cần sinh ra 1 hoán vị ngẫu nhiên của dãy số (xâu) này, kết quả đưa vào dãy (xâu) NListOut (WListOut).

Phân tích.

Chúng ta sẽ thiết kế thuật toán trên dãy số NList, hoàn toàn tương tự với dãy xâu WList. Thuật toán này khá đơn giản như sau.

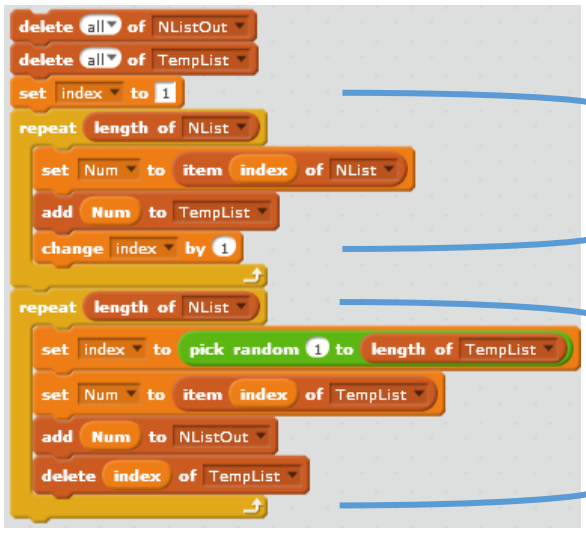
Thiết lập 1 dãy tạm TempList. Sao chép toàn bộ NList sang TempList.

Thiết lập vòng lặp độ dài = length(NList)

Lấy 1 phần tử ngẫu nhiên của TempList, đưa vào cuối của NListOut.

Xóa phần tử này khỏi dãy TempList.

Mô tả thuật toán trên bằng lệnh Scratch như sau:



Sao chép toàn bộ dãy **NList** sang dãy **TempList**.

Sinh 1 hoán vị ngẫu nhiên của **TempList** và đưa kết quả vào dãy **NListOut**.

8. Thuật toán sinh ngẫu nhiên 1 bộ đáp án, trong đó có 1 đáp án đúng

Trong hoạt động này chúng ta sẽ làm quen với 1 số các thuật toán hay dùng khi sinh ngẫu nhiên các câu hỏi trắc nghiệm trong các phần mềm giáo dục. Chúng ta nhớ lại rằng câu hỏi trắc nghiệm luôn bao gồm 4 đáp án (có thể nhiều hơn), trong đó có 1 đáp án đúng. Trên thực tế có rất nhiều bài toán như vậy, chúng ta sẽ chỉ tìm hiểu 1 vài trong chúng.

Để mô phỏng các bài toán sinh ngẫu nhiên câu hỏi trắc nghiệm chúng ta dùng các bộ dữ liệu sau:

(a) Bộ các biến nhớ độc lập để lưu các phương án pA, pB, pC, pD và bộ 4 trạng thái để mô tả phương án đúng sai: pAstatus, pBstatus, pCstatus, pDstatus. Các giá trị status có ý nghĩa = 0 nếu phương án sai, = 1 nếu phương án đúng. Một phương án khác là chỉ cần 1 biến nhớ pCorrect có ý nghĩa là chỉ số của phương án đúng trong số các phương án A, B, C, D.

(b) 2 mảng dữ liệu có độ dài 4 phần tử: pList và statusList.

pList[k], k=1, 2, 3, 4 là các phương án A, B, C, D.

sList[k], k=1, 2, 3, 4 là các trạng thái = 0, nếu sai, = 1, nếu đúng.

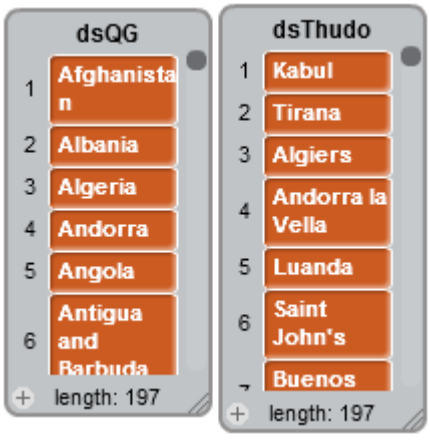
Như vậy bộ dữ liệu sẽ cần được cập nhật đồng thời cho các mảng pList, sList và các biến nhớ pA, pB, pC, pD, pAstatus, pBstatus, pCstatus, pDstatus.

Xét các bài toán cụ thể sau:

Giả sử cho trước 1 dãy số (hoặc xâu ký tự). Cần lấy ra ngẫu nhiên 4 giá trị của dãy trên dùng làm 4 phương án, sau đó lại lựa chọn ngẫu nhiên 1 trong số các giá trị này để làm phương án đúng.

Chúng ta sẽ cùng thực hiện bài toán trên thông qua 1 ví dụ cụ thể sau.

Giả sử có 2 mảng dữ liệu: **dsQG** và **dsThudo** như hình dưới đây.



Cho trước 2 mảng xâu ký tự:

dsQG: danh sách các quốc gia trên thế giới.

dsThudo: danh sách tên thủ đô tương ứng.

Cần sinh ngẫu nhiên các câu hỏi trắc nghiệm như sau, ví dụ:

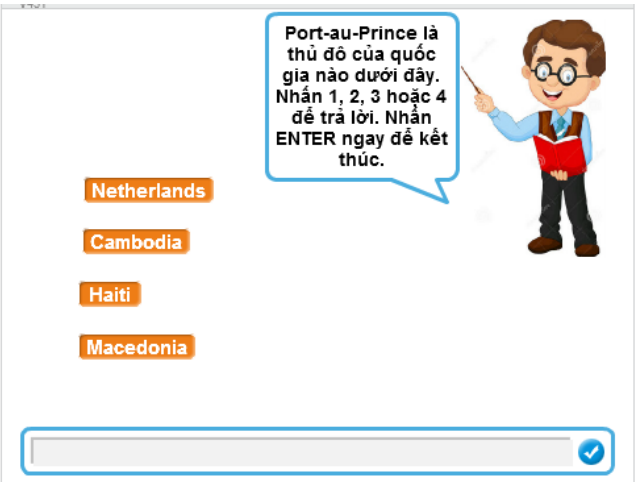
> **Tirana** là thủ đô của quốc gia nào dưới đây:
>
> A. **Angola** B. **Albani** C. **Vietnam** D. **Japan** (đáp án đúng: B)

Thuật toán sinh bộ dữ liệu đơn giản như sau:

Lần lượt sinh 4 giá trị ngẫu nhiên lấy từ bảng dsQG (đồng thời dsThudo) và đưa vào các biến pA, pB, pC, pD, sau đó lựa chọn 1 giá trị ngẫu nhiên trong 4 số này lấy làm phương án đúng để sinh câu hỏi dạng như trên.

Minh họa thuật toán trên bằng chương trình sau dùng để kiểm tra kiến thức môn Địa lý cho học sinh phổ thông.

Random multiple choice.sb2



Chương trình sẽ sinh tự động các câu hỏi trắc nghiệm theo mẫu trên và thể hiện như hình bên.

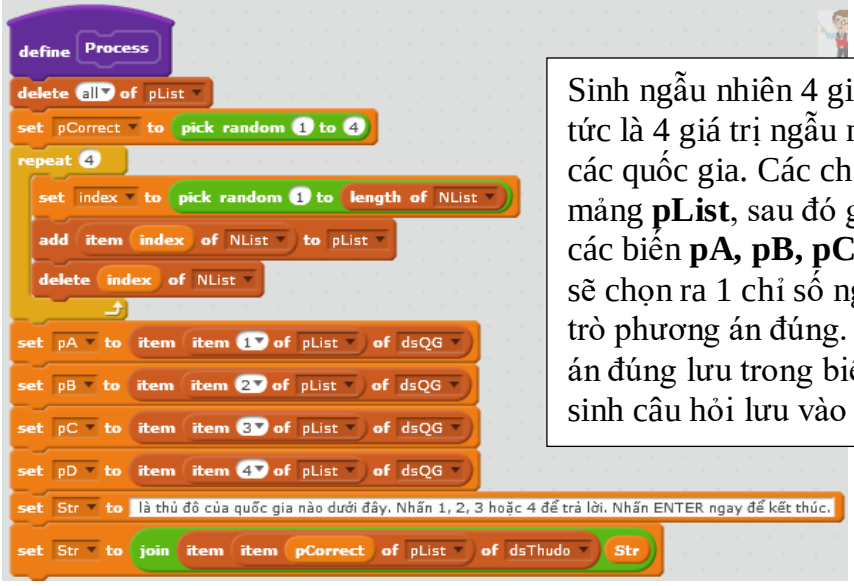
Người dùng sẽ trả lời bằng cách nhập đáp án bằng số như 1, 2, 3, 4. Ngay sau khi nhập đáp án, chương trình sẽ thông báo đúng, sai và sinh tiếp câu hỏi.

Chương trình sẽ dừng khi người chơi chỉ nhấn Enter mà không nhập số.

Thủ tục **Init** có nhiệm vụ thiết lập bộ dữ liệu đệm của chương trình.

Bộ dữ liệu đệm **NList** được khởi tạo mỗi lần cần sinh câu hỏi trắc nghiệm. Bộ dữ liệu này bao gồm 1 dãy số tự nhiên 1, 2, 3, … có độ dài đúng bằng độ dài của mảng **dsQG**.

Thủ tục **Process** dùng để khởi tạo ngẫu nhiên 1 bộ dữ liệu câu hỏi trắc nghiệm.



Sinh ngẫu nhiên 4 giá trị từ dãy NList, tức là 4 giá trị ngẫu nhiên từ danh sách các quốc gia. Các chỉ số này sẽ lưu trong mảng **pList**, sau đó gán tên quốc gia cho các biến **pA, pB, pC, pD**. Trong số này sẽ chọn ra 1 chỉ số ngẫu nhiên đóng vai trò phương án đúng. Chỉ số của phương án đúng lưu trong biến **pCorrect**. Sau đó sinh câu hỏi lưu vào xâu **Str**.

Chương trình chính sẽ có dạng đơn giản như sau:

Trong vòng lặp vô hạn, sau khi thực hiện 2 thủ tục Init, Preocess, sẽ thực hiện yêu cầu người sử dụng trả lời câu hỏi trắc nghiệm vừa được tạo ra.

Có thể tóm tắt thuật toán sinh ngẫu nhiên câu hỏi trắc nghiệm như sau:

- Sinh ngẫu nhiên 4 giá trị chỉ số khác nhau từ 1 đến độ dài dsQG. Các giá trị này sẽ lưu vào 4 phần tử của dãy pList.

- Sinh ngẫu nhiên giá trị pCorrect từ 1 đến 4 để lưu phương án đúng.

- Gán tên các quốc gia đã chọn vào các biến pA, pB, pC, pD để hiện trên màn hình.

- Sinh câu hỏi trắc nghiệm, lấy tên thủ đô của đáp án đúng lồng ghép trong nội dung câu hỏi.

- Lặp lại quá trình trên cho mỗi lần sinh câu hỏi.

Em hãy hoàn thiện chương trình trên.

9. Ví dụ: trò chơi điều khiển bóng

Chúng ta sẽ hoàn thiện phần mềm trò chơi **thả bóng** đã được nêu ra trong các hoạt động của bài học này, bổ sung thêm các yêu cầu mới.

Tha bong extended 5.sb2

Các nhân vật chính bao gồm Bóng, Thanh ngang, Star, Number và nút Game Over.

Mỗi lần bóng chạm thanh ngang được tăng 5 điểm. Nếu thời gian chơi vượt qua mỗi số là bội của 15 thì tăng điểm 3 điểm. Nếu điểm số vượt ngưỡng 200, 400, 600, … thì thưởng lượt chơi. Điểm số hiện to trên màn hình.

Nếu bóng rơi xuống phía dưới thì mất 1 lượt. Nếu hết lượt thì xuất hiện Game Over và trò chơi kết thúc.

Chương trình được thiết kế với các nhân vật và dữ liệu chính sau:

- Nhân vật chính: Bóng; Thanh ngang, Ngôi sao (Star), Chữ số (Number) và nút Game Over.

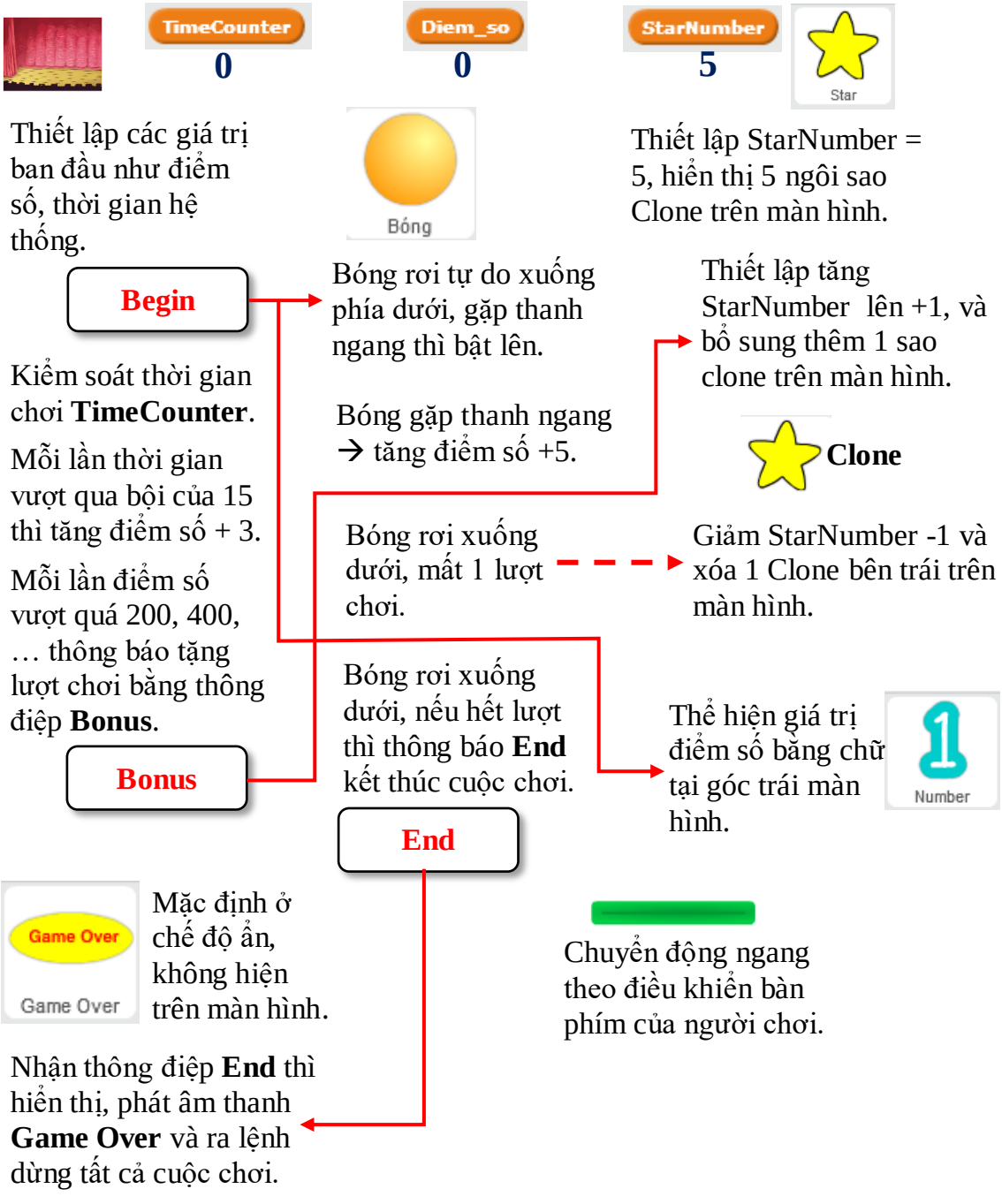
- Các biến nhớ chính:

TimeCounter: Tổng thời gian chơi. Chơi càng lâu sẽ được thưởng điểm và thưởng lượt chơi.

Diem_so: Điểm số của người chơi.

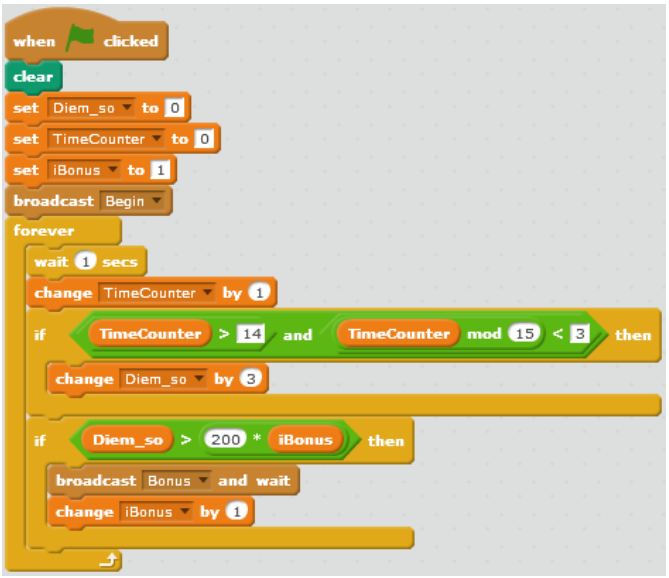
StarNumber: Số lượt chơi thể hiện bằng các ngôi sao trên màn hình.

Sơ đồ hoạt động của phần mềm được mô tả trong hình sau.



Sau đây là phần mô tả chi tiết chương trình.

Chương trình của **sân khấu**.



Thiết lập các giá trị ban đầu cho Điểm số, Thời gian hệ thống, truyền thông điệp **Begin** để bắt đầu trò chơi.

Sau đó điều hành các tham số này, thưởng điểm và tăng lượt chơi cho người dùng.

- Nếu thời gian chơi vượt qua các bội của 15 thì tăng điểm số +3.

- Nếu điểm số vượt quá 200, 400, … thì thông báo thưởng lượt chơi bằng cách truyền thông điệp **Bonus**.

Chương trình của **Bóng** sau khi nhận thông điệp **Begin**.

Bóng rơi từ phía trên xuống, nếu gặp thanh ngang thì bật lên.

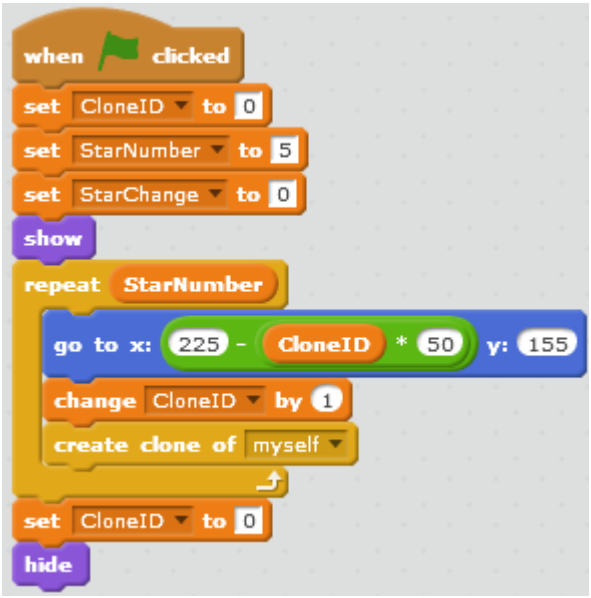
- Nếu gặp thanh ngang thì tăng điểm lên 5.

- Nếu bóng bị rơi xuống phía dưới:

+ Nếu còn lượt thì sẽ thông báo giảm lượt chơi thông qua 1 biến nhớ **StarChange**.

+ Nếu hết lượt thì truyền thông điệp **End** để kết thúc chương trình.

Chương trình gốc của **Ngôi sao (Star)**.



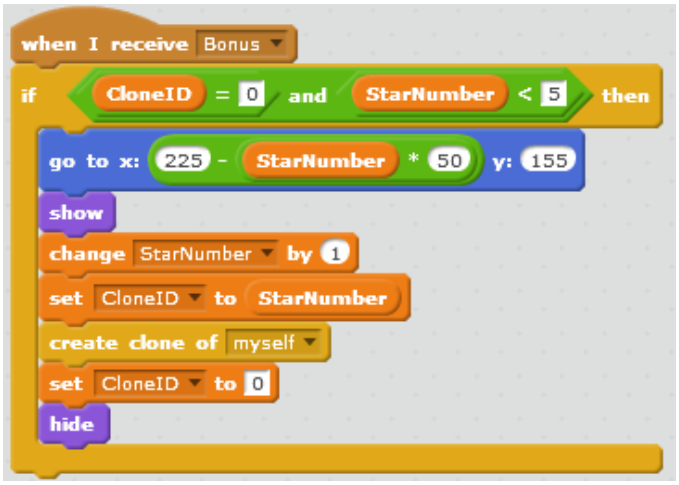
Thiết lập các giá trị ban đầu như StarNumber.

Sau đó sẽ khởi tạo quá trình tạo 5 Clone và thể hiện các ngôi sao clones này trên màn hình tại góc phải trên.

Các ngôi sao clone được thiết lập và qui định bởi biến riêng **CloneID**.

Ngôi sao gốc có CloneID = 0, các ngôi sao clone có CloneID > 0.

Chương trình của **Ngôi sao (Star)** khi gặp sự kiện **Bonus**.

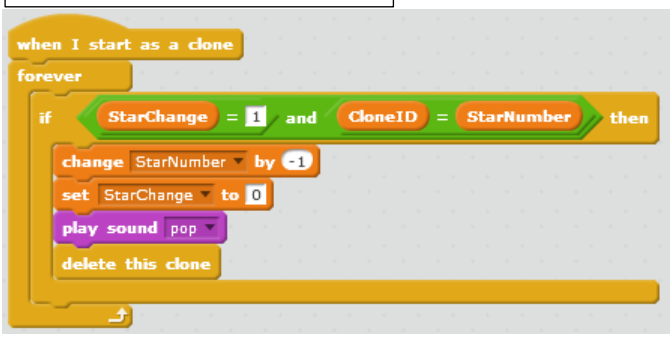


Nếu gặp thông điệp Bonus (tăng lượt), kiểm tra và tăng StarNumber lên 1 đơn vị.

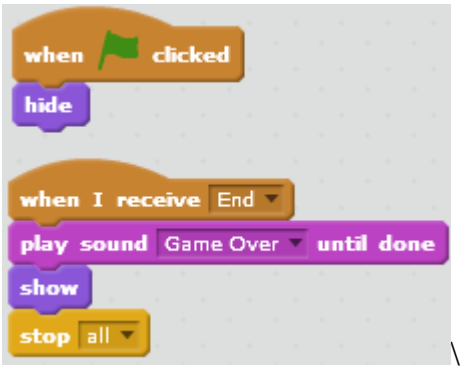
Sau đó khởi tạo 1 clone mới có CloneID = StarNumber và thể hiện clone này trên màn hình.

Chương trình Clone của **Ngôi sao (Star)**.

Clone



Luôn kiểm tra nếu có thông tin về giảm lượt (nếu StarChange = 1) thì tìm đến Clone có CloneID cao nhất và xóa Clone này trên màn hình, sau đó giảm StarNumber đi -1.

Chương trình điều khiển **Thanh ngang.**	Chương trình của nút **Game Over**.
	

Câu hỏi, bài tập

1. Em hãy thiết kế 1 bộ trang phục khác cho chương trình **Show Number**, các chữ số có độ rộng khác nhau để có thể hiện đúng chữ số trên màn hình.

2. Viết lại thủ tục **Show Number** dưới khuôn dạng sau:

3. Hoàn thiện các thủ tục **Init** và **Show Word** trong mục 6, dùng để thể hiện chữ trên màn hình.

4. Hoàn thiện thủ tục **Counting from** trong hoạt động 5 và viết chương trình thể hiện máy đếm số tăng trên màn hình.

5. Thiết kế thủ tục tương tự **Counting down** và viết chương trình thể hiện máy đếm số giảm trên màn hình.

6. Viết 1 chương trình thể hiện đồng thời việc hiện máy đếm số tăng dần tại góc trái trên màn hình và hiện máy đếm số giảm dần tại góc phải dưới màn hình.

7. Viết thêm các thủ tục sinh bộ câu hỏi trắc nghiệm sau cho bộ dữ liệu **dsQG** và **dsThudo** (xem hoạt động 8).

> Quốc gia **Balan** có thủ đô là thành phố nào trong số các tên sau?
>
> A. Kabun B. Hanoi C. Warsaw C. Paris.

8. Viết thêm các thủ tục sinh bộ câu hỏi trắc nghiệm sau cho bộ dữ liệu **dsQG** và **dsThudo** (xem hoạt động 8).

> Trong các tên sau, tên nào là tên của một quốc gia?
>
> A. Kabun B. Hanoi C. Warsaw C. Pháp.

9. 8. Viết thêm các thủ tục sinh bộ câu hỏi trắc nghiệm sau cho bộ dữ liệu **dsQG** và **dsThudo** (xem hoạt động 8).

Trong các tên sau, tên nào là tên của một thành phố thủ đô của một quốc gia?

A. Laos B. Hanoi C. Japan C. France.

10. Hoàn thiện chương trình, trò chơi thả bóng đã được phân tích trong hoạt động 9.

11. Viết thuật toán và thủ tục thực hiện công việc sau:

Cho trước 1 dãy số có thể có các phần tử bằng nhau. Ví dụ chúng ta có dãy số sau:

1 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1 0 0

Viết 1 thủ tục lấy ra ngẫu nhiên 4 phần tử khác nhau từng đôi một từ dãy số trên.

12. Cho trước 1 dãy số có thể có các phần tử bằng nhau. Ví dụ chúng ta có dãy số **NList** như sau:

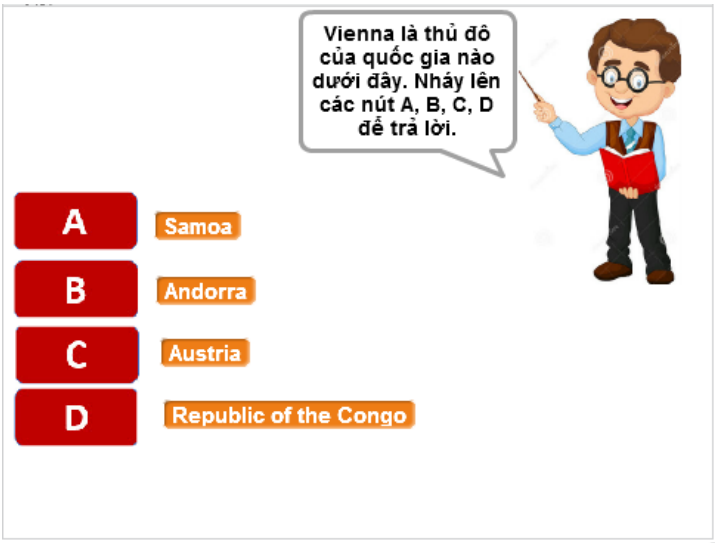
1 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1 0 0

Viết 1 thủ tục lấy ra tất cả các phần tử khác nhau của dãy trên và đưa vào 1 dãy mới **NListOut**. Ví dụ trong ví dụ trên, dãy cần tìm sẽ có các phần tử sau:

0 1 2 3 4 5 6 7 8 9 10

Mở rộng

1. Mở rộng chương trình đã có trong hoạt động 8, bổ sung thêm các nút có chữ cái A, B, C, D. Khi giáo viên đưa câu hỏi, người dùng sẽ bấm lên một trong các nút này để trả lời.



2. Mở rộng chương trình trên, cho phép giáo viên sẽ tự động sinh và hỏi 1 trong các dạng câu hỏi đã được mô tả trong các bài tập 7, 8, 9.

Bài 23. Thiết kế 1 số phần mềm giáo dục

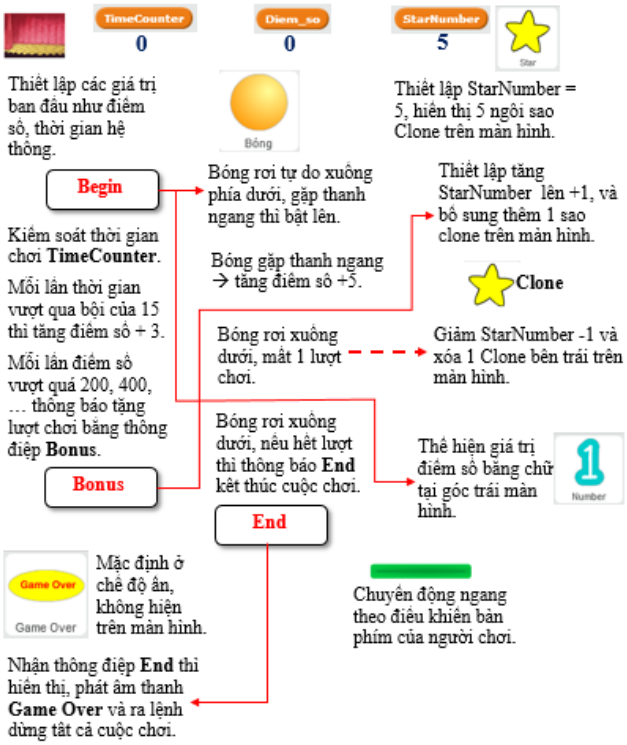
Mục đích

Học sinh hiểu được các ý tưởng và sơ đồ thiết kế của các chương trình, phần mềm được đưa ra trong bài học này.

Bắt đầu

Em hãy xem lại sơ đồ này từ chương trình ví dụ cuối cùng của bài học trước.

Trên sơ đồ này em nhìn thấy những gì?



- Các biến nhớ chính.
- Các nhân vật chính của chương trình.
- Tóm tắt chức năng và hoạt động chính của các nhân vật và nền sân khấu.
- Quan hệ giữa các nhân vật thông qua các sự kiện, thông điệp quan trọng trong quá trình thực hiện chương trình.

Từ sơ đồ trên em hãy trả lời các câu hỏi:

- Liệt kê các nhân vật.
- Liệt kê các biến nhớ chính.
- Mô tả hoạt động của các nhân vật.
- Liệt kê và mô tả ý nghĩa các thông điệp của chương trình trên.

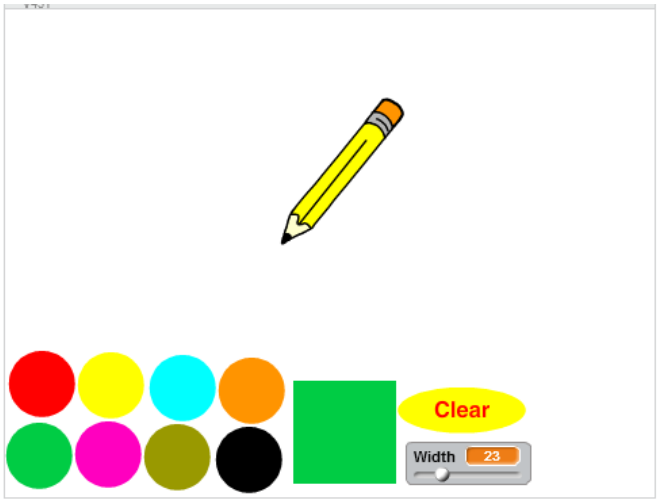
Nhiệm vụ của em trong bài học này là tìm hiểu, học tập để có thể đọc được những sơ đồ thiết kế này. Và thể hiện thiết kế này bằng chương trình.

Nội dung bài học

1. Chương trình Bút vẽ màu hoàn chỉnh

Thiết lập chương trình mô tả chức năng vẽ bằng bút vẽ tự do với bảng màu cho trước. Chương trình này sẽ mở rộng và tổng quát hóa các ví dụ mô phỏng bút vẽ mà chúng ta đã học.

Butvetudo.sb2



Yêu cầu ban đầu cho chương trình.

- Chương trình mô phỏng bút vẽ tự do, nhưng yêu cầu thao tác vẽ là nhấn giữ và rê chuột.

- Có bảng màu và 1 nút hiện màu hiện thời đang vẽ.

- Có nút xóa màn hình vẽ lại.

- Có nút điều khiển độ rộng của bút vẽ.

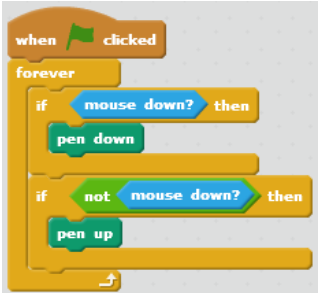
Hình sau mô tả các nhân vật tối thiểu, ban đầu của chương trình.

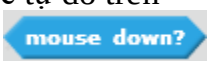
Các nhân vật của chương trình. Có thể chia thành các nhóm sau:
- Bút vẽ.
- Các nút dùng để thiết lập màu (tròn).
- Nút màu hiện thời (hình vuông).
- Nút **Clear** dùng để xóa màn hình.

Để mô tả việc đặt bút vẽ khi rê chuột chúng ta dùng 2 hàm, biểu thức logic 

và 

để xác định thời điểm hạ bút và nhấc bút.

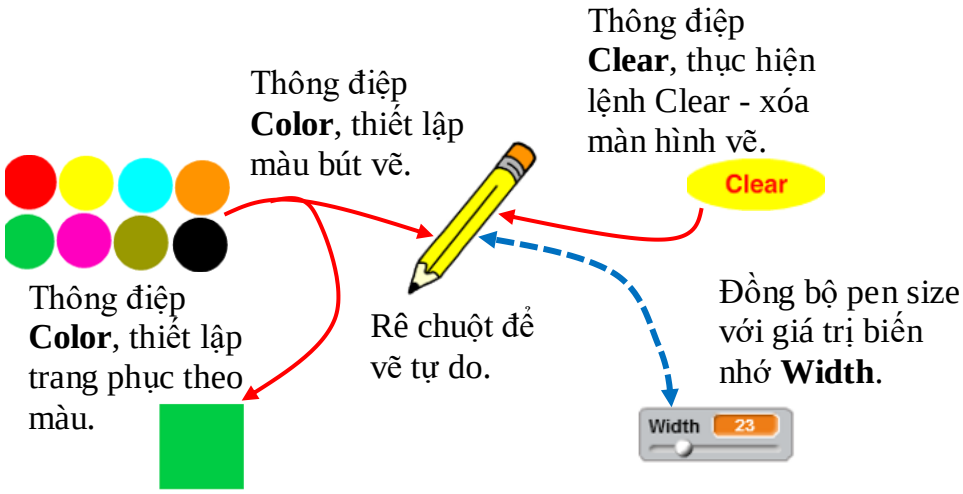


Đoạn chương trình mô tả bút vẽ tự do trên màn hình, sử dụng các sự kiện 

cho hạ bút và 

để nhấc bút.

Sơ đồ hoạt động của chương trình này khá đơn giản như sau:



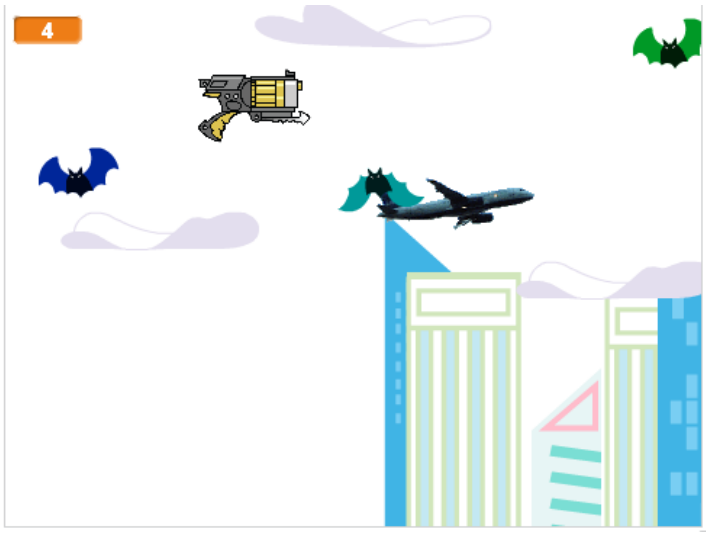
Nhận xét: đây là một chương trình khá đơn giản nhưng có nhiều ý nghĩa thực tế.

Em hãy hoàn thiện chương trình này và thực hiện tiếp các mở rộng của chương trình trong phần bài tập cuối bài học.

2. Kỹ thuật bắn súng

Phần mềm - trò chơi bắn súng đơn giản sau sẽ giúp em hình thành các tư duy ban đầu của các trò chơi loại này. Kỹ thuật trọng tâm của chương trình là bắn súng.

Bansung.sb2



Giao diện tượng trưng của chương trình. Nhân vật chính là khẩu súng và các con chim đen. Nhiệm vụ của người chơi là bắn đạn trúng càng nhiều chim đen càng tốt. Nếu súng bị va vào chim đen thì thua.

Phần lõi chính của chương trình là qui trình "bắn súng". Mô tả như sau: khi bấm phím Space, viên đạn sẽ bay ra từ nòng súng và đi thẳng. Người chơi cần điều khiển súng sao cho hướng đến đúng các con chim đen xuất hiện trước mặt.

Hình ảnh súng, viên đạn được bắn ra từ đầu nòng súng và chim đen.

Yêu cầu cụ thể của chương trình như sau:

- Người dùng sẽ dùng phím để điều khiển súng chuyển động lên, xuống, phải, trái trên màn hình.

- Nhấn phím Space báo hiệu **Shot** (bắn). Đạn sẽ bắn ra từ đầu nòng súng về phía phải màn hình. Nếu gặp chim đen thì nổ tung và tăng điểm lên 10 (chim + đạn sẽ biến mất). Nếu đạn gặp máy bay sẽ bị trừ điểm.

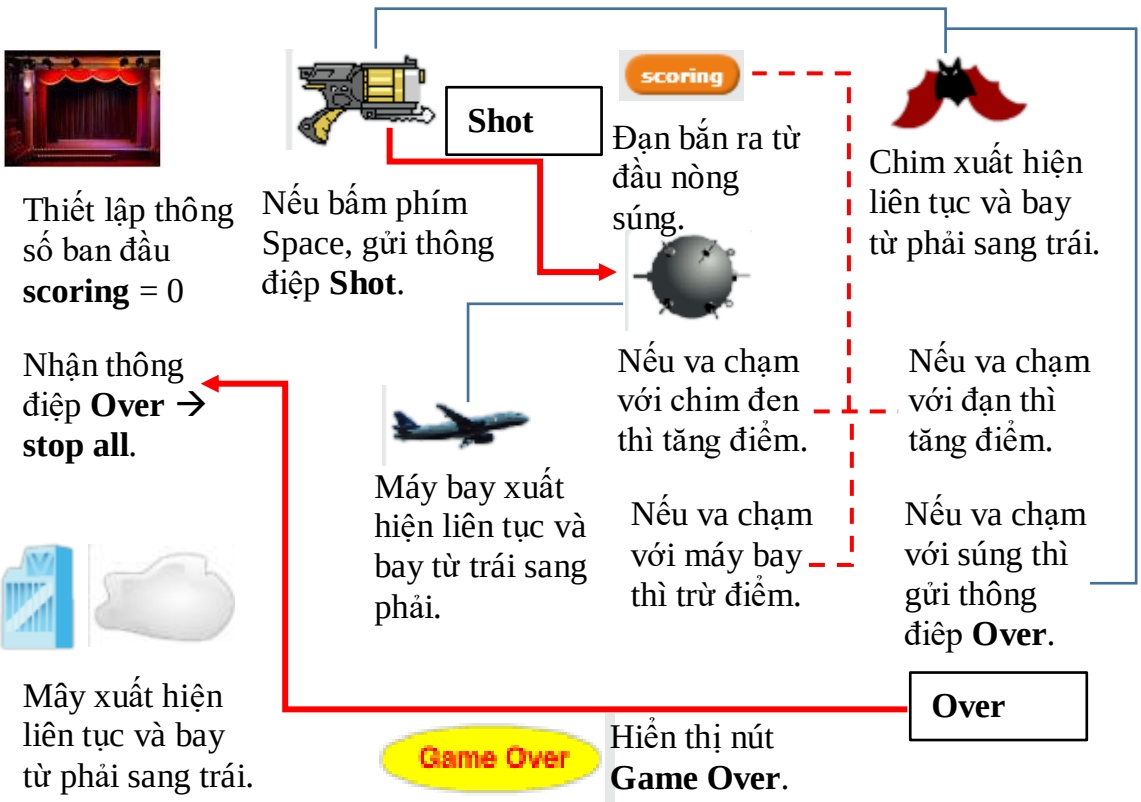
- Máy bay xuất hiện ngẫu nhiên từ bên trái và bay sang bên phải (với vận tốc nhỏ hơn vận tốc của đạn).

- Nếu để súng chạm vào chim đen sẽ thua, gửi thông điệp Over để kết thúc chương trình.

- Các tòa nhà được sắp xếp xuất hiện ngẫu nhiên. Mây sẽ xuất hiện ngẫu nhiên và trôi từ phải sang trái trên màn hình.

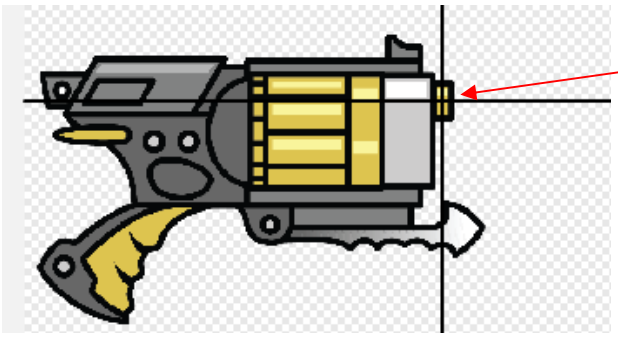
Các nhân vật của chương trình.

Sơ đồ thiết kế chương trình như sau:



Để mô tả được viên đạn sẽ bắn ra từ đầu nòng súng chúng ta cần sử dụng hàm lấy thuộc tính các nhân vật trong nhóm Cảm biến: 

. Cần thiết lập tâm của súng nằm gần đầu nòng súng như hình sau:



Vị trí tâm của súng.

Khi đối tượng đạn chuẩn bị tạo clone, cần 2 lệnh sau để thiết lập đạn sẽ luôn bắn ra từ đầu nòng súng.

Đoạn chương trình điều khiển viên đạn khi có thông điệp **Shot** như sau:

Em hãy hoàn thiện chương trình trên và thực hiện các bài tập bổ sung tiếp theo.

3. Flappy Bird

Trò chơi Flappy Bird đã nổi tiếng toàn thế giới. Trong hoạt động này, em sẽ cùng suy nghĩ để thiết kế 1 phương án đơn giản của phần mềm này.

Flappy Bird.sb2

Giao diện ban đầu của trò chơi.

Nháy phím Space để bắt đầu.

Mô tả trò chơi như sau:

- Trò chơi được bắt đầu khi người dùng bấm phím **Space**. Trước đó là màn hình ban đầu của Flappy Bird.

- Tiếp theo, người chơi sẽ phải điều khiển con chim bằng cách nhấn phím Space để đưa chim đi qua các khe hở của các ống.

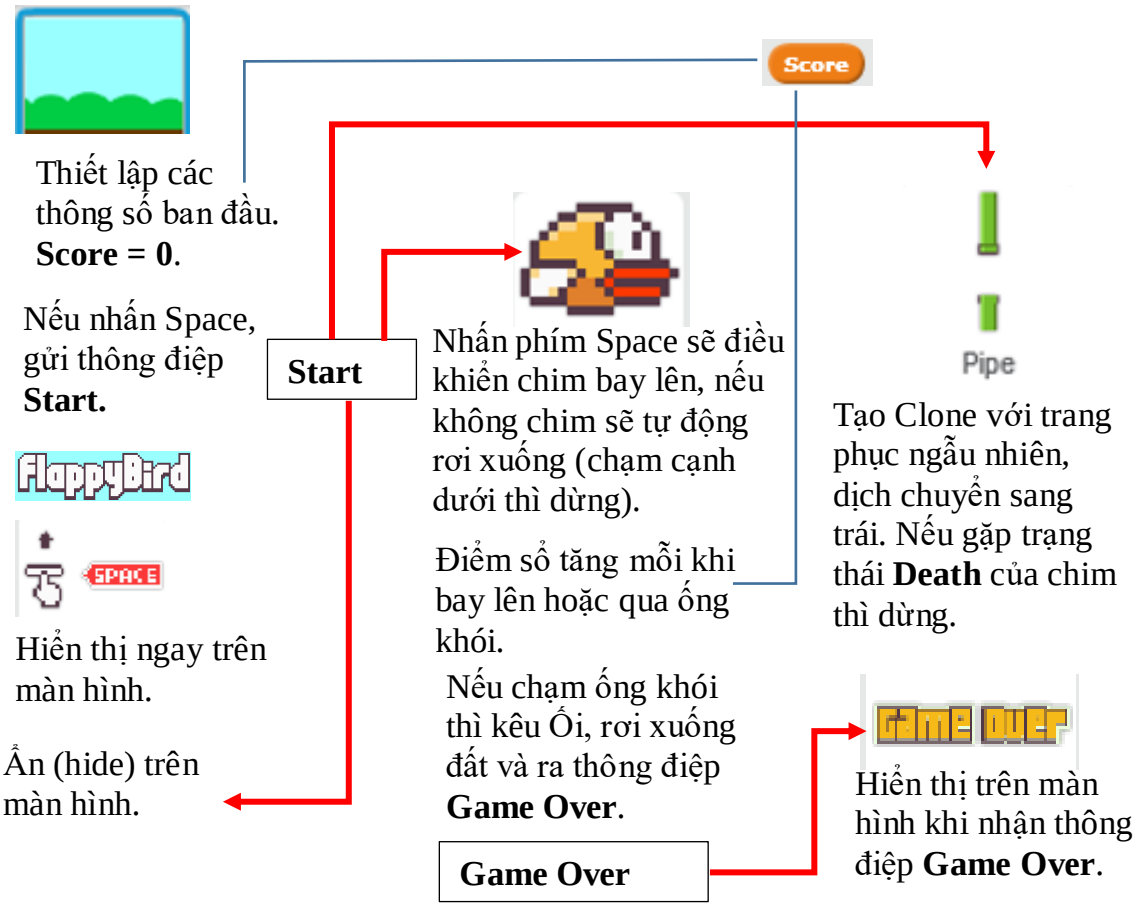
- Các ống sẽ lần lượt xuất hiện chuyển động từ phải sang trái màn hình. Tuy nhiên người chơi sẽ có cảm giác điều khiển chim bay về phía trước.

- Mỗi lần nhấn phím **Space**, chim lại bay lên và đi được 1 đoạn. Người chơi cần khéo léo điều khiển chim.

- Nếu chim chạm vào 1 ống khói bất kỳ thì trò chơi kết thúc, xuất hiện dòng chữ Game Over.

Các nhân vật của chương trình, trong đó chỉ có 2 nhân vật chính là chim (bird) và ống khói (pipe).

Sơ đồ thiết kế và hoạt động của phần mềm tương đối đơn giản như sau:



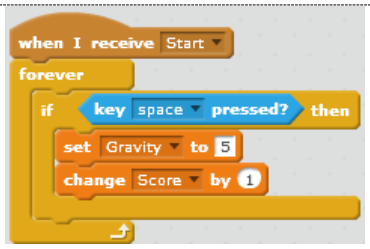
Điều khiển hoạt động của nhân vật chính (bird) thông qua tham số Gravity

Chúng ta sẽ thiết lập 1 biến nhớ riêng của nhân vật bird với tên **Gravity** (dịch: trọng lực). Tham số này điều khiển khả năng bay lên của chim khi chúng ta bấm phím **Space**. Để tạo hiệu ứng mỗi khi em bấm **Space**, chim sẽ bay lên, sau đó sẽ từ từ bị rơi xuống, thuật toán như sau:

- Mỗi lần nhấn Space sẽ thiết lập Gravity 1 giá trị cố định > 0 (ví dụ = 5).

- Chim sẽ liên tục thay đổi giá trị tọa độ Y theo Gravity. Như vậy nên Gravity > 0. chim sẽ bay lên, Gravity < 0, chim sẽ rơi xuống.

- Theo thời gian giá trị của Gravity sẽ liên tục giảm, cho đến khi người chơi bấm Space thì quá trình giảm kết thúc và Gravity được đặt lại > 0.

		
Nếu người dùng nhấn phím Space thì lập tức gán Gravity = 5.	Chỉ sau 0.2 giây ban đầu, giá trị Gravity sẽ liên tục giảm. Điều này sẽ yêu cầu người chơi phải bấm Space liên tục nếu không chim sẽ rơi xuống đất.	Trong suốt thời gian chơi, chim sẽ thay đổi liên tục độ cao (tọa độ Y) theo Gravity. Vậy nếu Gravity > 0 thì chim bay lên, nếu Gravity < 0 thì chim rơi xuống dưới.

Điều khiển nhân vật chim khi va chạm với ống khói

Nếu va chạm với ống khói, chim kêu "ối" và lập tức rơi thẳng xuống đất. Khi chạm đất thì ra lệnh dừng toàn bộ chương trình bằng thông điệp Game Over.

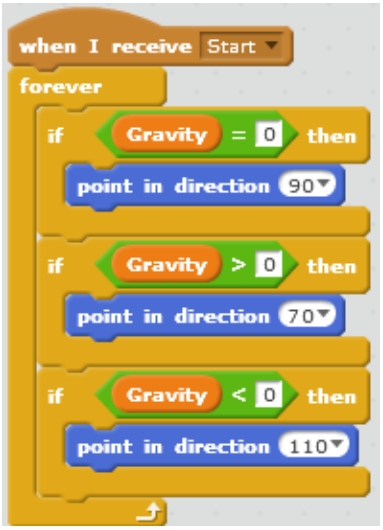
Nếu va chạm ống khói, chim kêu ối, dừng tất cả các đoạn chương trình khác của nhân vật này.

Đoạn chương trình này mô tả chim rơi thẳng xuống đất, sau đó sẽ thông báo Game Over và dừng chương trình.

Tạo dáng cho nhân vật chim bay

Các lệnh sau có tác dụng phụ, làm cho nhân vật chim thêm nổi bật.

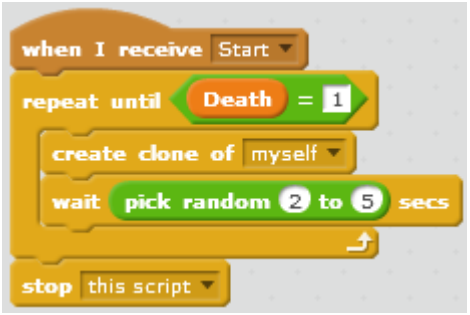
Trong khi bay chim luôn thay đổi trang phục của mình tạo hiệu ứng vẫy cánh.



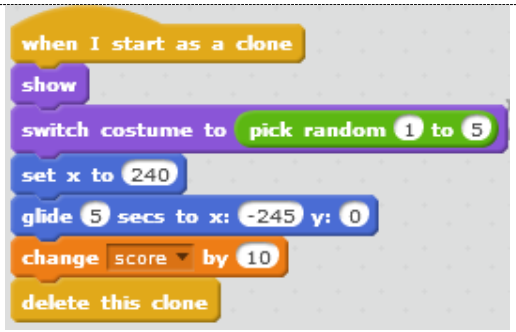
Đoạn chương trình này mô tả dáng điệu của chim trong khi bay. Tùy thuộc vào Gravity mà chim sẽ hướng lên trên, hướng ngang hoặc hướng xuống dưới trong khi bay.

Điều khiển hoạt động của ống khói (pipe)

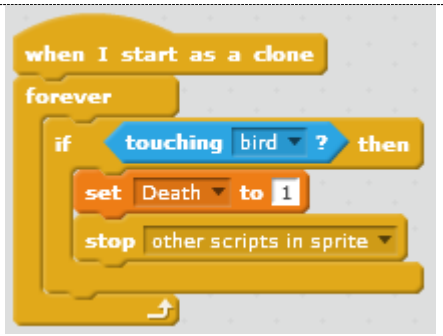
Chúng ta sẽ sử dụng 1 biến nhớ riêng Death để chỉ trạng thái trước và sau khi va chạm với chim. Giá trị ban đầu của Death = 0.



Đoạn chương trình chính của nhân vật ống khói, tự động sinh clone của mình sau khoảng mỗi 2-5 giây.



Để mô tả chuyển động các ống khói (clone), có nhiều cách. Ở đây chúng ta dùng lệnh **glide**, chuyển động sẽ được mô tả mượt hơn.

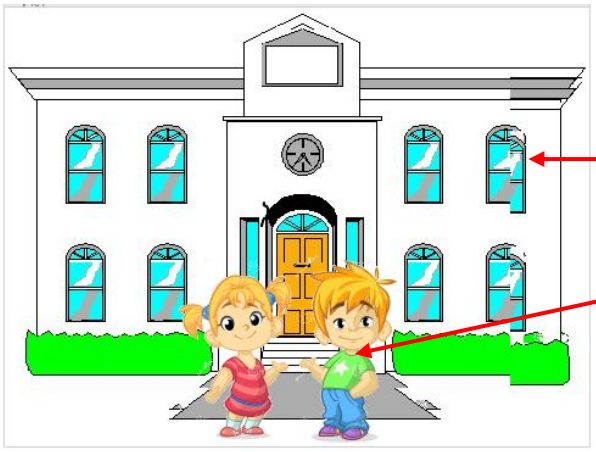


Nếu ống khói va chạm với chim thì lập tức thiết lập lại trạng thái Death = 1 và dừng toàn bộ các chương trình trong cửa sổ này.

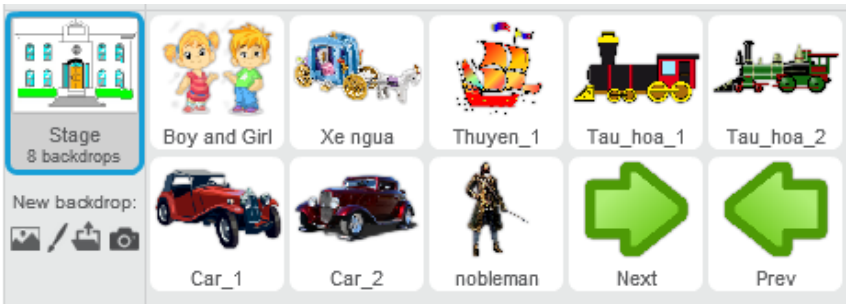
4. Trình diễn trong bảo tàng

Trong hoạt động này chúng ta sẽ cùng thiết kế một mô hình chương trình được sử dụng rất nhiều trong giáo dục, văn hóa, đó là việc mô phỏng trình diễn, giới thiệu theo trang trí của bảo tàng, triển lãm, nhà văn hóa, …. Một bảo tàng (hay di tích văn hóa) có nhiều phòng, cảnh. Trong mỗi phòng lại có nhiều hiện vật. Cần có 1 chương trình giúp người dùng có thể tham quan tất cả các phòng, gian hàng và hiện vật trong bảo tàng này.

Bao tang.sb2



Nhân vật của chương trình là người dẫn chương trình và các hiện vật trong bảo tàng đó.



Sân khấu chính là các gian phòng của bảo tàng, là nơi chúng ta cần giới thiệu cho người tham quan.


Cảnh 1


Cảnh 2


Cảnh 3


Cảnh 4


Cảnh 5


Cảnh 6


Cảnh 7


Cảnh 8

Ví dụ trong chương trình mẫu này, các nền sân khấu đánh số từ 2 đến 8 chính là các gian phòng chính của bảo tàng. Các hiện vật (nhân vật) được thể hiện bên trong các phòng này.

Yêu cầu chương trình.

- Bắt đầu chương trình là lời thuyết minh welcome trước khi bắt đầu thời gian tham quan triển lãm, bảo tàng.

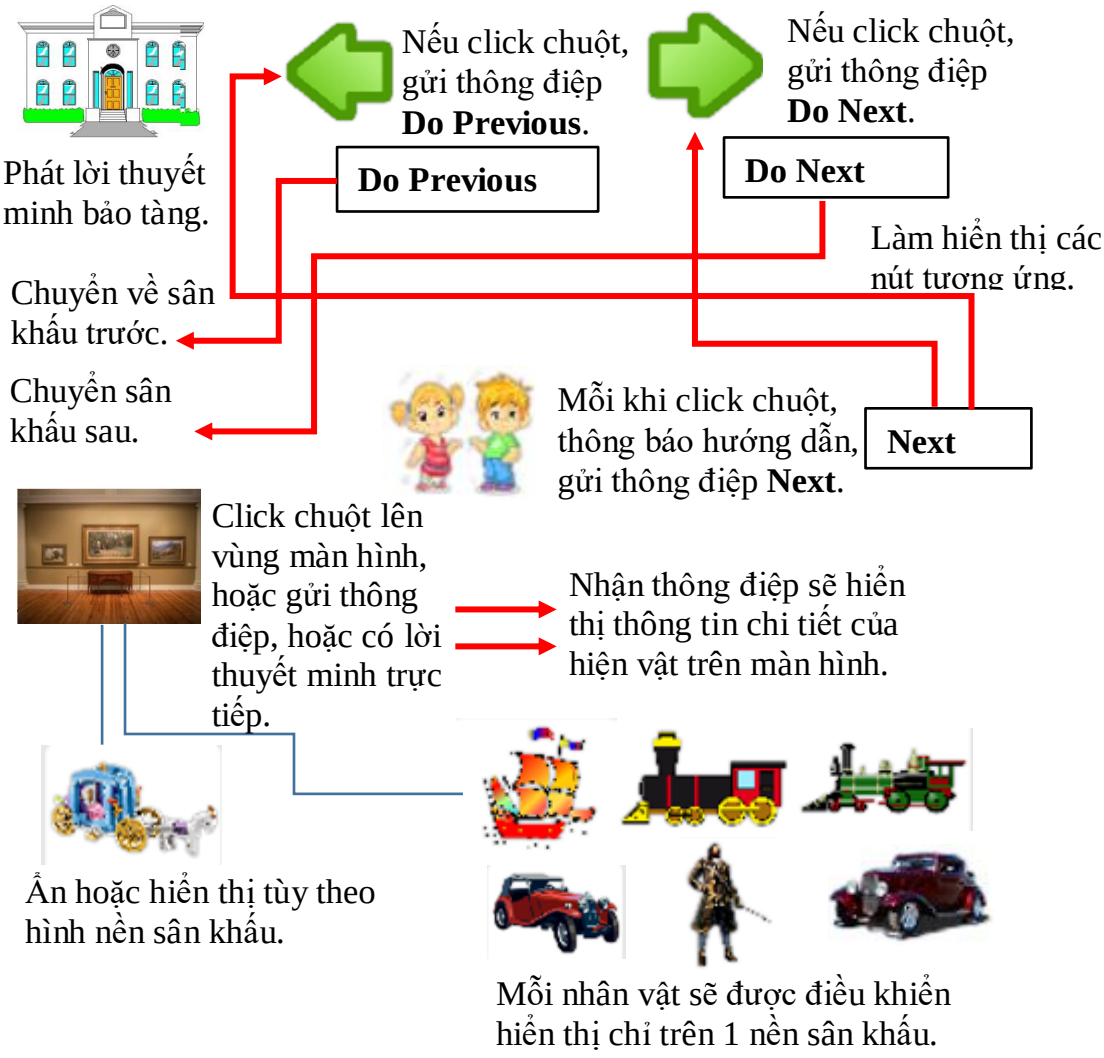
- Trong suốt quá trình khi tham quan qua các phòng, người dùng có thể nháy chuột lên các hiện vật hoặc vị trí hình ảnh trên màn hình để xem hoặc nghe giải thích chi tiết.

- Muốn chuyển phòng cần nháy lên 2 em bé người dẫn chương trình để xem hướng dẫn. Khi xuất hiện các nút Next, Previous thì nháy lên các nút này để chuyển phòng. Chú ý khi chuyển sang 1 phòng mới thì các nút này không xuất hiện.

- Qui định cụ thể mỗi hiện vật chỉ xuất hiện đúng tại 1 vị trí trong 1 gian phòng cụ thể. Như vậy mỗi nhân vật đóng vai trò hiện vật chỉ được phép xuất hiện trên 1 nền sân khấu xác định.

Chú ý: Chương trình này chỉ có thời điểm bắt đầu và không có kết thúc. Người tham quan có thể khám phá bảo tàng bao lâu tùy thích.

Sơ đồ thiết kế và hoạt động của chương trình như sau:



Em hãy hoàn thiện chương trình tham quan bảo tàng theo yêu cầu trên và thực hiện các mở rộng tiếp theo trong phần bài tập.

5. Trình diễn nhảy múa và hát

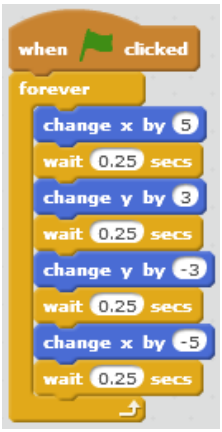
Chương trình tiếp theo của hoạt động này là mô phỏng nhảy múa, ca hát theo nhịp trống.

Nhay mua hat.sb2

Các nhân vật: 2 em bé, ca sĩ, Hùng và Lan.

Các nhân vật: 2 em bé, ca sĩ, Hùng và Lan.

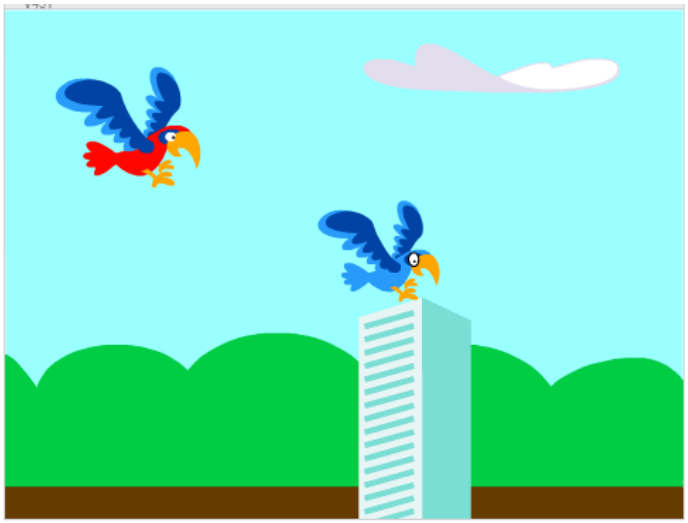
Sau đây là 1 số chương trình mẫu.

Sân khấu	2 em bé	Ca sĩ, Lan, Hùng
		
Thiết lập nhịp trống 1 giây (60 lần / phút). Bài hát được nghe là "chú ếch con". Trống đánh mỗi nhịp trống (1 giây).	Chương trình mô phỏng nhân vật 2 em bé (boy and girl) nhảy múa theo nhịp trống. Chú ý: tổng các thời gian nghỉ là 1 giây.	Mô phỏng nhảy múa bằng cách thay đổi trang phục. Thời gian chờ là 1/2 giây. **Chú ý:** vì chúng ta thiết lập nhịp trống là 1 giây nên tất cả các hoạt động mô phỏng đều phải khớp với thời gian 1 phút.

6. Hiệu ứng bay, chạy nhảy

Để mô phỏng chim bay, máy bay bay trên bầu trời chúng ta phải làm gì? Để tạo hiệu ứng bay có thể cho nhân vật của chúng ta "bay" thật, nhưng cũng có thể làm ngược lại: cho nhân vật đứng im và cho các hình ảnh nền như nhà cửa, mây trời chuyển động theo chiều ngược lại. Kỹ thuật này được dùng rất nhiều trong các phần mềm mô phỏng. Chúng ta cùng thiết kế 1 chương trình đơn giản như vậy.

Chim bay. sb2



Chương trình mô tả 2 con chim bay (bird1 và bird2) từ trái sang phải trên màn hình. Chim bird2 sẽ bay nhanh hơn và vượt qua bird1.

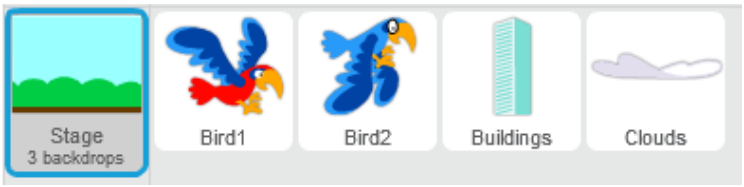
Yêu cầu là mô phỏng bay để người quan sát thấy y như thật.

Các yêu cầu bổ sung của chương trình:

- Mô phỏng 2 chim (bird1 và bird2) bay (có vỗ cánh) từ trái sang phải trên màn hình.

- Bird2 bay nhanh hơn Bird1.

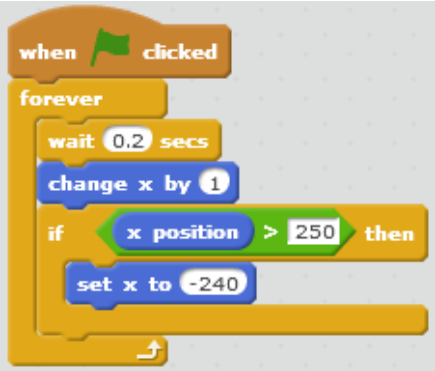
- Nền bao gồm các ngôi nhà và mây xuất hiện ngẫu nhiên từ bên phải và chuyển động đều sang bên trái để mô phỏng chim bay.



Chúng ta bắt đầu từ 2 nhân vật chim: bird1 và bird2. Đoạn chương trình sau mô tả sự vỗ cánh của 2 con chim này.

Dòng lệnh **wait … secs** mô tả việc vỗ cánh như thế nào. Ví dụ đặt 0.2 cho bird1 và 0.1 cho bird2.

Đoạn chương trình sau bổ sung cho bird2 mô tả hiệu ứng do bird2 bay nhanh hơn bird1 nên nó sẽ được mô phỏng có chuyển động sang phải. Khi bay đến cạnh bên phải màn hình, bird2 sẽ quay lại bay từ cạnh trái màn hình. Đoạn chương trình mô tả hiệu ứng này như sau:



Bird2 sẽ dịch chuyển sang phải 1 pixel sau mỗi 0.2 giây.

Bird2 khi vượt qua cạnh bên phải màn hình sẽ xuất hiện lại bên trái.

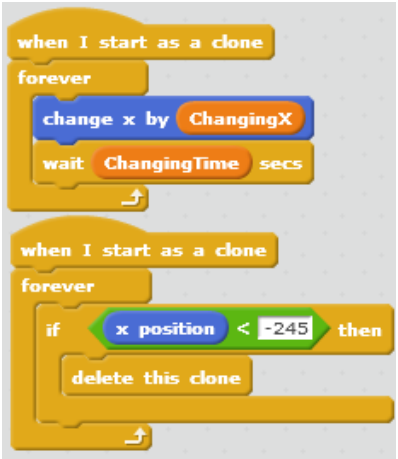
Kỹ thuật mô tả chuyển động của nhà và mây tương tự nhau, điều quan trọng cần nhớ là phải mô tả 2 đối tượng này chuyển động với vận tốc như nhau (từ phải qua trái) trên màn hình.

Chúng ta sẽ sử sụng 2 biến nhớ tổng thể **ChangingX** và **ChangingTime** để chỉ khoảng cách thay đổi tọa độ X và khoảng thời gian chờ khi mô tả chuyển động của clone của nhà và mây. Các giá trị này có thể thay đổi để tạo hiệu ứng cho việc thay đổi tốc độ bay của chim. Giá trị ban đầu của các tham số này.

Bây giờ có thể quan sát đoạn chương trình mô tả hoạt động của nhân vật tòa nhà, với mây hoàn toàn tương tự.

Đoạn chương trình sinh **clone** ngẫu nhiên.

Sau mỗi khoảng thời gian ngẫu nhiên thì tạo clone với trang phục ngẫu nhiên từ danh sách.



Mô tả chuyển động của clone khi xuất hiện: sẽ dịch chuyển dần về bên trái 1 khoảng cách **ChangingX** sau mỗi thời gian **ChangingTime**.

Khi khuất hẳn sau cạnh màn hình trái thì tự xóa clone.

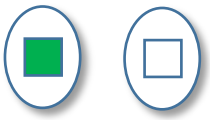
Em hãy hoàn thiện chương trình và thực hiện các bài tập mở rộng.

Câu hỏi, bài tập

1. Em hoàn thiện tất cả các chương trình đã mô tả trong bài học (5 chương trình chính).

2. Bổ sung tính năng sau cho chương trình vẽ tự do.

Bổ sung 1 nút có hình như dưới đây để thiết lập trạng thái có thể vẽ của chương trình. Bút vẽ chỉ có tác dụng trạng thái vẽ là **On**. Khi bắt đầu chương trình trạng thái mặc định là **Off**. Nháy lên nút này sẽ chuyển trạng thái **On/Off**.



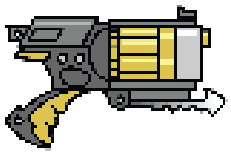
3. Bổ sung vào chương trình vẽ tự do: các phím điều khiển up, down sẽ có tác dụng tăng, giảm độ rộng của nét bút (trực tiếp làm tăng, giảm giá trị **Width**).

4. Điều chỉnh, nâng cấp 1 tính năng sau cho chương trình bắn súng.

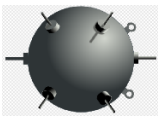
Khi bắn súng trúng máy bay, máy bay sẽ đứng lại và rơi xuống đất. Khi rơi xuống đất sẽ có tiếng nổ.

5. Bổ sung thêm 1 tính năng cho chương trình bắn súng: cho phép người dùng thay đổi súng và thay đổi kiểu đạn. Sử dụng các phím nhanh S và D để thực hiện sự thay đổi này. Ví dụ:

Súng:




Đạn:




6. Bổ sung tính năng thể hiện điểm số của chương trình bắn súng bằng công cụ thể hiện chữ số mà em đã biết trong bài học trước.

7. Hoàn thiện chương trình xem bảo tàng, thực hiện theo qui luật sau:

- Mỗi nhân vật chỉ được phép thể hiện trong đúng 1 phòng, tại 1 vị trí cố định.

- Nhân vật MC (2 em bé sẽ xuất hiện trong tất cả các cảnh nhưng đứng tại các vị trí khác nhau.

Qui định cụ thể như sau:


Cảnh 1


Cảnh 2


Cảnh 3


Cảnh 4

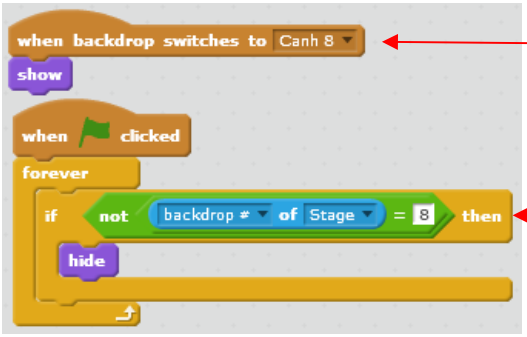

Cảnh 5


Cảnh 6


Cảnh 7


Cảnh 8

Có thể sử dụng đoạn code sau cho mỗi nhân vật để thực hiện yêu cầu chỉ thể hiện trên đúng 1 nền sân khấu. Ví dụ sau dành cho nhân vật Car_2, chỉ xuất hiện trên cảnh nền 8.



Nếu nền sân khấu chuyển cảnh 8 thì hiện nhân vật.

Nếu nền sân khấu chuyển cảnh không là 8 thì ẩn nhân vật.

8. Mở rộng chương trình xem bảo tàng như sau:

Thu âm sẵn lời thuyết minh cho mỗi phòng bảo tàng. Mỗi khi nháy nút Next hoặc Previous để vào 1 phòng thì lời thuyết minh đó sẽ bật lên cho người tham quan nghe trước khi thực hiện các quan sát khác.

9. Mở rộng thêm 1 chức năng nữa cho chương trình xem bảo tàng.

Mỗi nền sân khấu = 1 phòng triển lãm được đặt tên (ví dụ: Lịch sử châu Âu thế kỷ 17). Mỗi khi chuyển vào 1 phòng thì tên của phòng đó sẽ hiện trên góc của màn hình.

10. Mở rộng chương trình nhảy múa ca hát như sau:

- Bổ sung vào chương trình 2 bài hát.

- Sơ đồ hoạt động các nhân vật sẽ thay đổi như sau:

(a) Bắt đầu vào chương trình là cảnh ngoài sân khấu, MC giới thiệu bài hát đầu tiên. Nháy 1 nút sẽ bắt đầu bài hát thứ nhất. Khi hát sẽ mở ra cảnh sân khấu như chúng ta đã biết của chương trình hiện thời.

(b) Hết bài hát 1 thì đợi 2 giây sẽ xuất hiện trở lại cảnh ban đầu với MC (2 em bé) giới thiệu bài hát thứ 2. Nháy 1 nút sẽ bắt đầu bài hát này, lại mở ra cảnh sân khấu hát như chúng ta đã biết.

(c) Khi hát xong bài thứ 2 thì hiện 1 khung chữ nhật trong đó ghi: **Đã hết buổi trình diễn. Cám ơn.**

11. Tổng quát ý tưởng trên cho 1 chương trình với n bài hát bất kỳ.

12. Mở rộng ý tưởng trên bằng cách thay đổi giao diện của cửa sổ ban đầu như sau, sử dụng trong nhiều trường hợp cho phép người dùng lựa chọn các bài hát ưa thích.

Nháy chuột lên nút tương ứng để vào xem bài hát yêu thích. Khi xem xong thì tự động quay lại màn hình ban đầu.

13. Mở rộng chương trình chim bay, cho phép người chơi điều khiển cả 2 con chim bay lên, xuống bởi các phím. Hệ thống điều khiển này phải hoạt động đồng thời cho cả 2 con bird1 và bird2. Người dùng sẽ điều khiển sao cho các con chim này không va chạm với các tòa nhà. Nếu va chạm thì coi như thua, chương trình sẽ dừng lại.

14. Bổ sung khả năng tính điểm (score) của chương trình chim bay. Mở rộng theo 2 cách sau:

(a) Trong không gian, phía phải màn hình sẽ ngẫu nhiên xuất hiện các ngôi sao nhỏ lấp lánh, người điều khiển chim sẽ cố gắng cho các con chim này ăn được các ngôi sao nhỏ này. Nếu va chạm và ăn được sẽ được tăng điểm.

(b) Từ phía phải màn hình sẽ ngẫu nhiên xuất hiện các con chim đen. Các chim đen này sẽ bay theo 1 đường thẳng ngang từ phải sang trái. Cần trán va chạm với các chim đen này, nếu va chạm sẽ bị trừ điểm.

Em hãy mở rộng theo cả 2 hướng trên. Điểm số được thể hiện ngay tại góc trái trên màn hình.

15. Bổ sung tính năng cho chương trình Flappy Bird như sau: mỗi khi thua (Game Over) sẽ hiện trên màn hình (chính giữa) 2 thông tin sau:

Điểm số của bạn: xxxxx

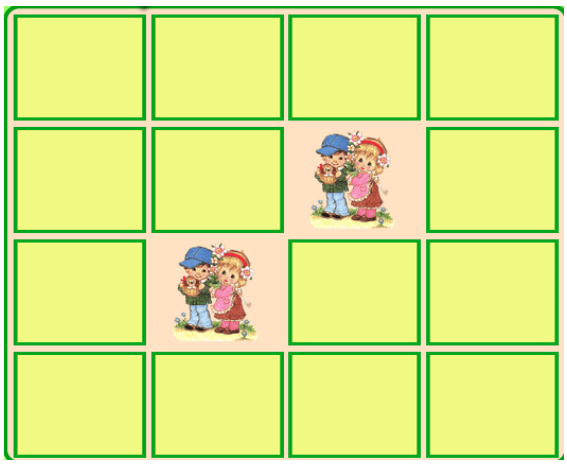
Thời gian đã chơi: yyyy giây.

Mở rộng

Tự thiết kế các chương trình, trò chơi sau.

1. Trò chơi Luyện trí nhớ (Memory)

Trên màn hình hiện 1 lưới bao gồm 12 quân cờ lập úp. Nhiệm vụ của người chơi là lần lật tất cả các quân cờ này. Bên dưới các quân cờ là các hình ảnh khác nhau. Sẽ có 12 hình với 6 cặp hình giống nhau.



Luật chơi như sau:

Người dùng cần nháy chuột liên tiếp lên 2 hình giống nhau mới lật được cả 2 quân cờ này.

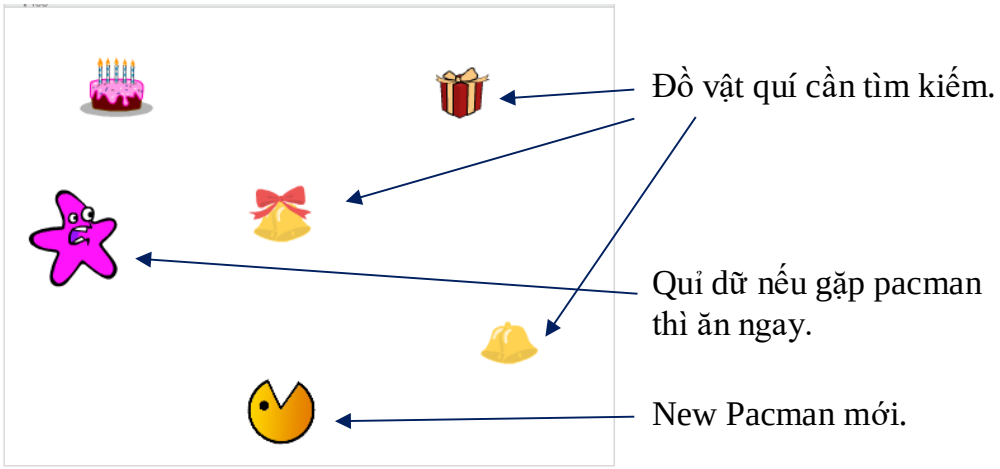
Ngược lại khi nháy lên 1 quân cờ không giống quân cờ đã lật trước đó thì cả 2 quân cờ sẽ đóng lại.

2. Trò chơi đuổi bắt (New pacman)

Trò chơi này học theo ý tưởng của phần mềm Pac-man nổi tiếng ngày xưa.

Pacman là nhân vật chính của trò chơi. Người dùng điều khiển pacman chuyển động bằng các phím lên, xuống, phải, trái. Nhiệm vụ của pacman là tìm để ăn được các đồ quí hiếm, càng nhiều càng tốt. Các đồ vật quí hiếm này sẽ xuất hiện ngẫu nhiên trên màn hình theo thời gian.

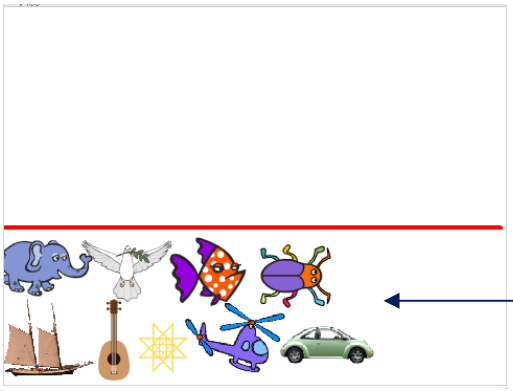
Pacman cần chú ý đến các con Quỉ dữ. Các quỉ dữ này cũng xuất hiện ngẫu nhiên và đi lại ngẫu nhiên trên màn hình, thậm chí còn tìm và muốn ăn thịt pacman.



Trò chơi sẽ tính điểm theo thời gian chơi và điểm số tích cóp được của pacman. Mỗi người chơi được phép 5 lượt dùng pacman. Nếu hết lượt thì thua.

3. Vẽ theo mẫu (Picture Paint)

Chương trình có tính năng vẽ lại theo các hình mẫu có sẵn.



Phía dưới màn hình là khung công cụ, chính là các hình mẫu.

Cách vẽ như sau: nháy chọn 1 hình mẫu, sau đó đưa ra khu vực chính, nháy chuột để vẽ theo đúng hình mẫu.

Bài 24. Các trò chơi với số

Mục đích

Học sinh hiểu được thiết kế các trò chơi, chương trình của bài học và có thể hoàn thiện, mở rộng, phát triển theo ý tưởng riêng của mình.

Bắt đầu

Chúng ta đã được làm quen với các trò chơi và kỹ thuật làm phần mềm giáo dục trong các bài học trước. Theo em các ý nào dưới đây là quan trọng nhất trong quá trình thiết kế 1 chương trình Scratch hoàn chỉnh.

- Thiết kế chính xác các nhân vật chính của chương trình.
- Mô tả các bộ dữ liệu, biến nhớ, mảng dữ liệu chính xác.
- Thiết lập sơ đồ hoạt động của chương trình hợp lý, logic.
- Mô tả một cách logic và chính xác chức năng chính của chương trình.
- Vẽ thật đẹp hình ảnh nhân vật và các nền sân khấu.

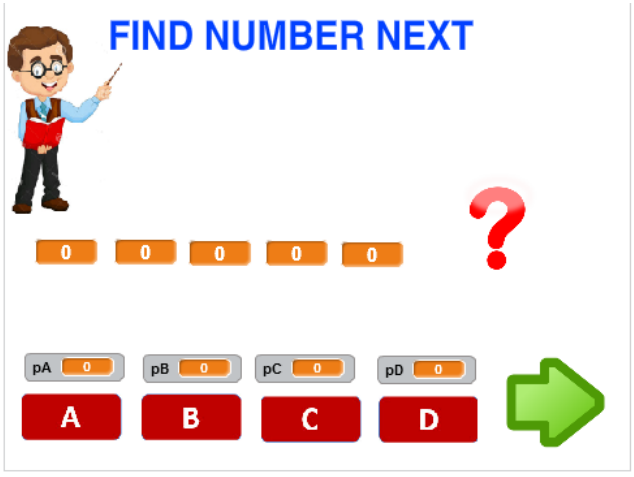
Trong bài học này, chúng ta sẽ làm quen với tất cả các hoạt động trên để hiểu được ý nghĩa quan trọng của chúng trong quá trình thiết kế phần mềm.

Nội dung bài học

1. Trò chơi điền số vào dãy, vào bảng

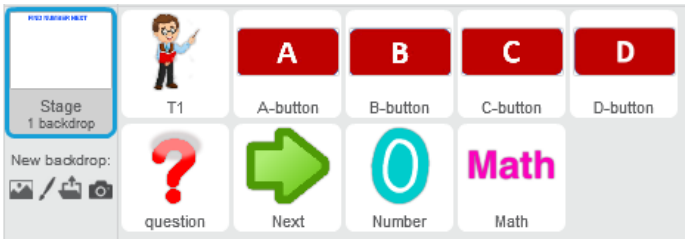
Dien so vao day.sb2

Một trong các trò chơi, bài toán vui hay gặp nhất là bài toán tìm số trong dãy hoặc bảng. Cho trước 1 dãy (hoặc bảng) số có qui luật, trong đó có 1 vị trí trống, cần tìm số còn thiếu này. Cách làm là lựa chọn 1 trong số các đáp án có sẵn (trắc nghiệm).



Giao diện đơn giản của chương trình.

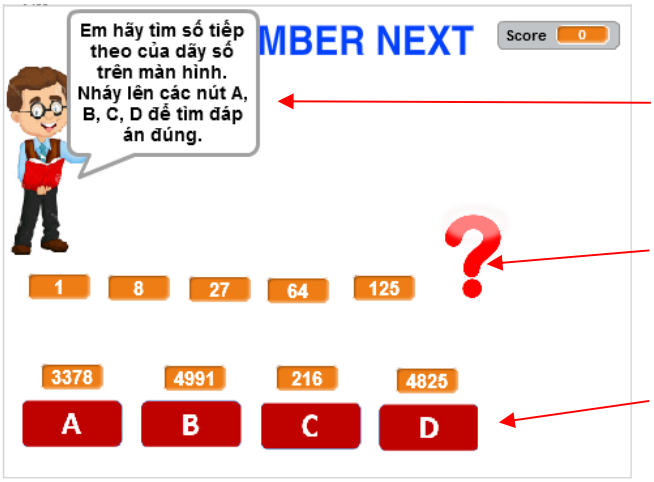
Dãy số hiện ở giữa màn hình, các đáp án trắc nghiệm phía dưới. Người dùng cần nháy lên 1 trong các nút A, B, C, D để giải (chơi).



Các nhân vật của chương trình.

Yêu cầu cụ thể của chương trình.

- Chương trình sẽ tự động sinh (ngẫu nhiên) các bài toán cho trò chơi. Mỗi bài toán sẽ bao gồm 1 dãy 5 số được thể hiện chính giữa màn hình. và 4 phương án chọn đáp án đúng.

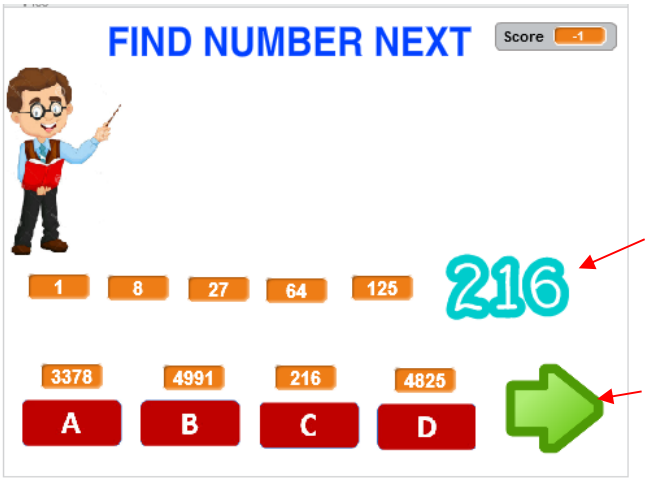


Giáo viên là người ra các thông báo hướng dẫn chính của chương trình.

Dãy số bao gồm 5 số, vị trí có dấu ? là còn thiếu 1 số cần tìm.

Người chơi cần nháy vào các nút A, B, C, D để làm bài.

- Người chơi làm bài bằng cách nháy chuột lên 1 trong các nút A, B, C, D bên dưới các đáp án có sẵn. Chương trình sẽ lập tức sẽ thông báo đúng / sai và số hạng tiếp theo của dãy sẽ hiện ngay trên màn hình. Đồng thời xuất hiện nút Next (xem hình dưới).



Đáp số sẽ hiện tại đây ngay sau khi nháy chuột làm bài.

Nút lệnh Next xuất hiện để làm tiếp.

- Nháy lên nút **Next** để tự động chuyển sang bài toán tiếp theo. Trò chơi cứ như vậy mãi mãi.

Phân tích yêu cầu của chương trình.

Bộ dữ liệu chính mô tả yêu cầu và hoạt động của chương trình.

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**N1, N2, N3, N4, N5**	5 biến nhớ tổng thể lưu 5 giá trị đầu tiên của dãy. Bài toán chính là tìm phần tử tiếp theo (một cách hợp lý) của dãy trên.	(biến nhớ)
2	**pA, pB, pC, pD**	4 phương án của bài toán. Trong 4 phương án này sẽ có 1 phương án đúng.	(biến nhớ)

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
3	**pCorrect**	Biến nhớ chỉ ra số thứ tự của phương án đúng trong số 4 phương án trên. Biến nhớ này không hiện trên màn hình.	(biến nhớ)
4	**Choice**	Biến nhớ ghi lại phương án lựa chọn của người dùng. Như vậy nếu Choice=pCorrect thì làm đúng, ngược lại là sai.	(biến nhớ)
5	**Score**	Điểm số của người chơi. Nếu giải đúng sẽ được tặng 10 điểm, nếu làm sai sẽ bị trừ 1 điểm.	(biến nhớ)

Vấn đề chính, bài toán chính của trò chơi này là tự động thiết lập các bài toán và cập nhập vào bộ dữ liệu đã nêu trên.

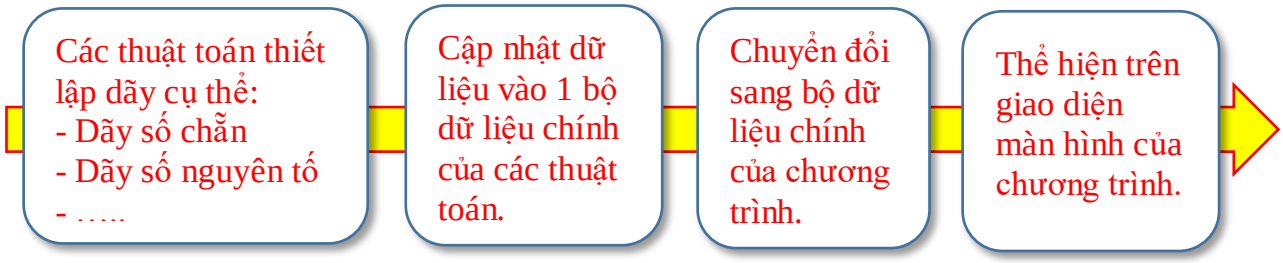
Có rất nhiều thuật toán có thể viết để thiết lập các bài toán cụ thể cho chương trình. Mỗi thuật toán chỉ ra cách thiết lập 1 dãy các số có quan hệ logic. Bảng sau ghi lại 1 vài thiết kế như vậy (sẽ được viết trong chương trình).

Stt	Mô tả qui luật dãy số	Ví dụ
1	Dãy số chẵn liên tục	2 4 6 8 10 12 14 16
2	Dãy số lẻ liên tục	1 3 5 7 9 11 13 15 17 19
3	Dãy số nguyên tố liên tục	2 3 5 7 11 13 17 19 23
4	Dãy số Fibonaci	1 1 2 3 5 8 13 21
5	Dãy số chính phương	1 4 9 16 25 36
6	Dãy số lập phương	1 8 27 64 125
7	Dãy các số tạo thành 1 cấp số cộng.	1 4 7 10 13 16 19 22
8	Dãy số tạo thành 1 cấp số nhân.	3 6 12 24 48

Bộ dữ liệu sau dùng để lưu trữ dữ liệu cho tất cả các thuật toán trên. Bộ dữ liệu này được thiết kế trong đối tượng **Math** của chương trình.

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**pNumber**	Số lượng phần tử của dãy sẽ được sinh tự động. Giá trị có trong chương trình là pNumber = 6.	(biến nhớ)
2	**NumList**	Dãy số chính (gốc) được sinh tự động của thuật toán.	(list)
3	**pList**	Dãy 4 phương án được sinh tự động. Đây chính là các giá trị ứng với các phương án A, B, C, D.	(list)
4	**sList**	Dãy 4 giá trị chỉ ra phương án nào đúng, phương án nào sai. Theo qui định giá trị = 0 (sai), = 1 (đúng).	(list)

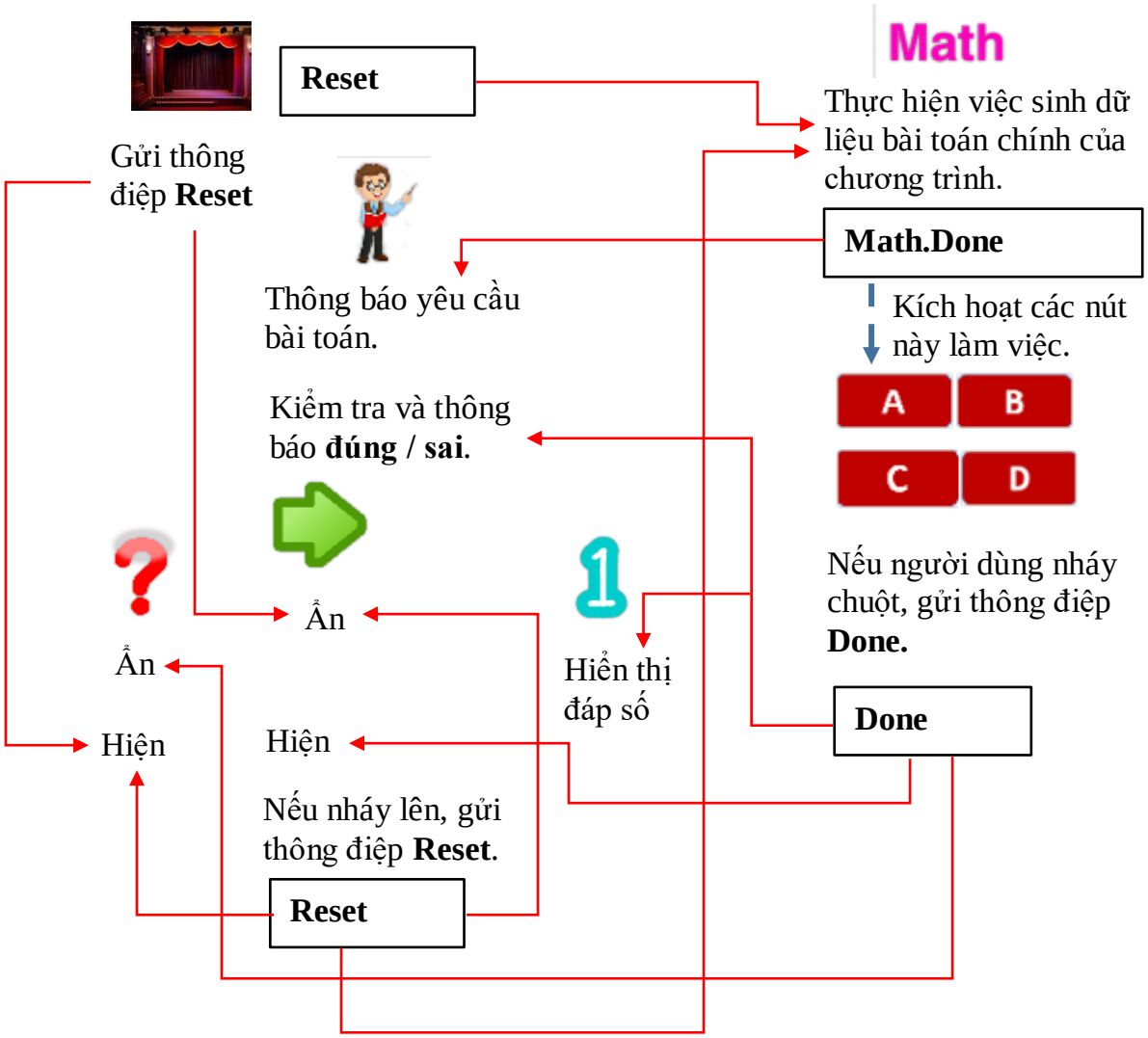
Sơ đồ thiết kế dữ liệu chính của chương trình.



Các nhân vật của chương trình được thiết kế trong bảng sau:

Biểu tượng	Tên nhân vật	Chức năng và hoạt động	Các chú ý khác
Math	Đối tượng **Math**	Đối tượng này được xây dựng đặc biệt để lưu trữ tất cả các dữ liệu và thuật toán sinh tự động các bài toán và dữ liệu của chương trình.	Bộ dữ liệu chính lưu dữ liệu của các thuật toán sẽ là dữ liệu riêng (local) của đối tượng này.
	Giáo viên	Giáo viên đóng vai trò là người dẫn chương trình và thực hiện giao tiếp với người chơi.	
A B C D	Các nút A, B, C, D	Các nút dành cho người chơi lựa chọn đáp án của trò chơi. Mỗi lần chỉ được nháy chuột 1 lần lên 1 nút lệnh này.	Nháy lên nút này sẽ sinh ra thông điệp **Done**.
	Nút Câu hỏi (**Question**)	Nhân vật này chỉ có nhiệm vụ hiển thị như 1 dấu hỏi tại vị trí cần điền số. Khi người dùng đã chọn đáp án, hình ảnh này sẽ ẩn đi, và tại vị trí này sẽ hiện đáp án đúng của bài toán.	
	Nút **Next**	Nút này sẽ chỉ hiện khi người chơi đã làm xong 1 bài. Khi người dùng nháy lên nút này thì sẽ bắt đầu 1 bài toán tiếp theo.	Nháy lên nút này sẽ sinh ra thông điệp **Reset**.
	Đối tượng Number	Nhân vật này có nhiệm vụ hiển thị đáp số của bài toán sau khi người dùng nháy chuột nhập đáp án. Sử dụng kỹ thuật hiển thị số bằng chữ số đã có trong bài học 21 (một số kỹ thuật thiết kế games),	Đối tượng này có 10 trang phục với hình ảnh các số từ 0 đến 9. Tên của trang phục cũng là 0 đến 9.

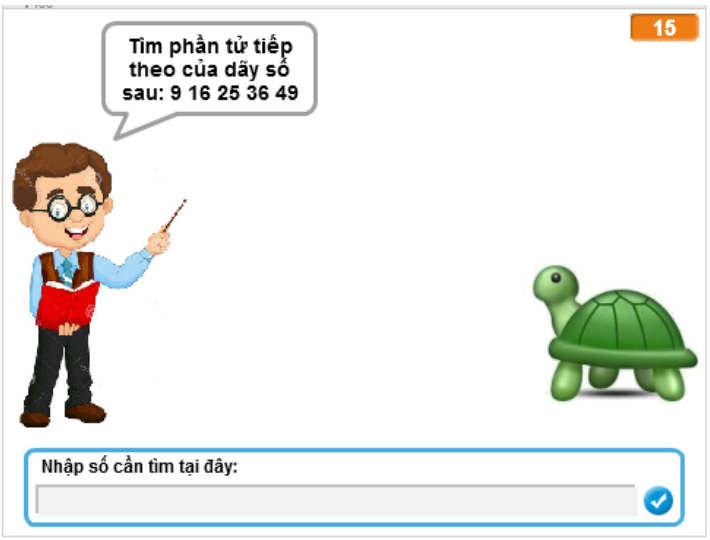
Sơ đồ hoạt động của chương trình.



2. Trò chơi tìm số

Tim so.sb2

Trong hoạt động này chúng ta cùng làm quen và thiết kế 1 chương trình, trò chơi khác nhưng cũng dựa trên việc sinh tự động các bài toán. Điểm khác biệt quan trọng là trong chương trình này, việc giải toán chỉ cần nhập 1 đáp án (là số) duy nhất.



Chương trình (thông qua giáo viên) sẽ liên tục đưa ra bài toán, người chơi nhập đáp số thông qua hộp hội thoại nhập số. Con rùa xuất hiện để hỗ trợ người chơi.

Nhân vật của chương trình.

Yêu cầu chức năng của chương trình như sau:

- Chương trình sẽ liên tục sinh và đưa ra màn hình bài toán, yêu cầu người chơi nhập 1 đáp số duy nhất. Giáo viên luôn hiện đề bài trên màn hình. Người chơi tương tác với chương trình thông qua dòng nhập số ở phía dưới.

- Trên màn hình có hình con rùa. Nháy lên con rùa để xem gợi ý.

- Người chơi phải nhập liên tục đáp số cho đến khi đúng thì thôi. Nếu nhập sai bị trừ điểm, nếu đúng được thưởng điểm.

- Nếu nhập đúng, GV sẽ thông báo ngay là đúng, nhân vật **Next** xuất hiện. Bấm vào nút **Next** để tiếp tục bài toán khác.

Tương tự như chương trình Điền số vào dãy (mục 1), chúng ta sẽ đưa tất cả các thuật toán sinh đề bài vào trong 1 nhân vật có tên là Math. Các thuật toán này sẽ được lập trình theo từng dạng. Ví dụ.

Các bài toán có thể đưa vào trò chơi này.

Stt	Tên bài toán	Diễn dải bài toán
1	Tính ước số chung lớn nhất.	Hãy tìm ước số chung lớn nhất của các số <N1> và <N2>.
2	Tính bội số chung nhỏ nhất.	Hãy tìm bội số chung nhỏ nhất của các số <N1> và <N2>.
3	Đếm số các ước số thực sự.	Đếm số các ước số thực sự (kể cả 1) của số <N>
4	Tính nhanh 1 phép toán +, - đơn giản.	Ví dụ: Tính nhanh tổng của 3 số <N1>, <N2>, <N3>. Tính nhanh tích của 2 số <N1> và <N2>.
5	Tính phần tử tiếp theo của 1 dãy số có qui luật.	Tìm phần tử tiếp theo của dãy số sau: <N1>, <N2>, <N3>, <N4>.
6	Chuyển đổi số thập phân sang nhị phân.	Hãy chuyển đổi số thập phân sau <N> sang số nhị phân.
7	Chuyển đổi số nhị phân sang thập phân.	Hãy chuyển đổi số nhị phân sau <N> sang số thập phân.
8	Tìm giá trị Max, Min của 1 dãy số cho trước.	Tìm giá trị lớn (nhỏ) nhất của dãy số sau: <N1> <N2> ….. <Nk>.

Bộ dữ liệu sau dùng để lưu trữ dữ liệu cho tất cả các thuật toán trên. Bộ dữ liệu này được thiết kế trong đối tượng **Math** của chương trình.

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**Problem.Content**	Nội dung bài toán.	(biến nhớ)
2	**Problem.Hint**	Gợi ý của bài toán.	(biến nhớ)
3	**Problem.Answer**	Đáp số của bài toán.	(biến nhớ)

Chú ý: Các bài toán sinh dãy số trong ví dụ ở mục 1 có thể được sử dụng lại trong bài này. Đoạn chương trình sau mô tả cách chuyển đổi dữ liệu đã sinh bài toán điền số vào dãy số dạng trắc nghiệm sang dạng bài toán có 1 đáp số duy nhất.

Cách chuyển đổi như sau:

Problem.Hint: nhập trực tiếp lời gợi ý cách giải bài toán.

Problem.Answer = giá trị phần tử cuối cùng của dãy NumList.

Problem.Content = "Tìm phần tử tiếp theo của dãy số sau: " + dãy các phần tử của NumList.

Thiết kế theo module

Cách thiết kế nhân vật Math trong 2 ví dụ trên, trong nhân vật này chứa tất cả các thuật toán sinh dữ liệu của các bài toán chính, được gọi là thiết kế theo module. Chúng ta có thể gọi là **Math module**.

Em hãy thực hiện tiếp các công việc sau:

- Trình bày sơ đồ hoạt động của chương trình này.

- Viết các thuật toán của module Math.

- Hoàn thiện chương trình theo sơ đồ của em thỏa mãn yêu cầu đặt ra của chương trình.

- Thực hiện tiếp các bài tập mở rộng.

3. Bài toán và trò chơi vẽ hình mẫu

Chúng ta quay trở lại chương trình Vẽ tự do, mục 1 của bài học 22. Trong chương trình của bài học trước, chúng ta đã mô phỏng 1 bút vẽ tự do với màu sắc và độ rộng bút có thể chọn ngay trên màn hình.

Bây giờ chúng ta sẽ mở rộng ra để bổ sung thêm 1 chức năng đặc biệt: vẽ theo các hình mẫu có sẵn.

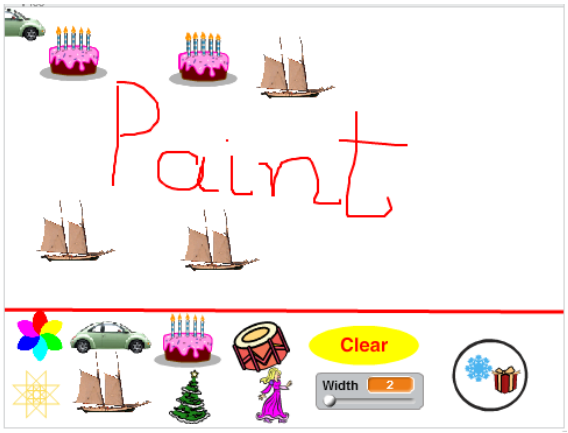
Như vậy chương trình sẽ có 2 chế độ làm việc độc lập. Biến nhớ Pain_Type dùng để phân biệt 2 chế độ này.

- Với Paint_Type = 1, chế độ vẽ tự do như em đã biết trong bài học trước.

- Với Pain_Type = 2, sẽ chuyển sang chế độ vẽ theo hình mẫu. Trên màn hình, khu vực công cụ, các màu vẽ sẽ được thay thế bằng các hình mẫu. Nháy chuột lên 1 hình mẫu, sau đó nháy lên màn hình để thực hiện vẽ theo hình mẫu đã chọn.

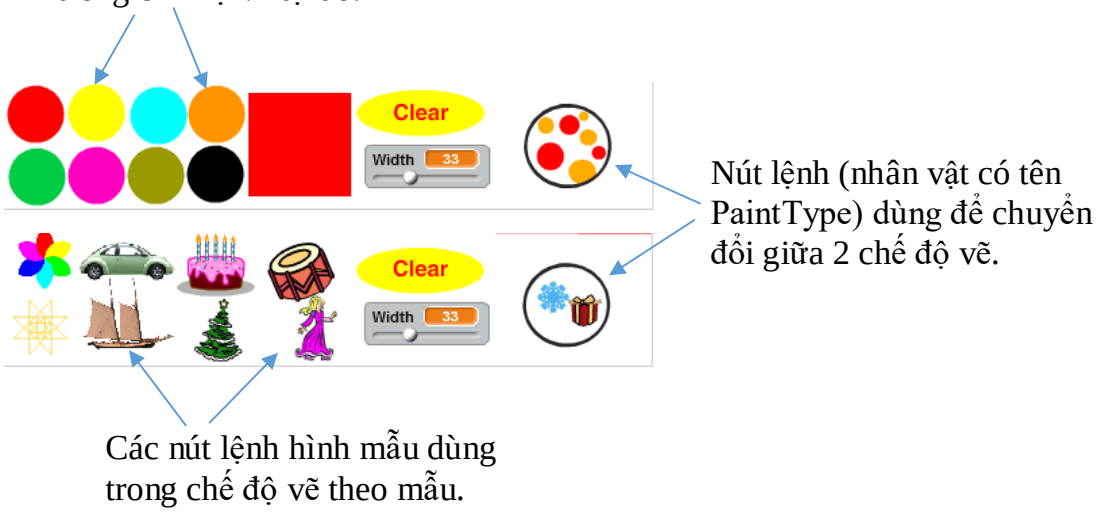
Ve nang cao.sb2

Chế độ vẽ tự do với bút vẽ và màu sắc, **PaintType=1**.

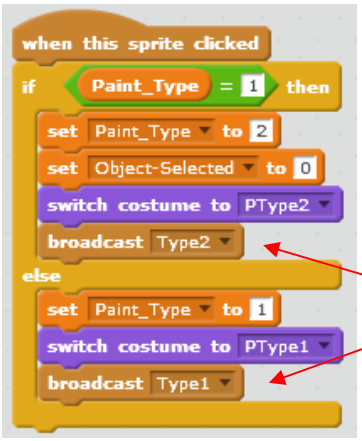


Chế độ vẽ theo các mẫu hình có sẵn, **PaintType=2**.

Các hình mẫu sẽ được bổ sung vào chương trình như các nhân vật. Chúng ta cần 1 nút lệnh (nhân vật) nữa để chuyển đổi giữa 2 chế độ vẽ (vẽ tự do và vẽ theo mẫu) của chương trình.



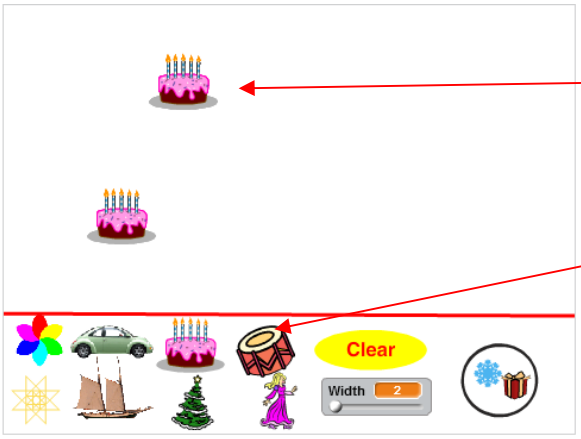
Đoạn chương trình sau mô tả đoạn chương trình của nhân vật Paint_Type dùng để chuyển đổi giữa 2 chế độ vẽ. Nháy chuột lên nút này để thay đổi chế độ vẽ.



Các thông điệp **Type1** và **Type2** có tác dụng làm cho các nhân vật tương ứng với 2 chế độ vẽ này tự động ẩn, hiện đúng lúc.

Kỹ thuật vẽ hình theo mẫu - nhân vật

Chúng ta sẽ mô tả nhanh kỹ thuật vẽ theo hình mẫu là 1 nhân vật như sau.

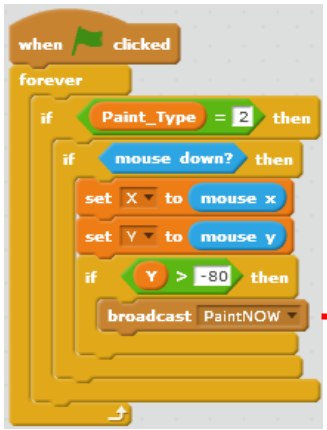


Bước 2: nháy chuột lên màn hình để vẽ hình này. Có thể vẽ liên tục nhiều hình.

Bước 1: nháy chuột lên nhân vật để xác định hình muốn vẽ.

- Khi nháy chuột lên nhân vật, đoạn chương trình sau sẽ gán cho biến tổng thể Object-Selected 1 giá trị duy nhất xác định đúng ID của nhân vật này.

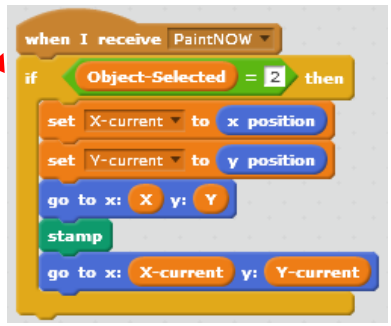
Khi nháy chuột lần tiếp theo trên màn hình 2 đoạn chương trình sau sẽ thực hiện việc vẽ đúng nhân vật này bằng lệnh **stamp**.



Khi nháy chuột lên màn hình sẽ ghi lại tọa độ vị trí chuột vào biến X, Y.

Gửi thông điệp cho nhân vật tương ứng.

Chương trình của nhân vật Bút vẽ (Pencil).



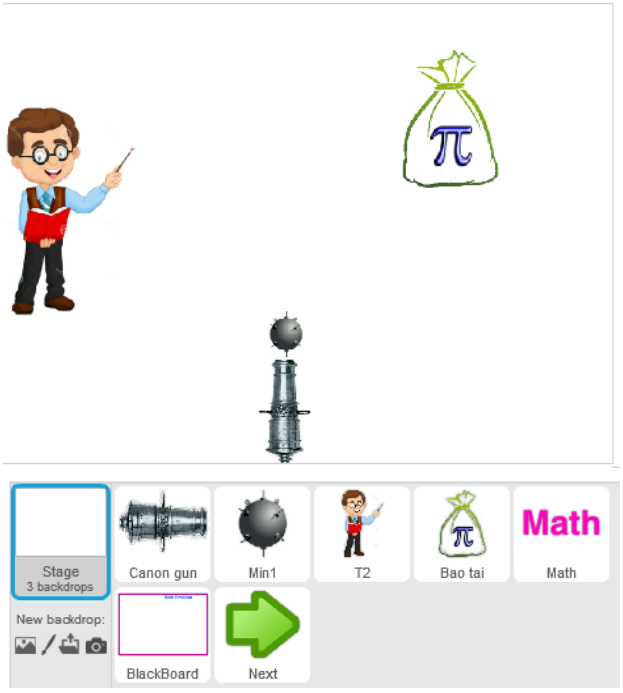
Chương trình của nhân vật hình vẽ: di chuyển tới điểm (X, Y) và thực hiện **stamp**.

4. Trò chơi bắn súng giải toán

Ý tưởng **vừa chơi vừa học**.

Chúng ta sẽ cùng thiết kế thêm 1 trò chơi "giải toán" nữa nhưng theo một cách khác, hấp dẫn hơn. Người chơi sẽ được "chơi" trước khi phải "giải toán". Ý tưởng chung của tất cả các phần mềm kiểu này là "vừa chơi, vừa học", hay cụ thể hơn là: trong khi chơi, để vượt qua các chướng ngại vật, người chơi bắt buộc phải giải toán. Chúng ta sẽ hiện thực hóa ý tưởng phần chơi là "bắn súng", còn phần học là "giải toán có đáp số" như trong ví dụ trên.

Ban sung giai toan.sb2

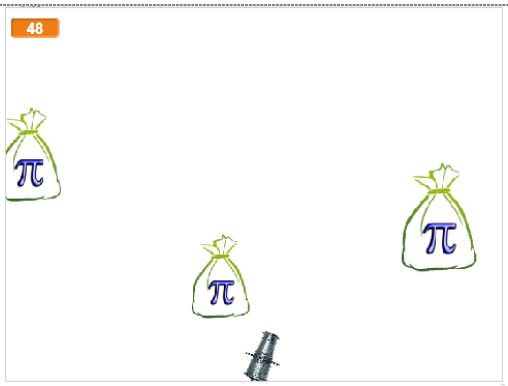


Ý tưởng thiết kế của chương trình là:

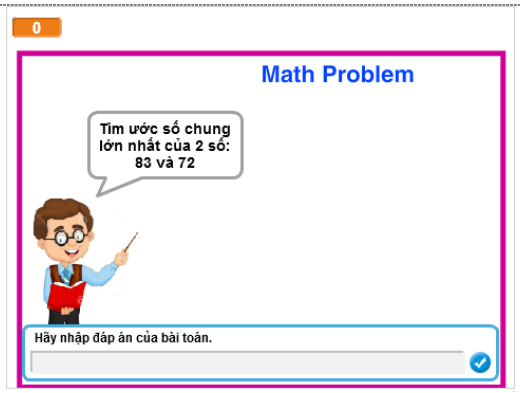
Người chơi cần bắn súng trúng vào các bao tải tiền để được thưởng điểm, nhưng muốn có điểm thì phải giải 1 bài toán đúng đã.

Module Math được thiết kế tương tự như các ví dụ đã biết.

Giao diện của chương trình được chia làm 2 giai đoạn (2 pha): chơi và học.



Phần chơi của chương trình: cần bắn súng trúng vào các bao tải tiền. Bao tải càng nhỏ, điểm số sẽ càng cao.



Phần giải toán của chương trình: cần nhập đáp số đúng cho bài toán mà GV đang yêu cầu làm trên màn hình.

Mô tả hoạt động của chương trình.

- Chương trình bắt đầu từ trạng thái "chơi" (StartMath = 0). Người dùng điều khiển súng bằng các phím phải, trái để quay hướng nòng. Bấm phím Space để bắn.

- Từ phía trên các bao tải tiền (có ký hiệu π) rơi xuống với kích thước khác nhau. Kích thước càng nhỏ, càng khó bắn trúng sẽ có điểm thưởng cao hơn. Nếu bị đạn

bắn trúng, lập tức toàn bộ chương trình chuyển sang chế độ "giải toán" (StartMath =1), các nhân vật như súng, đạn, bao tải sẽ biến mất.

- Khi chuyển sang chế độ giải toán, xuất hiện 1 bảng đen, GV sẽ xuất hiện để điều khiển tương tác với người chơi. GV thể hiện yêu cầu bài toán ngay trên bảng.

- Người chơi cần nhập đáp số của bài toán, nhập liên tục cho đến khi đúng thì thôi. Nếu nhập sai sẽ bị trừ 1 điểm. Nếu nhập đúng, sẽ được thưởng số điểm = CurrMathPoint, khi đó xuất hiện nút Next. Bấm vào nút này sẽ chuyển ngay sang chế độ chơi.

Một số biến nhớ quan trọng của chương trình.

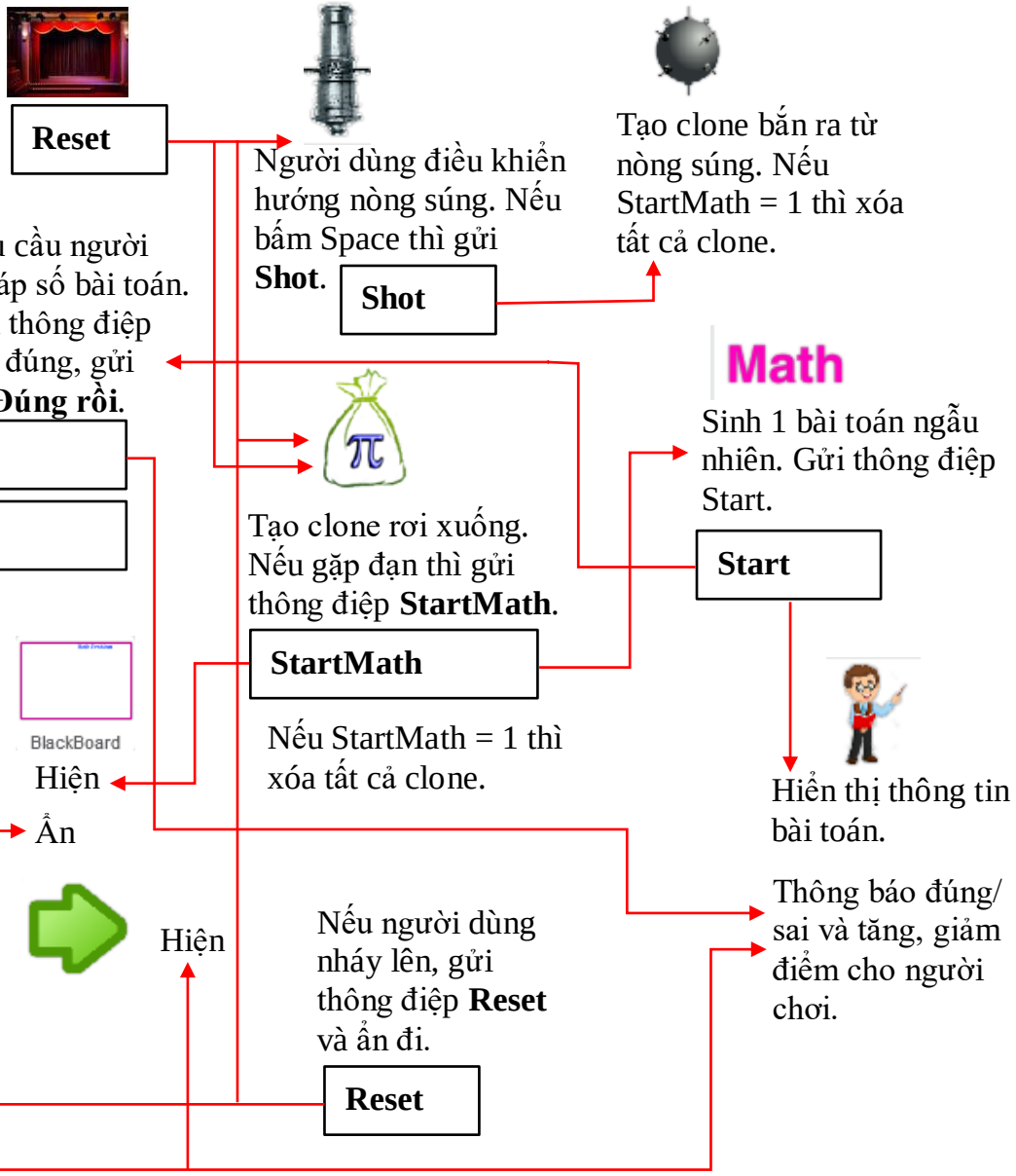
Stt	Tên biến nhớ	Ý nghĩa
1	**StartMath**	Biến nhớ chỉ ra trạng thái hoạt động chính của chương trình. = 0, trạng thái "chơi", bắn súng vào bao tải. = 1, trạng thái giải toán.
2	**CurrMathPoint**	Biến nhớ dùng để lưu giá trị điểm số sẽ được thưởng. Khi người chơi bắn trúng 1 bao tải, giá trị này được xác định và lưu trữ lại. Khi người chơi giải được bài toán thì mới thưởng điểm.

Thiết kế danh sách nhân vật.

Stt	Biểu tượng	Ý nghĩa, hoạt động của nhân vật
1		Súng thần công. Khi người dùng nhấn phím Space thì sẽ phát thông điệp **Shot**.
2		**Đạn.** Khi nhận thông điệp **Shot** sẽ xuất hiện tại đầu nòng súng như 1 clone và dịch chuyển theo hướng nòng súng cho đến khi gặp cạnh màn hình hoặc nếu nhận thông điệp StartMath thì tự xóa.
3		**Bao tải tiền.** Bao tải tiền xuất hiện như các clone với kích thước = **size** là % so với gốc. Giá trị size sẽ lấy ngẫu nhiên từ 10-50%. Nếu va chạm với đạn thì lập tức tính toán **CurrMathPoint**, gửi thông điệp **StartMath**, thiết lập biến nhớ **StartMath = 0** và xóa tất cả các clone trên màn hình.
4		Đối tượng này được xây dựng đặc biệt để lưu trữ tất cả các dữ liệu và thuật toán sinh tự động các bài toán và dữ liệu của chương trình. Bộ dữ liệu của module Math như sau: **Problem.Hint:** Gợi ý bằng lời cách giải bài toán. **Problem.Answer:** Đáp số duy nhất của bài toán. **Problem.Content:** Nội dung bằng lời của bài toán.

Stt	Biểu tượng	Ý nghĩa, hoạt động của nhân vật
5		**Giáo viên.** Giáo viên xuất hiện khi nhận được thông điệp **StartMath**, sau đó sẽ điều khiển tương tác với người chơi để giải toán.
6		Nút lệnh **Next**. Nút này xuất hiện khi nhận thông điệp **Đúng rồi**. Khi nháy lên nút này sẽ phát thông điệp **Reset**.
7		**Bảng làm toán.** Bảng này xuất hiện khi nhận thông điệp **StartMath**. Bảng sẽ ẩn đi nếu nhận thông điệp **Reset**.

Sơ đồ hoạt động của chương trình.

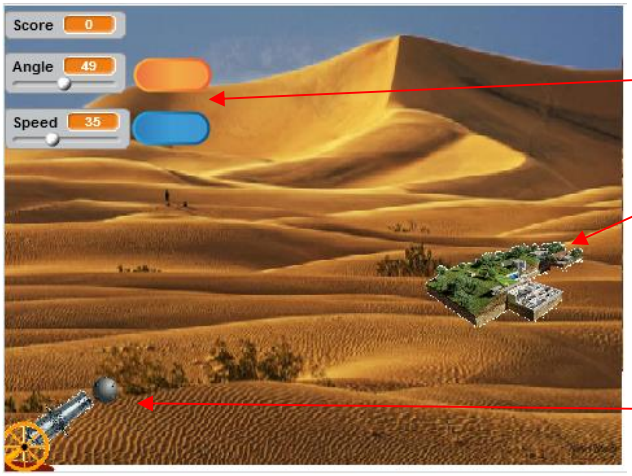


Em hãy hoàn thiện chương trình trên và làm các bài tập bổ sung.

5. Trò chơi bắn súng pháo binh

Trong hoạt động này chúng ta sẽ thiết kế trò chơi mô tả bắn súng pháo binh tiêu diệt mục tiêu cho trước. Yêu cầu của chương trình là mô phỏng đường đi của viên đạn chính xác gần như thực tế. Giao diện của trò chơi sẽ như hình sau.

Phao binh.sb2

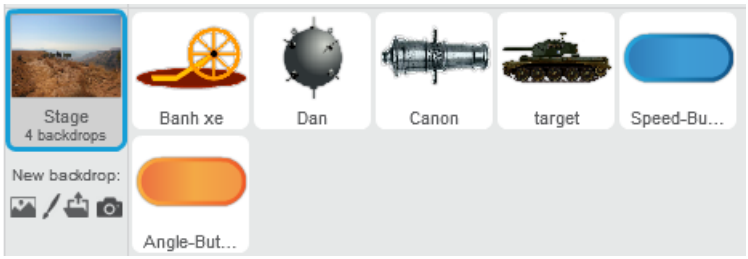


2 thông số quan trọng: góc quay (Angle) của súng và vận tốc đạn (Speed) có thể điều chỉnh trực tiếp.

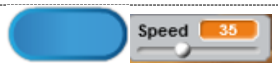
Mục tiêu cần bắn trúng.

Vị trí của súng và đạn.

Danh sách nhân vật của trò chơi.



Mô tả chi tiết nhân vật.

Stt	Biểu tượng	Ý nghĩa, hoạt động của nhân vật
1		**Súng Canon.** Người dùng điều khiển góc (hướng) và vận tốc ban đầu của đạn bằng phím và nút lệnh ngay trên màn hình. Bấm Space để ra lệnh bắn (Fire).
2		**Đạn.** Đạn được điều khiển theo 1 chương trình đặc biệt để mô phỏng chính xác quĩ đạo bay của viên đạn. Nếu đạn rơi trúng mục tiêu thì chiến thắng, được tăng 50 điểm, nếu trượt thì bị trừ 7 điểm.
3		**Mục tiêu tác chiến.** Nhân vật này có nhiều trang phục. Nếu đạn bay gặp mục tiêu thì phát ra tiếng nổ, đạn biến thành gọn lửa.
4		**Bánh xe.** Bánh xe chỉ có ý nghĩa trang trí. Chú ý tâm của bánh xe chính là tâm quay của súng và là vị trí xuất phát của viên đạn.
5		Nút nhập trực tiếp vận tốc.
6		Nút nhập trực tiếp góc.

Mô tả hoạt động của chương trình.

- Chương trình sẽ tự động thiết lập màn hình nền và vị trí của mục tiêu trên màn hình.

- Người dùng được điều chỉnh vận tốc đạn và góc quay của súng, có thể điều chỉnh trực tiếp trên thanh trượt của biến nhớ hoặc nháy nút để nhập chính xác 1 số.

- Bấm phím Space để thực hiện lệnh bắn (File).

- Viên đạn sẽ bắn ra khỏi nòng súng và bay trên 1 quĩ đạo chính xác trên màn hình.

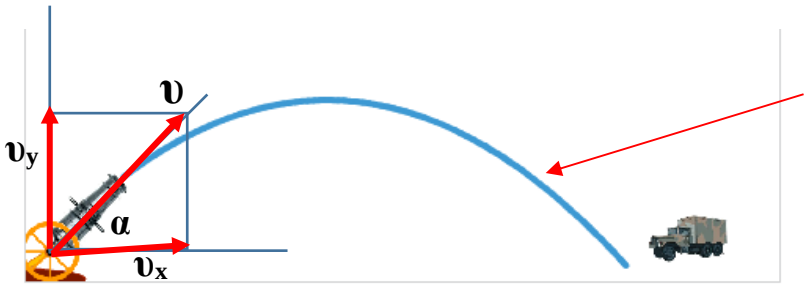
- Nếu trúng mục tiêu sẽ nghe tiếng nổ và viên đạn bốc lửa như hình sau.

- Nếu bắn trượt bị trừ 7 điểm, nếu bắn trúng được thưởng 50 điểm.

Như vậy thuật toán quan trọng nhất của trò chơi này là thiết lập quĩ đạo chính xác của viên đạn khi ra khỏi nòng súng.

Quĩ đạo viên đạn

2 tham số quan trọng của chương trình: Góc hướng súng (Angle = **α**) và Vận tốc viên đạn (Speed = **υ** m/s).



Đường đi - quĩ đạo của viên đạn khi ra khỏi nòng súng.

Vận tốc ban đầu được phân bổ thành 2 thành phần υ_x và υ_y.

$$\upsilon_x = \upsilon.\cos(\alpha);\ \upsilon_y = \upsilon.\sin(\alpha)$$

quĩ đạo chuyển động của viên đạn tính theo thời gian t như sau:

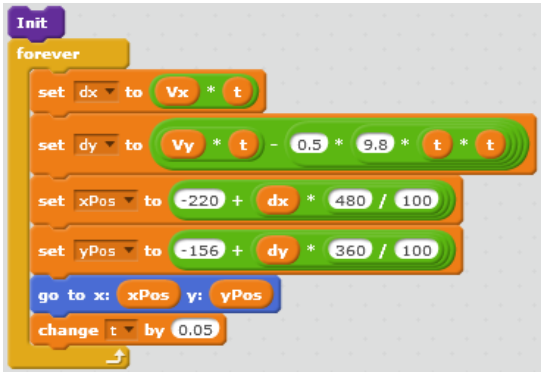
$$x(t) = \upsilon_x.t;\ y(t) = \upsilon_y.t - \frac{g.t^2}{2}$$

Ở đây t tính bằng giây (s), g = 9.8 m/s^2 ; vận tốc tính bằng m/s.

Từ các công thức trên chúng ta sẽ tính được tọa độ và vẽ được đường đi của viên đạn với vận tốc ban đầu = Speed (υ) và góc ban đầu = Angle (α).

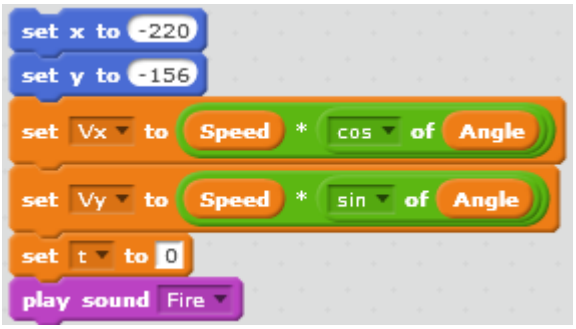
Viên đạn sẽ được lập trình chuyển động khi người dùng bấm phím Space, thông điệp **Fire** được gửi từ nhân vật Canon.

Đoạn chương trình chính điều khiển viên đạn bắn ra khỏi nòng súng.



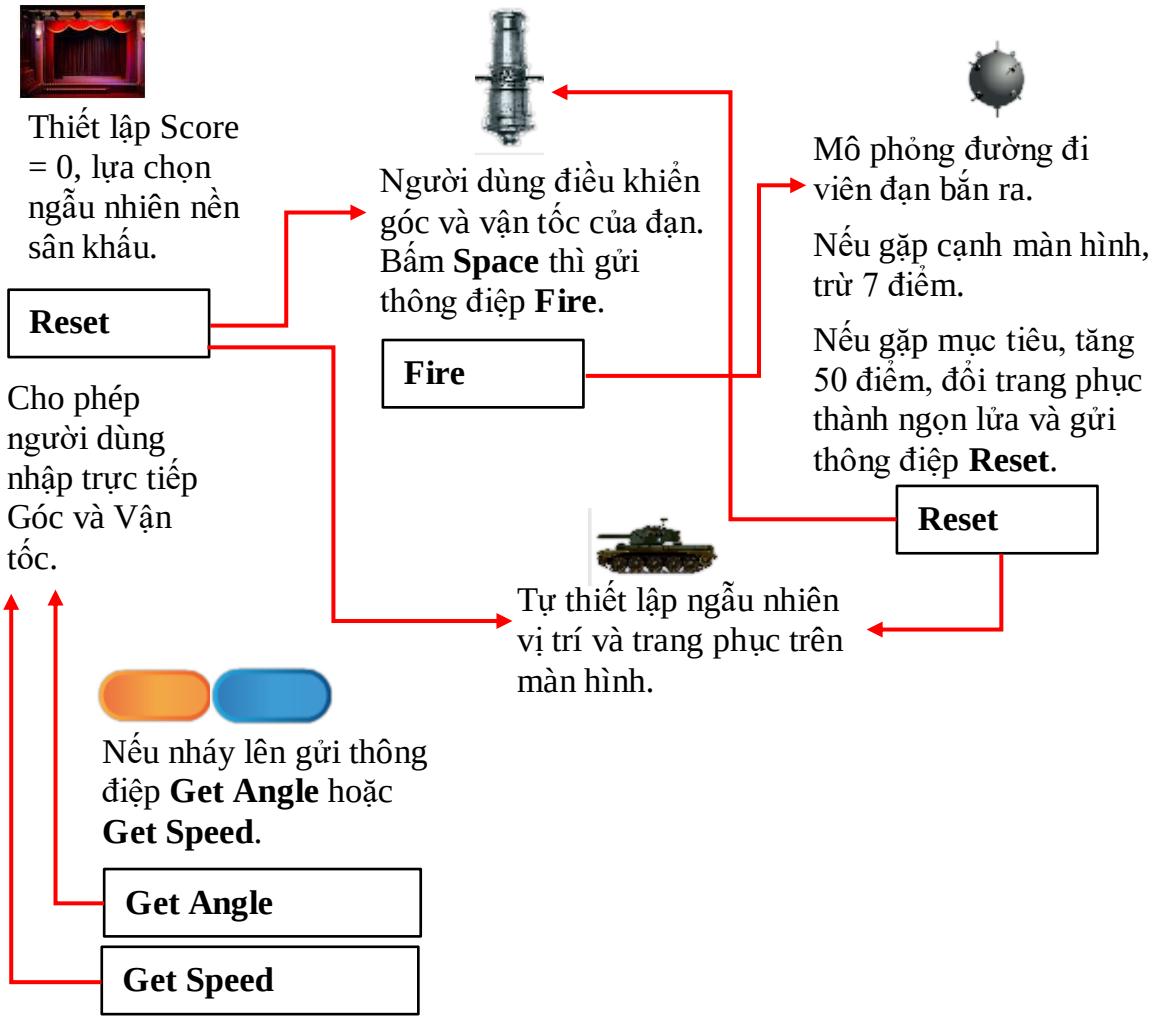
Chú ý: cần lập trình để khi đạn đi tới biên của màn hình thì chương trình dừng lại.

Thủ tục **Init** thiết lập các tham số ban đầu của viên đạn bao gồm các vận tốc thành phần (υ_x, υ_y) và thời gian t = 0.



Chú ý: tọa độ (-220, -156) là điểm gốc của súng, từ đó viên đạn được bắn ra.

Sơ đồ hoạt động của chương trình



Một số chú ý khi viết chương trình chi tiết.

(a) Chú ý giá trị góc **Angle**.

Trong Scratch qui định hướng thẳng đứng lên trên là góc 0 độ, nhưng góc Angle của bài toán mô phỏng là góc so với trục ngang X, do vậy khi thực hiện các thao tác với số đo góc này cần thực hiện như hình sau.

Các chương trình của nhân vận **Súng Canon**.

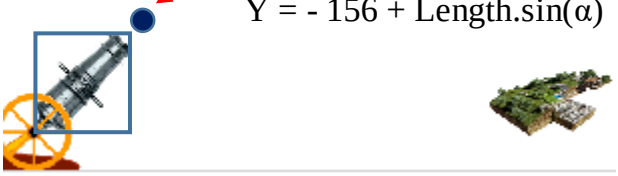
Góc của lệnh Point in direction phải là **90-Angle**.

Phím trái sẽ tăng Angle lên 1 độ, phím phải sẽ giảm Angle xuống 1 độ.

(b) Vị trí xuất phát của viên đạn.

Theo phân tích trên thì vị trí viên đạn sẽ xuất phát từ điểm (-220, -156) tức là từ gốc của nòng súng (tâm bánh xe). Chúng ta có thể mô phỏng viên đạn phóng ra từ đầu nòng súng. Cách làm như sau:

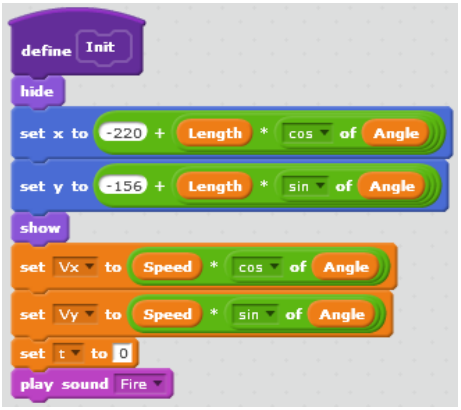
Gọi độ dài nòng súng là Length, khi đó tọa độ điểm đầu của nòng súng sẽ được mô tả như trong hình sau.



Vị trí này có tọa độ là:
X = -220 + Length.cos(α)
Y = - 156 + Length.sin(α)

Khi đó các đoạn chương trình mô phỏng chuyển động của viên đạn sẽ được sửa lại như sau:

Thủ tục **Init**.

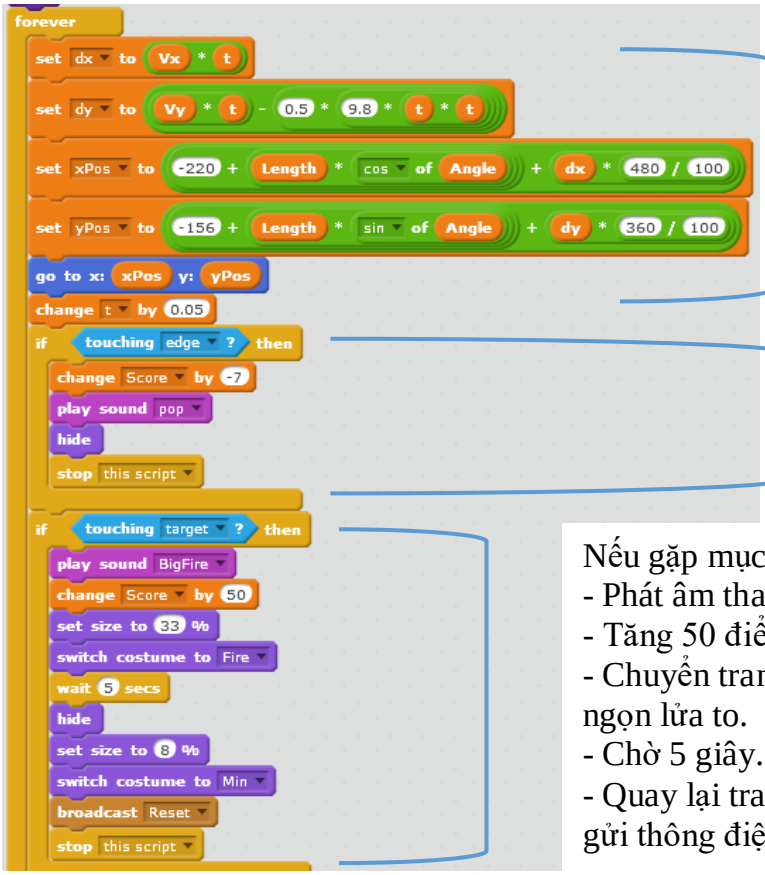


Thiết lập vị trí ban đầu của viên đạn ở đầu nòng súng.

Thiết lập các vector thành phần của vận tốc υ_x, υ_y.

Âm thanh tiếng nổ ở đầu nòng súng.

Chương trình chính của viên đạn khi nhận thông điệp Fire như sau (chú ý đã cắt bỏ hình ảnh lệnh Init):



Vòng lặp chính mô phỏng quĩ đạo viên đạn tính từ đầu nòng súng.

Khi gặp cạnh thì giảm 7 điểm, dừng chương trình.

Nếu gặp mục tiêu:
- Phát âm thanh nổ lớn.
- Tăng 50 điểm.
- Chuyển trang phục thành ngọn lửa to.
- Chờ 5 giây.
- Quay lại trang phục cũ và gửi thông điệp **Reset**.

Em hãy hoàn thiện chương trình trên.

Câu hỏi, bài tập

1. Hoàn thiện chương trình **Điền số vào dãy** đã thiết kế trong hoạt động 1 của bài học.

2. Mở rộng chương trình trên bằng cách kéo dài dãy số thành 9, các số của dãy được sắp xếp trên màn hình như 1 bảng 3 x 3. Như vậy chương trình sẽ được đặt tên lại là **Điền số vào bảng**.

3. Mở rộng **module Math**, bổ sung thêm các thuật toán sinh dãy số theo mô tả của bảng sau.

Stt	Mô tả qui luật dãy số	Mô tả ngắn gọn thuật toán
1	Dãy chỉ toàn các số lẻ.	Các số trong dãy được sinh ngẫu nhiên. Đáp án sai phải là số lẻ.
2	Dãy chỉ toàn các số chẵn.	Các số trong dãy được sinh ngẫu nhiên. Đáp án sai phải là số chẵn.
3	Dãy chỉ bao gồm các số chính phương.	Các số trong dãy được sinh ngẫu nhiên. Đáp án sai phải là các số không chính phương. Các số không chính phương có thể lấy từ các dạng

Stt	Mô tả qui luật dãy số	Mô tả ngắn gọn thuật toán
		sau: $n^2 + 1$, $n^2 - 1$, $n^2 + 2$, $n^2 - 2$ với n > 2.
4	Dãy chỉ bao gồm các số lập phương.	Các số trong dãy được sinh ngẫu nhiên. Đáp án sai phải là các số không là lập phương đúng. Các số không lập phương có thể lấy từ các dạng sau: $n^3 + 1$, $n^3 - 1$, $n^3 + 2$, $n^3 - 2$ với n > 1.
5	Dãy chỉ bao gồm các số tự nhiên chia hết cho 1 số k nào đó (ví dụ k=2, 3, …., 10).	Các số trong dãy được sinh ngẫu nhiên. Đáp án sai lấy từ các số trong dãy, sau đó + hoặc - 1.
6	Dãy chỉ bao gồm các số nguyên tố.	Để sinh ngẫu nhiên các số không là nguyên tố có thể lấy ngẫu nghiên các tích k1.k2 hoặc k1.k2.k3, ở đây k1, k2, k3 là các số tự nhiên > 1.

4. Mở rộng **module Math**, bổ sung thêm các thuật toán sinh bài toán có 1 đáp số duy nhất.

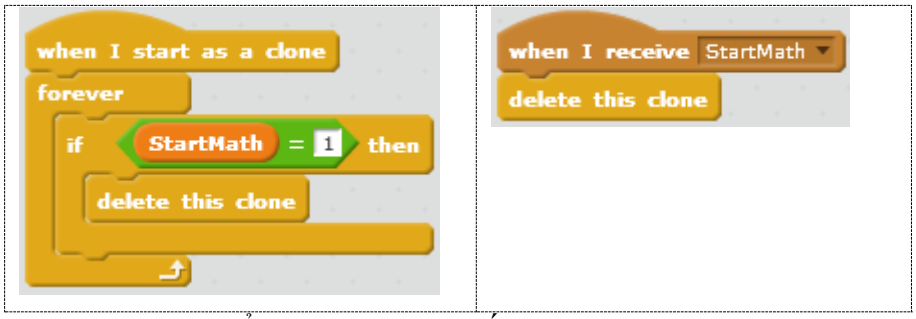
Stt	Mô tả bài toán	Mô tả ngắn gọn thuật toán
1	Tính trung bình của 1 dãy số cho trước.	Dãy số này được sinh ngẫu nhiên nhưng đáp số phải là số nguyên. Gọi tổng của N-1 số hạng đầu tiên là S, M = số tự nhiên làm tròn của S/(N-1). Khi đó phần từ thứ N được tính như sau: $A_N = N*M - S$. M chính là đáp số của bài toán.
2	Đếm số các ước số nguyên tố khác nhau của 1 số tự nhiên n cho trước.	Chú ý 1 không được coi là ước số nguyên tố. Vậy nếu n = 1 thì đáp án = 0.
3	Tính tổng các ước số thực sự của số tự nhiên N cho trước.	Chú ý: tổng này tính cả 1.
4	Tính N!	Bài toán có ý nghĩa khi N < 7.
5	Cho trước số tự nhiên N, tìm số nguyên tố nhỏ nhất sát với N.	Bài toán có ý nghĩa khi N < 100.
6	Tính nhẩm thương số nguyên của 2 số tự nhiên cho trước n_1, n_2.	

5. Mở rộng **module Math** đối với các thuật toán sinh tự động bài toán có 1 đáp số theo hướng: mở rộng thông tin gợi ý Hint theo nhiều bước, nếu tại bước này HS vẫn

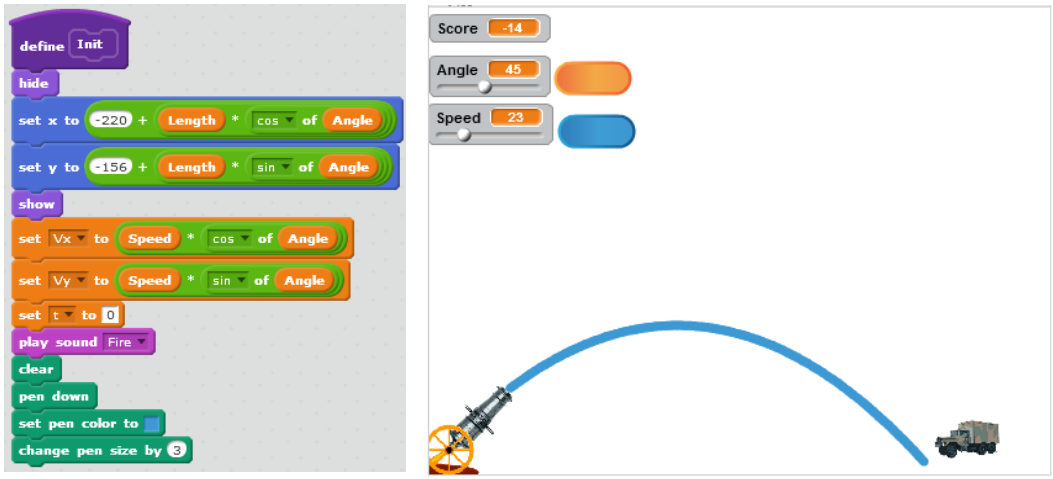
làm chưa đúng thì có thể tìm tiếp các gợi ý tiếp theo. Mỗi lần dùng thêm gợi ý sẽ bị trừ nhiều điểm hơn. Hãy thiết kế một mở rộng dữ liệu như vậy.

6. Mở rộng thiết kế và chương trình Bắn súng giải toán, bổ sung thêm nhân vật con rùa để giúp người chơi các gợi ý.

7. Em đã biết có 2 cách có thể truyền thông tin sự kiện từ nhân vật này sang nhân vật khác. Ví dụ trong chương trình bắn súng giải toán, 2 đoạn chương trình sau có tương đương nhau không?



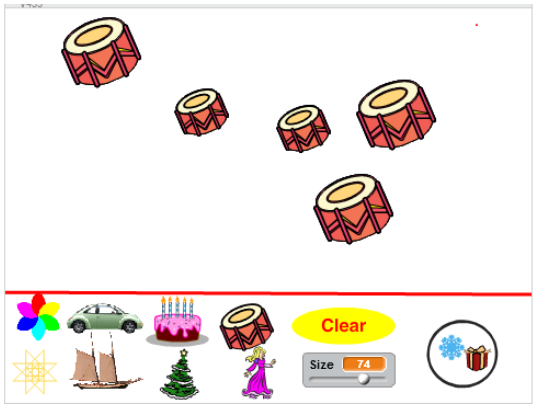
8. Em hãy thay đổi chương trình Bắn súng pháo binh, cho phép khi bắn súng, đạn sẽ vạch ra 1 quĩ đạo trên màn hình. Ví dụ có thể thay đổi 1 chút cho thủ tục **Init** của viên đạn khi bắt đầu bắn ra khỏi nòng.



9. Hoàn thiện chương trình **Vẽ hình theo mẫu** theo thiết kế của bài em đã học.

10. Mở rộng chương trình **Vẽ hình theo mẫu** như sau: bổ sung 1 tham số (ví dụ **Size**) để có thể chọn kích thước của hình muốn vẽ mẫu.

Ví dụ có thể vẽ được 1 hình mẫu với nhiều kích thước to nhỏ khác nhau.



Gợi ý:

Thiết lập thêm 1 biến nhớ tổng thể CurrSize dùng để lưu lại kích thước của hình mẫu trước khi vẽ theo Size mời.

Trước lệnh Stamp bổ sung 2 lệnh sau. Lệnh đầu để lưu lại kích thước hiện thời của mẫu, lệnh thứ 2 thiết lập kích thước mới theo Size.

Sau lệnh Stamp bổ sung 1 lệnh sau đây để trả lại cho hình mẫu kích thước ban đầu.

11. Em hãy bổ sung vào chương trình trên (bài tập 9) chức năng sau:

- Trong quá trình vẽ tự do, các phím trái, phải dùng để giảm, tăng giá trị **Width**.

- Trong quá trình vẽ theo mẫu, các phím trái, phải dùng để giảm, tăng giá trị **Size**.

12. Quan sát và cho nhận xét về thủ tục **Show Number** dưới đây. Có điểm gì đặc biệt trong thủ tục này.

13. Sử dụng công cụ Clone để thể hiện các số bằng hình ảnh trong bài tập trên trong các chương trình có yêu cầu thể hiện số. Em hãy xây dựng 1 bộ công cụ thủ tục thể hiện số bằng Clone, dựa vào ý tưởng của bài tập 12.

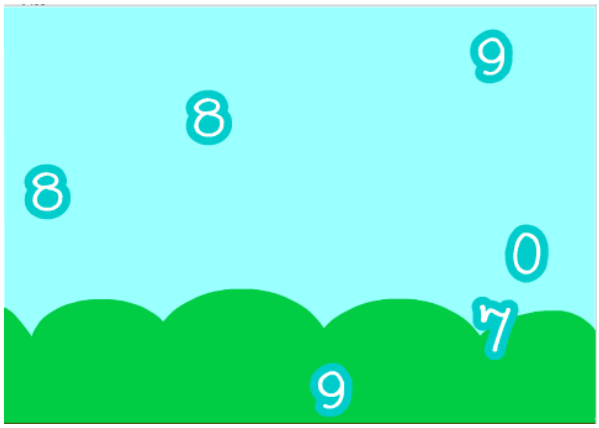
14. Nâng cấp chương trình Điền số vào dãy (hoạt động 1) bằng cách thay thế công cụ thể hiện số bằng vẽ (stamp) bằng công cụ dùng Clone.

Mở rộng

1. Thiết kế chương trình, trò chơi Mưa số (Number Rain) để rèn luyện nhanh tay nhanh mắt như sau.

Từ phía trên của màn hình sẽ có các số ngẫu nhiên rơi xuống. Người chơi cần nhanh tay, nhanh mắt nháy chuột lên các số này. Nếu nháy đúng các số này sẽ biến mất khỏi màn hình và người chơi được tăng điểm đúng bằng số thể hiện.

Giao diện của chương trình có thể như sau:



Các hình ảnh số này rơi xuống ngẫu nhiên từ phía trên với các giá trị khác nhau.

2. Mở rộng trò chơi trên bằng cách bổ sung thêm 2 chức năng sau:

- Khi nháy chuột lên 1 số, chương trình sẽ phát âm thanh cách đọc số đó (bằng tiếng Anh hoặc tiếng Việt) trước khi số đó bị xóa khỏi màn hình.

- Phía dưới màn hình luôn hiện chắc năng cho phép người chơi gõ 1 số từ bàn phím. Nếu gõ đúng 1 số có trên màn hình, số này cũng bị xóa đi khỏi màn hình.

3. Mở rộng chương trình Vẽ hình nâng cao (hoạt động 3) như sau:

Bổ sung các công cụ mới, mỗi công cụ cho phép bút chì sẽ vẽ 1 hình phức tạp trên màn hình theo các thông số mặc định cho trước. Ví dụ có thể đưa vào chương trình các công cụ vẽ sau:

- Vẽ hình tròn với tâm và bán kính cho trước.

- Vẽ hình sao 5 cánh biết tâm và bán kính.

- Vẽ các hình fractal phức tạp như cây cối, hoa tuyết, ….

Bài 25. Các trò chơi với chữ

Mục đích

Học sinh hiểu được thiết kế các trò chơi, chương trình của bài học và có thể hoàn thiện, mở rộng, phát triển theo ý tưởng riêng của mình.

Bắt đầu

Trong bài học này, chúng ta sẽ cùng học thiết kế các chương trình, trò chơi giáo dục liên quan đến chữ, tức là liên quan đến việc xử lý chữ, xâu ký tự và dãy xâu ký tự. Các bài toán, ý tưởng thiết kế trò chơi, phần mềm giáo dục dựa trên ký tự, chữ, xâu ký tự là rất phong phú. Cũng tương tự như bài học trước, chúng ta sẽ cùng xem xét, thiết kế 1 số trò chơi, chương trình cụ thể. Qua đó em sẽ học được nhiều ý tưởng mới cho các dự án mà em có thể tự sáng tạo sau này. Học xong bài học này, em sẽ hiểu ra 1 phần quan trọng của công thức mà các nhà khoa học máy tính hay viết:

Program = Data + Algorithm

Chương trình = Dữ liệu + Thuật toán.

Để có 1 chương trình tốt, ngoài ý tưởng kịch bản, các yếu tố cốt lõi là tổ chức dữ liệu và thiết kế logic thuật toán.

Nội dung bài học

1. Trò chơi ghép chữ

Trò chơi này rất đơn giản: từ 2 cụm từ rời nhau, người chơi cần ghép chúng lại để tạo thành 1 từ có ý nghĩa hoàn chỉnh. Khi ghép không được thay đổi thứ tự các chữ cái trong 1 cụm từ.

Ví dụ với 2 cụm từ: **ORK** và **WING** có thể ghép lại thành 1 từ **WORKING**.

Ghep chu.sb2

Nội dung câu hỏi do giáo viên công bố.

Các từ ban đầu cần ghép nối.

Vị trí nhập từ cần tìm tại đây.

Nhân vật của chương trình.

Yêu cầu của chương trình.

- Chương trình sẽ liên tục sinh tự động đầu vào của chương trình bao gồm 1 từ gốc cần tìm, và 2 từ đã được tách ra từ từ gốc để hiển thị trên màn hình.

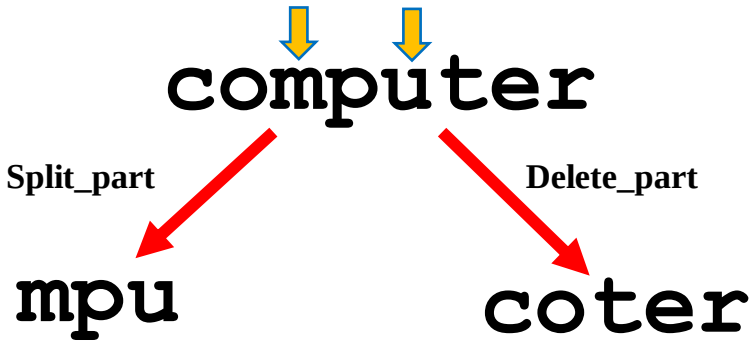
- Người chơi sẽ liên tục được yêu cầu nhập từ gốc cho đến khi nhập đúng.

- Nếu nhập đúng, chương trình thông báo chiến thắng, hiện ngay trên màn hình từ gốc, và sau 5 giây sẽ bắt đầu lại từ đầu.

Các biến nhớ chính và dữ liệu nguồn của chương trình.

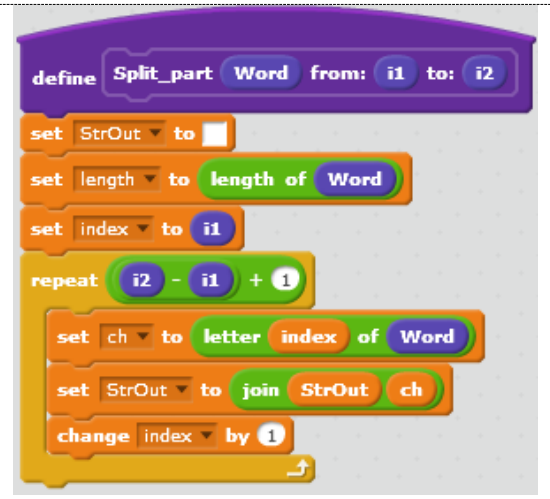
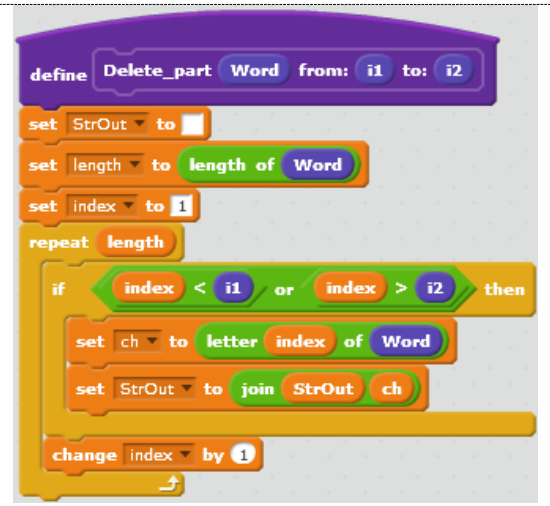
Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**W1, W2**	2 biến nhớ tổng thể, dùng để lưu 2 từ ban đầu cần ghép lại thành **Word**, được sinh ngẫu nhiên và là đầu vào chính của chương trình.	(biến nhớ)
2	**Word**	Biến nhớ tổng thể lưu từ cần tìm, là đầu vào chính của chương trình. Word sẽ được sinh ngẫu nhiên từ mảng dữ liệu **WList**.	(biến nhớ)
3	**Wdescription**	Mô tả ý nghĩa của từ Word. Thông tin này sẽ lấy từ mảng **WList.Description.**	(biến nhớ)
4	**WList**	Dữ liệu các từ gốc có thể lấy làm từ mẫu của trò chơi.	(list)
5	**WList.Description**	Mô tả của các từ trong dãy **WList**.	(list)

Thuật toán chính của chương trình là việc tách từ gốc Word thành 2 từ W1, W2 để người chơi ghép từ. Có nhiều lời giải cho bài toán này. Tại đây tôi sẽ cùng các bạn thực hiện 1 lời giải đó, thông qua 2 thủ tục **Split_part** và **Delete_part**.

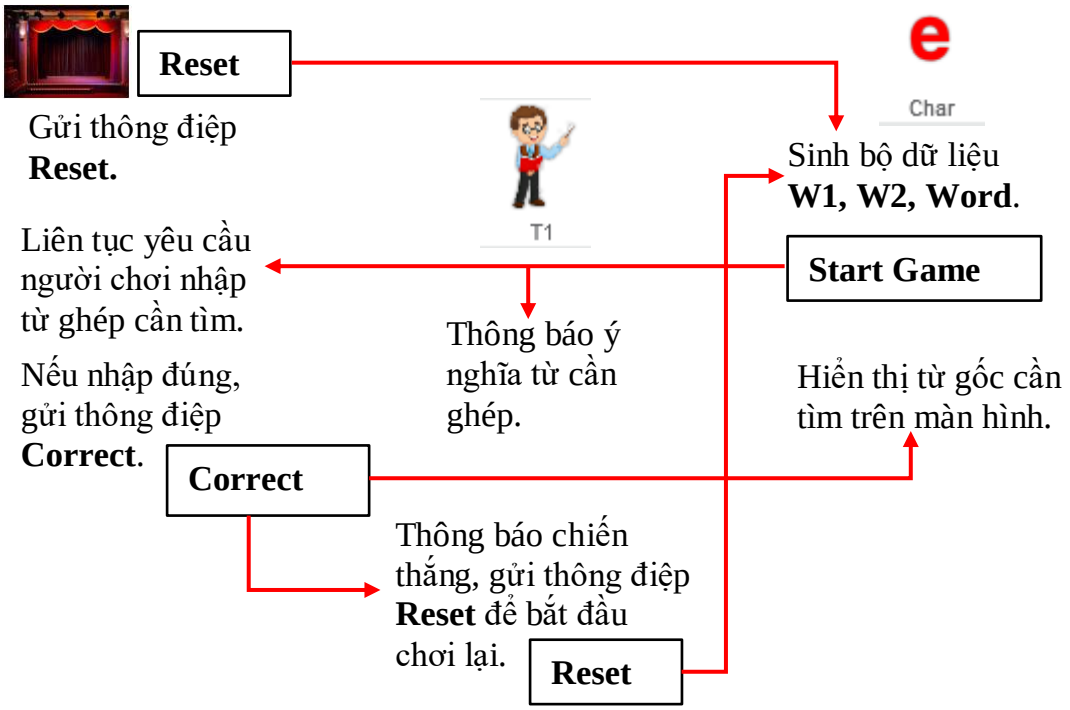


Thủ tục **Split_part(Word, t1, t2)** có chức năng lấy ra 1 xâu con của Word, bắt đầu từ ký tự thứ t1 đến t2. Kết quả lưu vào biến **StrOut**.

Thủ tục **Delete_part(Word, t1, t2)** có chức năng xóa đi 1 xâu con của Word, bắt đầu từ ký tự thứ t1 đến t2. Kết quả lưu vào biến **StrOut**.

Thủ tục Split_part()	Thủ tục Delete_part()
	

Sơ đồ hoạt động của chương trình như sau:



Em hãy hoàn thiện chương trình và thực hiện các bài tập, mở rộng bổ sung cho chương trình.

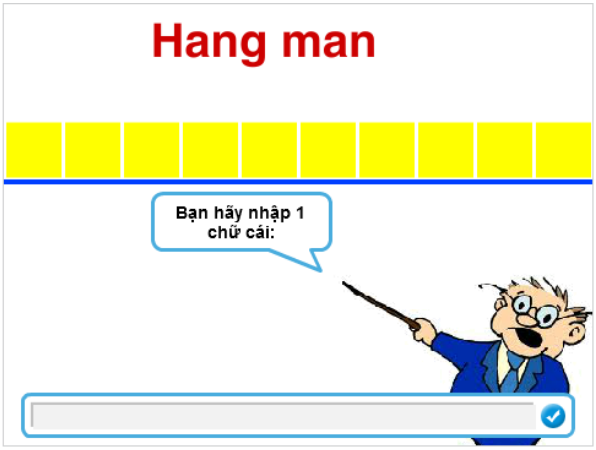
2. Trò chơi hangman (tổng quát)

Chúng ta sẽ quay lại 1 trò chơi rất nổi tiếng mà em đã làm quen trong bài học 15 (xử lý xâu ký tự 2), đó là trò chơi Tìm từ **Hangman**.

Lần này chúng ta sẽ thiết kế trò chơi tổng quát hơn và cho phép người thiết kế có thể mở rộng nhiều trong tương lai.

Giao diện chương trình mới Hangman lần này sẽ có dạng như hình sau:

Hangman Pro.sb2



Giao diện và cách chơi trò chơi này tương tự như các phần mềm Hangman khác, điểm khác là:

- Sử dụng các hình vuông che khuất chữ bên dưới.

- Độ dài của từ cần tìm có thể tùy ý.

Các nhân vật sử dụng trong chương trình.

Các yêu cầu cụ thể của chương trình.

- Chương trình sẽ liên tục sinh tự động bộ dữ liệu cho trò chơi hangman.

- Khi chơi, máy tính sẽ liên tục yêu cầu người chơi nhập 1 chữ cái, nếu chữ cái đó có trong từ cần tìm, phần mềm sẽ thông báo, ví dụ: "Có 2 chữ cái q" và chương trình sẽ tự động xóa 2 ô vàng để hiển thị chữ cái q bên dưới.

- Nếu người chơi nhập chữ cái đã đoán thì thông báo "Chữ cái này đã nhập rồi".

- Nếu người chơi nhập chữ cái không có trong từ thì thông báo "Chữ cái này không đúng".

- Nếu nhập xong toàn bộ các chữ cái của từ, chương trình thông báo chiến thắng và sau đó lặp lại quá trình chơi từ đầu.

Các biến nhớ chính của chương trình.

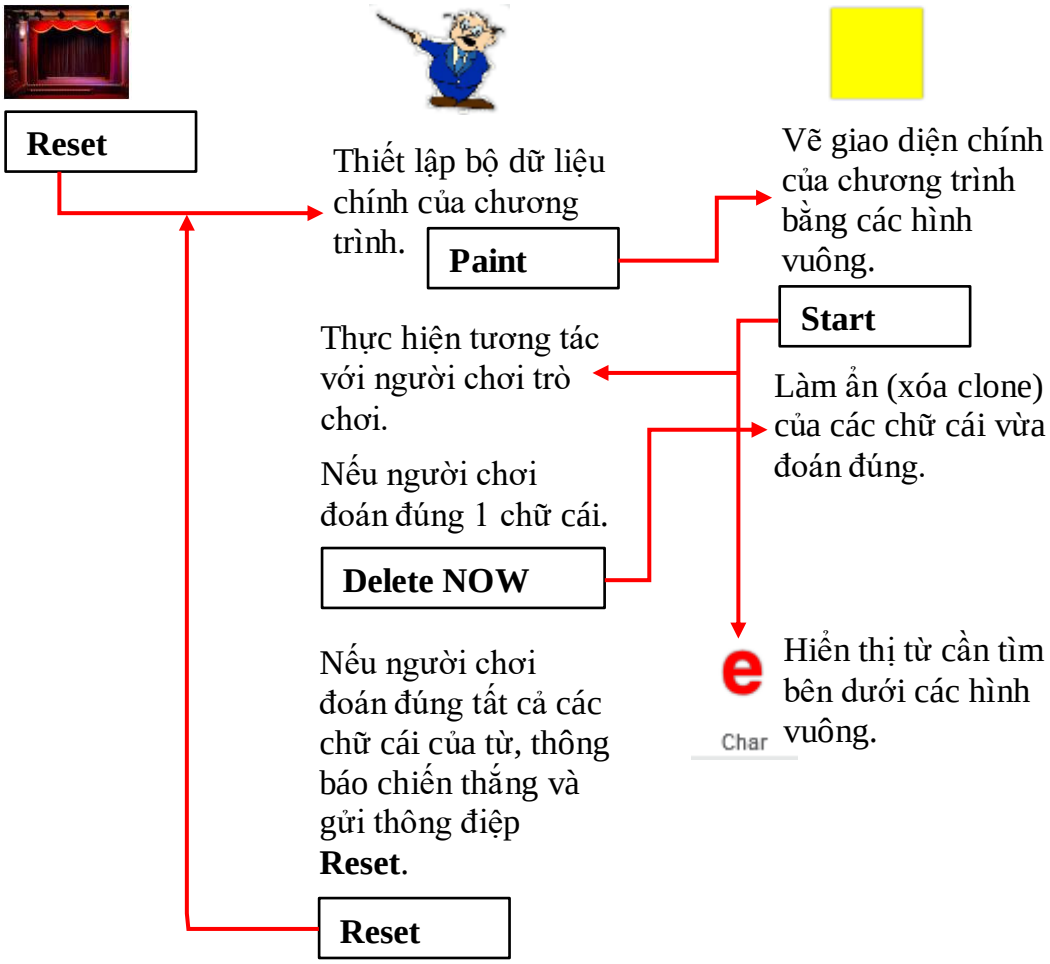
Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**Word**	Biến nhớ lưu từ khóa chính cần tìm của trò chơi. Từ này sẽ được lấy ngẫu nhiên từ nguồn dữ liệu **WList**.	(biến nhớ)
2	**WList**	Mảng dữ liệu chính là nguồn các từ gốc sẽ được lấy ra sử dụng cho chương trình.	(list)
3	**XList**	Dãy các tọa độ theo trục X để hiển thị nhân vật Square (hình vuông) che khuất từ cần tìm.	(list)

Vì kích thước màn hình bị hạn chế, chúng ta sẽ thiết kế ô vuông có kích thước 48x48, do đó chiều ngang của màn hình chỉ có thể hiển thị được 1 từ 10 ký tự. Tọa độ các tâm của lưới thể hiện từ khóa có tọa độ theo trục X như sau:

Dãy **XList** có dữ liệu sau:

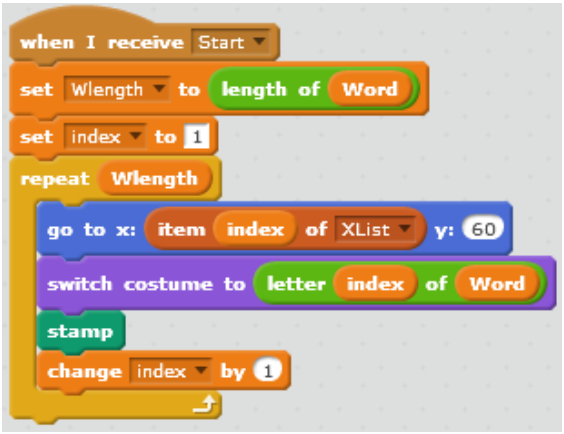
-216	**-168**	**-120**	**-72**	**-24**	**24**	**72**	**120**	**168**	**216**

Sơ đồ hoạt động của các nhân vật như sau:



Đoạn chương trình mô tả vẽ lưới ô vuông tương ứng với từ khóa **Word** bằng việc tạo clone cho nhân vật **Square**. Kỹ thuật dùng **clone** để vẽ lưới các hình vuông lần đầu tiên được sử dụng trong chương trình này và còn được sử dụng trong nhiều chương trình khác của Scratch.

Đoạn chương trình của nhân vật **Char** hiển thị dãy các chữ cái của từ **Word** theo lưới (nằm dưới các ô vuông) như sau:



Thuật toán xử lý chính của trò chơi khi tương tác với người chơi.

Chúng ta sẽ sử dụng bộ dữ liệu sau để thực hiện thuật toán này.

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**W-Array**	Mảng này sẽ lưu các chữ cái tương ứng của từ **Word** chính. Bảng này được tự động tạo ra ngay sau khi sinh từ khóa **Word**.	(list)
2	**Ch-Array**	Mảng này dùng để kiểm tra xem các chữ cái tương ứng của Word đã tìm được chưa trong quá trình chơi. Bảng này có số phần tử bằng **W-Array** và bằng độ dài từ **Word**. Mỗi phần từ của bảng này có ý nghĩa sau: = ? nếu ký tự tương ứng chưa được tìm ra. = 1 ký tự cụ thể (khác ?) nếu ký tự tương ứng đã được tìm thấy. Bảng này được tự động tạo ra ngay sau khi sinh từ khóa **Word**. Ban đầu bảng này bao gồm toàn các phần tử = ?	(list)
3	**D-Array**	Bảng này lưu các chỉ số của các ký tự khi người dùng đoán đúng 1 chữ cái, các vị trí này sau đó sẽ được mở ô hình vuông (thông qua thông điệp **DeleteNOW**. Ví dụ nếu người dùng nhập chữ cái a, trong từ Word có 2 chữ a ở các vị trí 2, 5 thì bảng D-Array sẽ có 2 phần tử {2, 5}.	(list)

Thuật toán sau mô tả các bước thực hiện khi người chơi nhập 1 chữ cái từ bàn phím để đoán tiếp từ hiện thời. Giả sử ký tự người dùng nhập là Ch.

- Kiểm tra Ch có trong bảng W-Array hay không? Nếu không có thì thông báo "Chữ đã nhập không đúng",

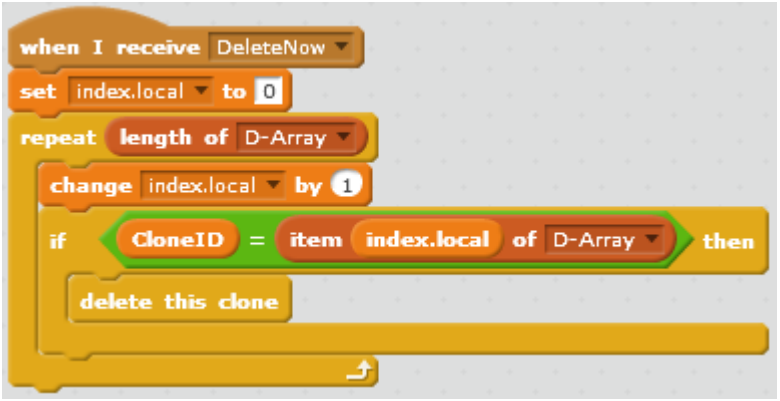
- Nếu Ch có trong W-Array mà tại vị trí đó phần tử tương ứng của Ch-Array không = "?" thì thông báo "Chữ này đã nhập rồi".

- Nếu Ch có trong W-Array mà tại vị trí đó phần tử tương ứng của Ch-Array bằng ? thì thực hiện tiếp như sau:

+ Kiểm tra, lặp theo bảng W-Array xem tại những vị trí nào có giá trị = Ch, tại các vị trí đó sẽ bổ sung chỉ số tìm thấy vào bảng D-Array.

+ Thông báo, ví dụ "Đã tìm thấy 2 chữ a", sau đó gửi thông điệp DeleteNOW để nhân vật hình vuông sẽ thực hiện việc xóa clone để mở ra các vị trí tương ứng với các chỉ số có trong bảng D-Array.

Đoạn chương trình của nhân vật Square khi xử lý thông điệp DeleteNOW như sau:



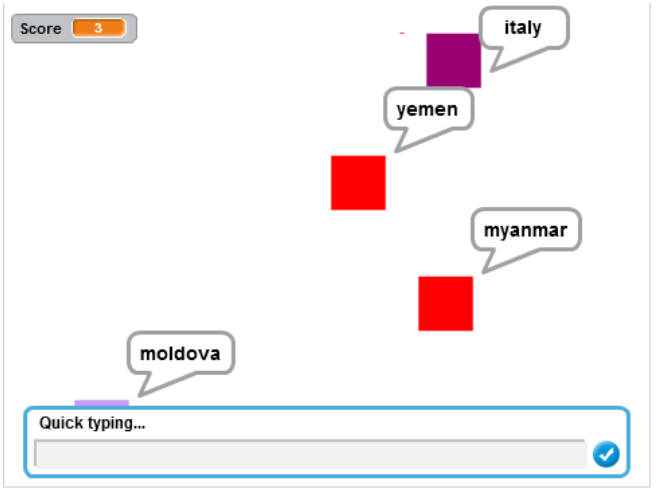
Thực hiện lệnh lặp theo mảng D-Array, kiểm tra nếu CloneID = chỉ số trong bảng này thì xóa clone tương ứng.

Em hãy hoàn thiện nốt các phần còn lại của chương trình này và thực hiện tiếp các bài tập bổ sung.

3. Trò chơi mưa từ

Trong hoạt động này chúng ta sẽ cùng thiết kế 1 trò chơi giáo dục đơn giản nhưng có nhiều ý nghĩa dành cho các bé nhỏ tuổi, trò chơi **Mưa từ (Word Rain)**.

Mua tu.sb2



Giao diện của chương trình và danh sách các nhân vật chính. Nhân vật chính là 1 hình vuông (tên **Square**) có các trang phục nhiều màu sắc khác nhau.

Mô tả hoạt động, yêu cầu của chương trình.

- Màn hình khởi động của chương trình.

- Chương trình bắt đầu chạy khi người dùng bấm phím Space hoặc nháy chuột bất kỳ trên màn hình.

- Từ phía trên của màn hình, các hình vuông với màu sắc khác nhau sẽ rơi xuống, mỗi hình kèm theo 1 từ (vì vậy chúng ta gọi là mưa từ). Các từ này có thể giống nhau và được lấy ngẫu nhiên từ 1 từ điển từ cho trước.

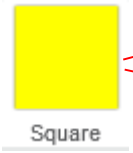
- Người chơi phải quan sát và ngăn không cho các hình này rơi xuống phía dưới. Nếu rơi xuống quá nhiều từ sẽ bị thua.

- Người dùng có thể thực hiện 1 trong 2 cách sau:

(a) Nháy chuột nhanh lên các hình vuông để chúng biến mất.

(b) Gõ nhanh từ đang rơi trên màn hình. Nếu gõ đúng 1 từ, từ và hình tương ứng sẽ biến mất, điểm số sẽ tăng cao hơn.

Để mô phỏng các từ được rơi xuống cùng hình ảnh nhân vật Square, chúng ta sử dụng kỹ thuật clone. Để cho từ có thể gắn kết liền với hình, chúng ta tạo 1 biến nhớ riêng của nhân vật này với tên Word để lưu trữ các từ này. Khi hình rơi xuống cùng với từ, chúng ta sẽ có cảm giác mưa từ.



CloneID - chỉ số riêng của hình.

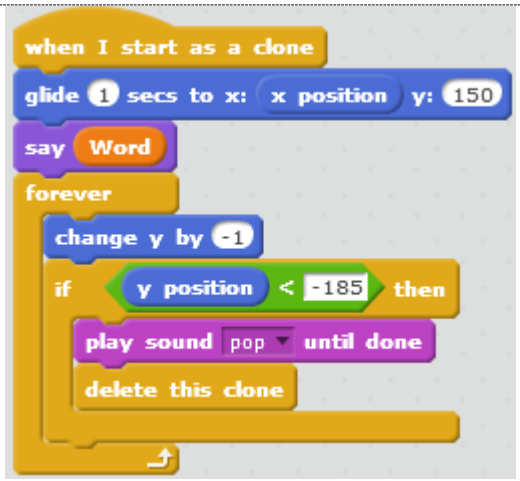
Word - từ luôn đi kèm hình.

Bộ dữ liệu chính của chương trình:

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**WList**	Mảng, bộ từ điển các từ gốc dùng trong chương trình. Các từ được sinh ngẫu nhiên từ danh sách này.	(list)
2	**Word**	Biến nhớ này sẽ lưu trữ từ được sinh ra (lấy từ WList) và gán cứng với clone hiện thời.	Biến nhớ riêng của nhân vật hình vuông (Square)

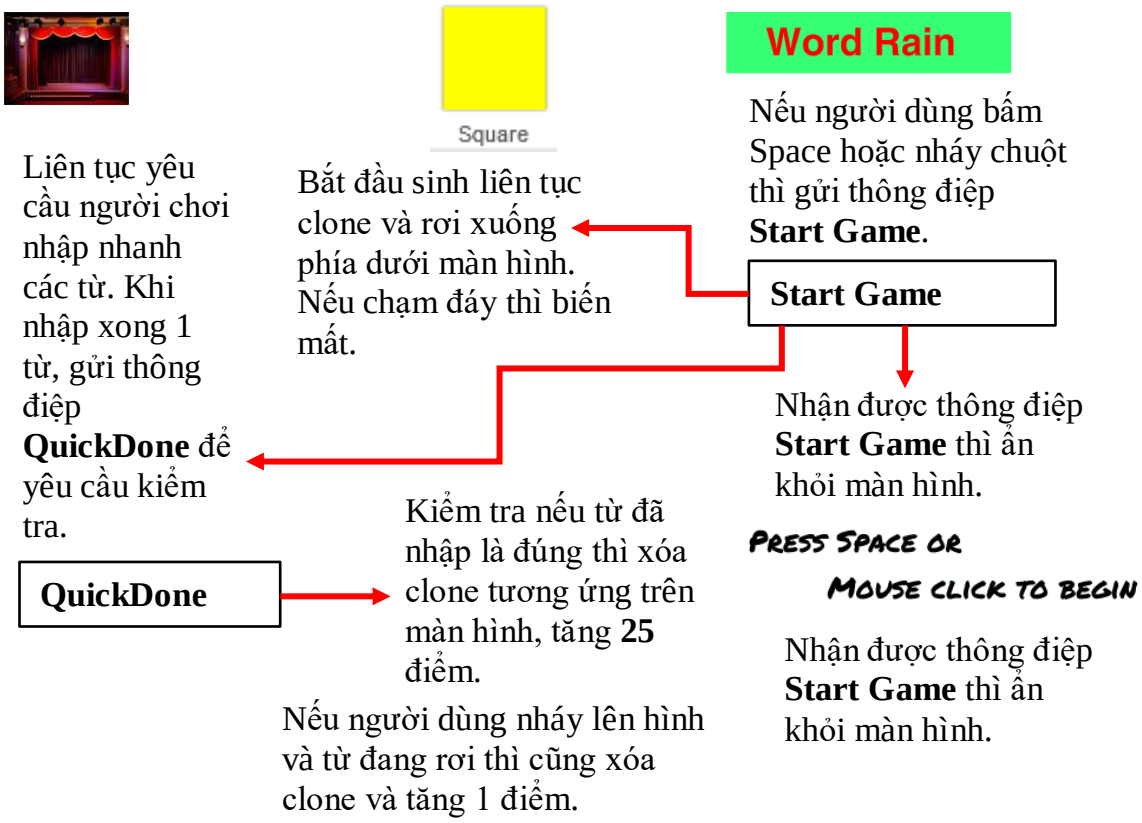
Các đoạn chương trình chính xử lý nhân vật Square khi tạo clone và rơi xuống như sau.

Sau mỗi 3 giây 1 clone lại được khởi tạo. Clone này trước khi xuất hiện sẽ được chuyển màu sắc và gán từ Word được sinh ngẫu nhiên từ từ điển WList.



Trong suốt quá trình rơi xuống, lệnh **say <Word>** luôn được thực hiện đảm bảo từ này luôn được gắn với hình trong suốt quá trình mưa từ.

Sơ đồ hoạt động của chương trình như sau:

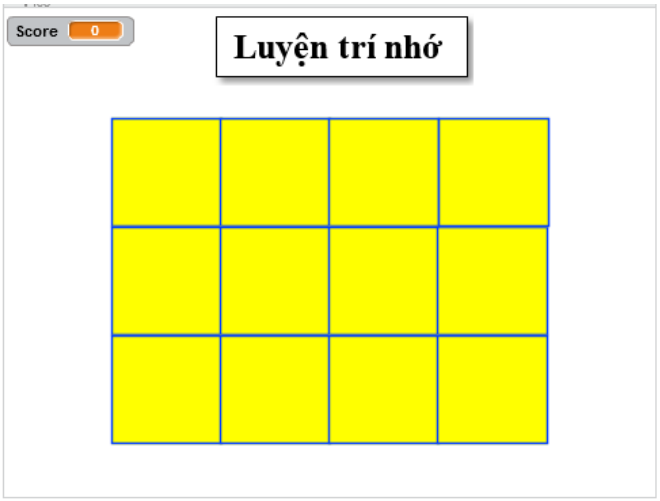


Em hãy hoàn thiện chương trình và thực hiện các bài tập bổ sung tiếp theo đối với chương trình này.

4. Trò chơi luyện trí nhớ

Chúng ta sẽ cùng nhau phát triển và thiết kế 1 trò chơi giáo dục rất quen thuộc đối với học sinh: trò chơi luyện trí nhớ.

Memory.sb2

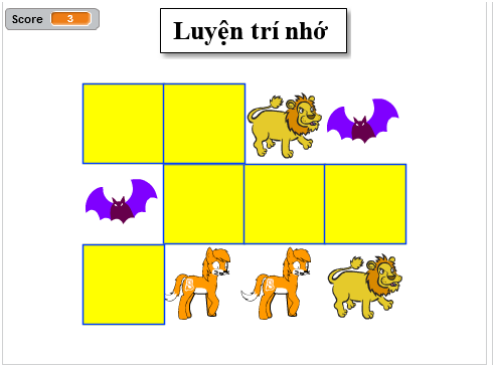


Giao diện trò chơi rất đơn giản: trên màn hình là 12 mảnh ghép, bên dưới mỗi mảnh ghép (hình vuông) là 1 nội dung nào đó (bằng chữ hoặc hình ảnh). Người chơi cần nháy chuột để lật tất cả các mảnh ghép trên màn hình.

Mô tả yêu cầu chương trình.

Bắt đầu chương trình.

Giao diện khi đang chơi.



- Giao diện của trò chơi bao gồm 12 mảnh ghép (mỗi mảnh là 1 hình vuông). Bên dưới các mảnh ghép là 1 nội dung (chữ hoặc hình ảnh), có 6 cặp thông tin giống nhau. Nhiệm vụ người chơi là lật được tất cả các mảnh ghép bằng cách nháy chuột lên các mảnh ghép.

- Qui tắc lật như sau: người chơi sẽ lật được 2 mảnh ghép liên tục nếu 2 mảnh đó chứa nội dung giống nhau.

- Ví dụ: lần đầu nháy và lật được 1 mảnh ghép, khi nháy lật lần thứ 2, nếu nội dung mảnh sau trùng với mảnh trước thì 2 mảnh đó sẽ được mở vĩnh viễn, nếu nội dung mảnh sau khác mảnh trước thì 2 mảnh đó sẽ đóng lại và người chơi phải thực hiện lại từ đầu.

- Qui tắc tính điểm như sau: mỗi khi lật được vĩnh viễn 2 mảnh có nội dung giống nhau thì tăng 1 điểm, khi hoàn thành 1 lần chơi mở tất cả các mảnh ghép sẽ được tăng 20 điểm.

Một số nhận xét và thiết kế sơ bộ

1) Để mô tả các mảnh ghép chúng ta dùng hình ảnh nhân vật hình vuông có kích thước 80 x 80. Tọa độ các tâm của hình vuông thể hiện trên lưới 3 x 4 trên màn hình được xác định bởi 2 mảng tọa độ **XList** (tọa độ theo X) và **YList** (tọa độ theo Y).

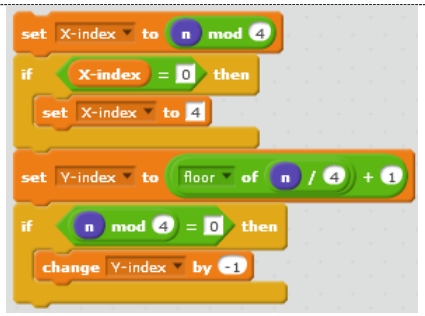
		1	2	3	4
		-120	**-40**	**40**	**120**
1	**60**	**1**	**2**	**3**	**4**
2	**-20**	**5**	**6**	**7**	**8**
3	**-100**	**9**	**10**	**11**	**12**

7 → (2, 3)

11 → (3, 3)

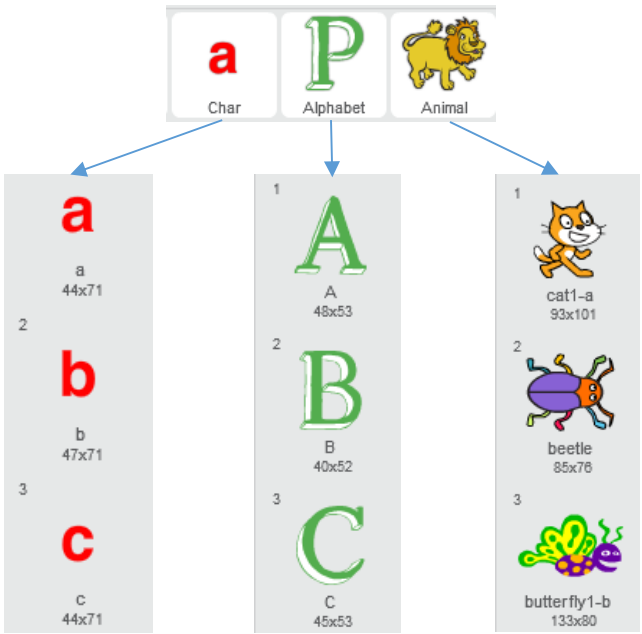
Mỗi mảnh ghép chính là 1 clone của nhân vật Square có giá trị CloneID được ghi trong các ô của bảng trên, đánh số từ 1 đến 12. Thuật toán sau tính được chỉ số theo hàng **X-index** và chỉ số theo cột **Y-index** theo giá trị n = số thứ tự mảnh ghép.

```
Gán X-index:= n mod 4;
Nếu X-index = 0 thì gán
X-index:=4;
Gán Y-index:= n div 4 + 1 (làm
tròn)
Nếu n mod 4 = 0 thì gán
Y-index := Y-index - 1;
```



2) Để mô tả nội dung hiện bên dưới các mảnh ghép, chúng ta dùng các nhân vật sau: chữ cái thường, chữ cái in hoa và hình ảnh con vật.

Mỗi nhân vật có nhiều trang phục như mô tả trong hình sau:



3) Thiết kế bộ dữ liệu cho chương trình. Các biến nhớ chính được ghi trong bảng sau:

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**XList**	Dãy các tọa độ X theo chiều ngang từ trái sang phải (4 giá trị).	(list)
2	**YList**	Dãy các tọa độ Y theo chiều dọc từ trên xuống dưới (3 giá trị).	(list)
3	**MType**	Kiểu của dữ liệu hiển thị trên ô lưới. = 1, hiển thị chữ cái thường (theo nhân vật Char). = 2, hiển thị chữ cái hoa (theo nhân vật Alphabet). = 3, hiển thị các con vật (theo nhân vật Animal).	(biến nhớ)
4	**GameType**	Biến nhớ chỉ trạng thái của trò chơi: = 0, đang chuẩn bị, không thể chơi. = 1, đang chơi.	(biến nhớ)
5	**NumList**	Dãy số tự nhiên 1, 2, 3, … có độ dài = số lượng trang phục của nhân vật tương ứng với MType. Bộ dữ liệu này dùng để sinh ngẫu nhiên bộ dữ liệu chính thể hiện nội dung trên lưới.	(biến nhớ)
6	**Memory.List**	Đây là bảng dữ liệu chính của chương trình, được sinh ra cho mỗi lần chơi. Bảng này 12 số, bao gồm 6 cặp giống nhau, mỗi số sẽ tương ứng với 1 giá trị của nội dung thể hiện trên lưới. Các phần tử của bảng này được lấy ngẫu nhiên từ bảng NumList.	(list)
7	**num_square_solve**	Biến nhớ được dùng trong quá trình tương tác với người chơi, chỉ ra hiện đã có bao nhiêu mảnh ghép đã được lật. Nếu **num_square_solve** = 12 thì việc chơi kết thúc.	(biến nhớ)
8	**Last.Square**	Biến nhớ này dùng để chỉ trạng thái người chơi nháy chuột lên 1 mảnh ghép. = 0 nếu trước đó chưa có mảnh ghép nào được lật. = k > 0 nếu trước đó mảnh ghép thứ k đã được lật.	(biến nhớ)

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
9	**Last.Square.Content**	Biến này có ý nghĩa khi Last.Square > 0, chỉ ra nội dung của ô mảnh ghép đã được lật trước đó. Như vậy khi người chơi lật 1 mảnh ghép, nếu nội dung của ô bên dưới = **Last.Square.Content** và **Last.Square** > 0 thì chứng tỏ đã nháy đúng 2 mảnh ghép giống nhau.	(biến nhớ)

Sơ đồ hoạt động của chương trình.

Reset

Thực hiện công việc vẽ 12 mảnh ghép cho ô chữ, sử dụng kỹ thuật clone.

Thực hiện các công việc thiết lập bộ dữ liệu gốc bao gồm NumList, Memory.List, sau đó gửi thông điệp **Make Content**.

Make Content

Start Game

Chờ người chơi nháy chuột lên các mảnh ghép để xử lý. Khi người dùng nháy chuột:

1. Nếu chưa có mảnh ghép nào mở (Last.Square=0) thì tiến mở mảnh ghép này và cập nhật các giá trị Last.Square, Last.Square.Content, tăng **num_square_solve** lên 1

2. Nếu đã có 1 mảnh ghép đang mở ((Last.Square>0), thì kiểm tra mảnh ghép vừa mở hiện thời với Last.Square.Content.

+ Nếu không trùng thì thực hiện thông báo **RestoreLast**.

RestoreLast

+ Nếu trùng nhau, thì mở luôn mảnh hiện thời, tăng **num_square_solve**, nếu giá trị này = 12 thì thông báo **End Game**.

End Game

Thực hiện công việc hiện lại mảnh ghép đã xóa mất của bước trước đây, khôi phục clone và giảm **num_square_solve** xuống 1

Thông báo chiến thắng, chờ 5 giây và gửi lại thông điệp **Reset**.

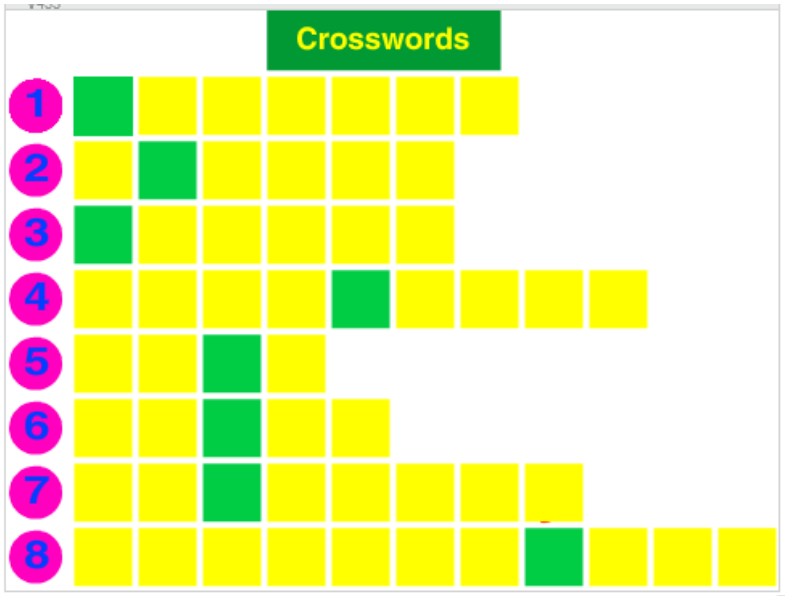
Reset

Thể hiện nội dung trên ô lưới bên dưới các mảnh ghép, sử dụng lệnh vẽ **stamp**. Phản hồi lại cho GV để chuẩn bị bắt đầu chơi.

5. Trò chơi đoán ô chữ

Chúng ta sẽ cùng thiết kế trò chơi Ô chữ (Crosswords), chương trình cuối cùng của cuốn sách và là phức tạp nhất. Trò chơi này rất nổi tiếng và đã có rất nhiều phần mềm mô phỏng trò chơi này.

O chu.sb2



Giao diện chương trình giải ô chữ. Vì chiều rộng màn hình bị hạn chế nên từ khóa chính theo hàng dọc sẽ được thể hiện bằng các ô màu xanh trên màn hình và không cần thẳng hàng.

Nhân vật của chương trình.

(a) Yêu cầu chương trình

- Chức năng chính của chương trình là cho phép người chơi giải các ô chữ. Mỗi ô chữ sẽ bao gồm: 1 từ khóa chính hàng dọc (main key) và nhiều từ khóa hàng ngang (key word). Quan hệ giữa các từ khóa hàng ngang với từ khóa chính hàng dọc: từ khóa hàng ngang thứ k sẽ phải chứa chữ cái thứ k của tư khóa chính hàng dọc.

- Vì cửa sổ sân khấu của Scratch có hạn nên chương trình chỉ thiết kế với mô hình các từ với hạn chế sau:

+ Các từ khóa hàng ngang có độ dài <= 11.

+ Từ khóa chính hàng dọc có độ dài <= 8.

+ Từ khóa hàng dọc không thể được xếp thẳng hàng, do đó các ô tương ứng với từ khóa hàng dọc sẽ được tô màu xanh (xem hình trên).

- Việc giải ô chữ như sau:

+ Giải các từ hàng ngang bằng cách nháy lên các nút tròn màu hồng tương ứng ở bên trái màn hình.

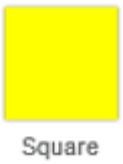
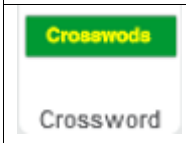
+ Giải từ khóa chính bằng cách nháy lên nút tên của chương trình.

+ Người chơi sẽ thắng nếu giải được tất cả các từ hàng ngang hoặc giải được từ khóa chính hàng dọc.

- Yêu cầu của chương trình là các bộ dữ liệu ô chữ phải được sinh tự động, ngẫu nhiên từ một CSDL được nạp sẵn trong chương trình.

(b) Thiết kế sơ bộ chương trình

Nhân vật và vai trò trong chương trình.

Biểu tượng	Tên nhân vật	Vai trò trong thiết kế
	Giáo viên	Giáo viên sẽ đóng vai trò người hướng dẫn chính của chương trình với người chơi. Đồng thời tất cả các hoạt động chính, thuật toán, các lệnh chính đều xuất phát từ nhân vật này.
	Hình vuông	Nhân vật này có 1 nhiệm vụ duy nhất là thực hiện việc che các chữ cái của các từ theo hàng ngang của ô chữ. Nhân vật này có 2 trang phục: màu vàng và màu xanh.
	Bảng chữ cái	Nhân vật **Char** dùng để thể hiện các từ khóa của ô chữ. Việc thể hiện sẽ sử dụng lệnh vẽ **stamp** của nhân vật này. Nhân vật sẽ có 26 trang phục tương ứng với 26 chữ cái tiếng Anh. Tên của trang phục trùng với tên chữ cái, ví dụ: a, b, c, d, e, f, …. Chú ý: chương trình chỉ dùng các hình ảnh chữ cái tiêng Anh thường, không có chữ hoa, do đó tất cả các các từ khóa của chương trình đều phải viết bằng chữ thường.
	Vòng tròn có số	Nhân vật này là hình tròn, có 8 trang phục tương ứng với các hình có số từ 1 đến 8. Nhiệm vụ của nhân vật này là hiện thành các nút ở cột bên trái dùng để cho người chơi đoán các từ khóa hàng ngang.
	Nút / tên của chương trình.	Nút này có 2 ý nghĩa: - Hiện phía trên của màn hình làm tiêu đề cho phần mềm. - Dùng là nút lệnh cho người chơi đoán từ khóa chính hàng học.

Sơ đồ hoạt động của chương trình.

Nhãn, đoạn chương trình	Hoạt động chính	Vai trò và nhiệm vụ của nhân vật	Các chú ý khác
Bắt đầu	Bắt đầu chương trình.	Các nhân vật sẽ thiết lập trạng thái ban đầu của mỗi nhân vật.	Nền sân khấu sẽ phát thông điệp **Reset**.

Nhãn, đoạn chương trình	Hoạt động chính	Vai trò và nhiệm vụ của nhân vật	Các chú ý khác
Reset	Bắt đầu quá trình chính của trò chơi. Tại bước này sẽ thực hiện các nhiệm vụ sau: - Thiết lập các giá trị mặc định, ban đầu cho chương trình. - Sinh ngẫu nhiên 1 bộ dữ liệu cho trò chơi ô chữ, bao gồm: + 1 Từ khóa chính (theo hàng dọc). MainKey. + Dãy các từ khóa theo hàng ngang. KeyWord. Mỗi từ khóa hàng ngang phải chứa ký tự tương ứng theo thứ tự của từ khóa chính. Ví dụ nếu Main Key = "computer" thì từ khóa thứ nhất phải có chữ c, từ thứ 2 phải chứa chữ o, ….	- Nhân vật GV sẽ thực hiện tất cả các hoạt động của bước này.	Khi GV thực hiện xong tất cả các công việc sinh dữ liệu, sẽ phát thông điệp **Paint** để yêu cầu vẽ giao diện cho trò chơi.
Paint	Thực hiện việc vẽ các giao diện của chương trình.	- Nhân vật **Char** thực hiện vẽ giao diện các chữ cái thể hiện các từ hàng ngang. - Nhân vật **Square** thực hiện vẽ giao diện là các hình vuông đè lên các chữ cái để che khuất. Các hình vuông tương ứng với từ khóa chính sẽ có màu xanh. - Nhân vật **Circle** sẽ vẽ giao diện là dãy 8 hình tại cột bên trái.	Khi vẽ xong thì phát thông điệp **Start** để bắt đầu quá trình tương tác của người dùng với phần mềm.
Start	Bắt đầu chơi đoán ô chữ. - Muốn đoán 1 từ khóa hàng ngay, nháy nút hình tròn tương ứng ở đầu mỗi từ khóa. - Muốn đoán ngay từ khóa chính nháy lên nút tên chương trình (có chữ Crosswords). - Nếu đoán đúng 1 từ khóa hàng ngang thì các hình vuông che khuất từ trên hàng ngay này sẽ mất đi, để hiện ra từ khóa này.	Nhân vật GV sẽ thực hiện các hoạt động chính của bước này, các nhân vật khác phối hợp thực hiện. - Khi người dùng nháy lên nút hình tròn tương ứng với từ chưa tìm ra, nhân vật Circle sẽ gửi thông điệp **Tim_hang_ngang** cho GV. - Khi người dùng nháy lên nút Crosswords, nhân vật	Đây là bước phức tạp nhất của chương trình.

Nhãn, đoạn chương trình	Hoạt động chính	Vai trò và nhiệm vụ của nhân vật	Các chú ý khác
	- Trò chơi kết thúc khi người chơi hoặc đã đoán đúng được toàn bộ các từ hàng ngang hoặc đoán đúng được từ hàng dọc. Khi kết thúc sẽ dừng chương trình hoặc quay lại bước **Reset**.	này sẽ gửi thông điệp **Tim_hang_doc** cho GV. - Khi người dùng đoán đúng 1 từ hàng ngang, GV sẽ gửi thông điệp **OpenSquare** cho nhân vật **Square** để mở hình cho từ hàng ngang này.	

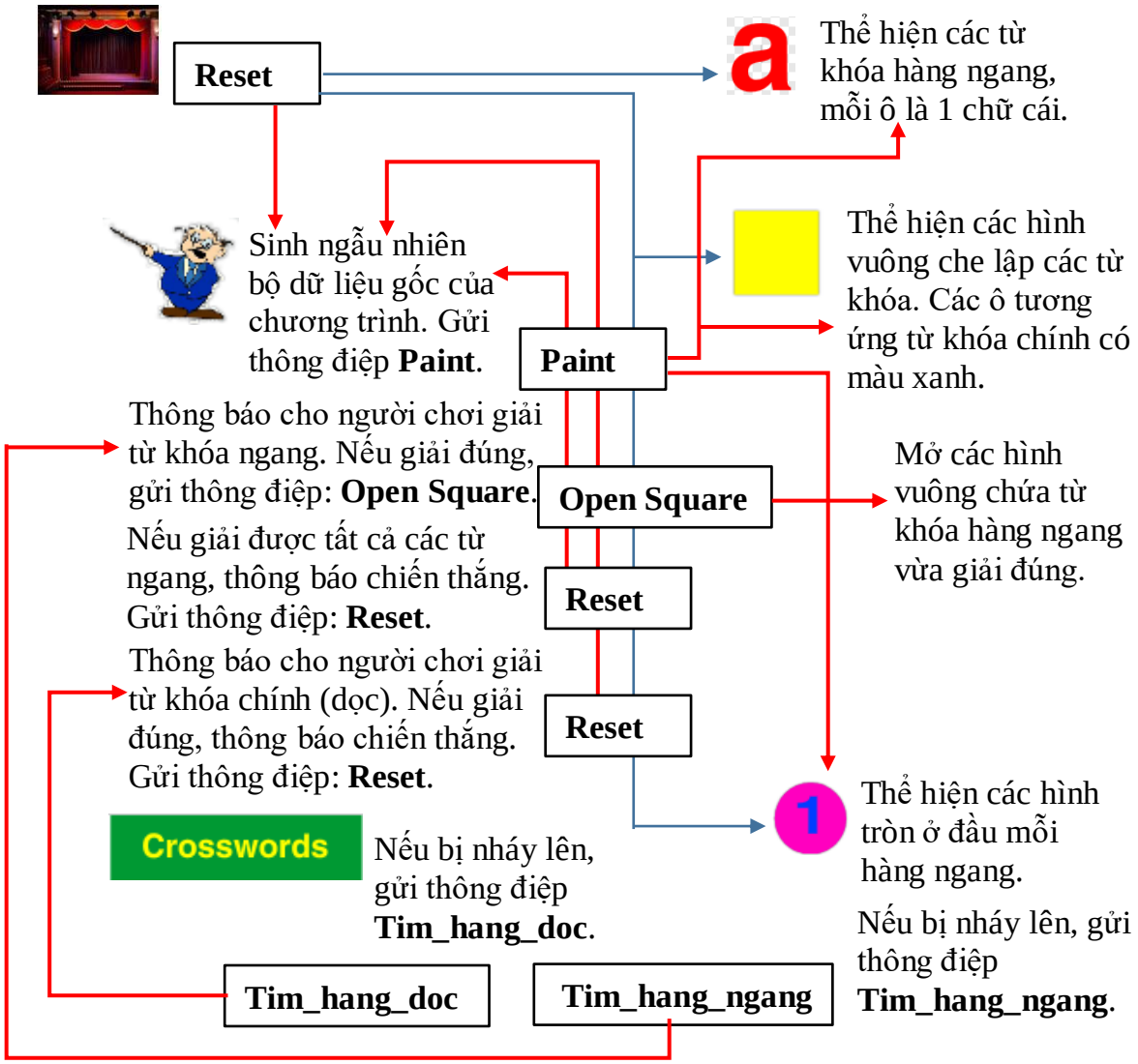
(c) Thiết kế sơ bộ dữ liệu cho chương trình.

Các biến nhớ chính và dữ liệu của chương trình.

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
1	**MKey**	Từ khóa chính của ô chữ hiện thời, theo chiều dọc thẳng đứng. Cách tính: lấy ngẫu nhiên 1 từ từ dãy **MKeyList**.	(biến nhớ)
2	**MKey.Description**	Mô tả, gợi ý của từ khóa chính.	(biến nhớ)
3	**KeyWList**	Các từ khóa theo chiều ngang của ô chữ hiện thời. Cách tính: lấy từ dãy WList và thỏa mãn chứa các chữ cái tương ứng của **MKey**.	(list)
4	**KeyWList.Description**	Các mô tả của các từ khóa theo chiều ngang.	(list)
5	**KeyWList.index**	Dãy các chỉ số, vị trí của từ trong dãy **KeyWList** chứa ký tự khóa. Trên màn hình, các ô chứa ký tự khóa chính sẽ đổi màu xanh.	(list)
6	**WList**	Dữ liệu các từ có thể làm từ khóa theo chiều ngang. Các từ này phải có độ dài <=11.	(list)
7	**WList.Description**	Mô tả của các từ trong dãy **WList**.	(list)
8	**MKeyList**	Dữ liệu gốc các từ có thể làm khóa chính. Các từ này phải có độ dài <=8.	(list)
9	**MKeyList.Desctiption**	Mô tả của các từ trong dãy **MKeyList**.	(list)

Stt	Tên biến nhớ	Ý nghĩa	Chú ý
10	**XList**	Dãy tọa độ các ô lưới theo trục X, tính từ trái sang phải.	(list)
11	**YList**	Dãy tọa độ các ô lưới theo trục Y, tính từ trên xuống dưới.	(list)
12	**main_key_solve**	Chỉ ra Main Key (từ khóa chính) đã được tìm ra chưa. = 0, chưa tìm được. = 1, đã tìm được.	(biến nhớ)
13	**num_key_solve**	Số lượng các từ khóa theo chiều ngang đã được tìm thấy.	(biến nhớ)
14	**KeyWList.solve**	Dãy các biến nhớ chỉ ra từ khóa hàng ngang nào đã được dự đoán, từ nào chưa.	(list)

(d) Sơ đồ hoạt động chi tiết

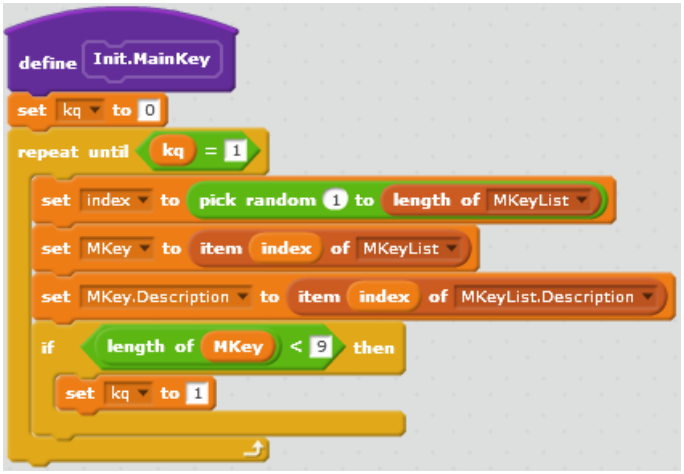


(e) Thiết kế chi tiết

1) Sinh ngẫu nhiên bộ dữ liệu ô chữ

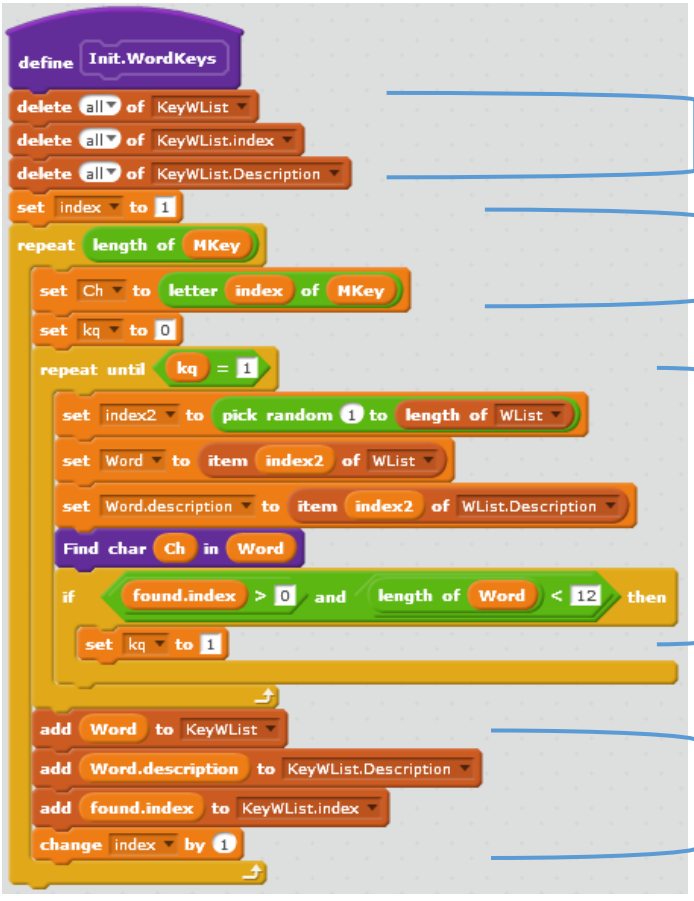
Khi nhận được thông điệp Reset, nhân vật GV sẽ thực hiện các lệnh sinh ngẫu nhiên bộ dữ liệu ô chữ. Các lệnh này sẽ được chia thành 2 thủ tục: **Init.MainKey** sinh từ khóa chính và **Init.WordKeys** sinh dãy các từ khóa hàng ngang.

Thủ tục **Init.MainKey**:



Từ khóa chính Mkey được lấy ngẫu nhiên từ mảng **MKeyList**. Thông tin giải nghĩa **MKey.Description** được lấy từ bảng MKeyList.Description tương ứng. Chú ý chỉ lấy các từ khóa chính có độ dài <=8.

Thủ tục **Init.WordKeys**. Thủ tục này sẽ xuất phát từ từ khóa chính **MKey** đã có, sẽ tìm lần lượt các từ khóa hàng ngang có chứa lần lượt các chữ cái trong **MKey**. Trong thủ tục này sẽ gọi 1 thủ tục khác **Find char <ch> in <Word>**, kết quả là vị trí của chữ cái <Ch> trong từ <Word>, giá trị này lưu trong biến nhớ tổng thể **found.index**.



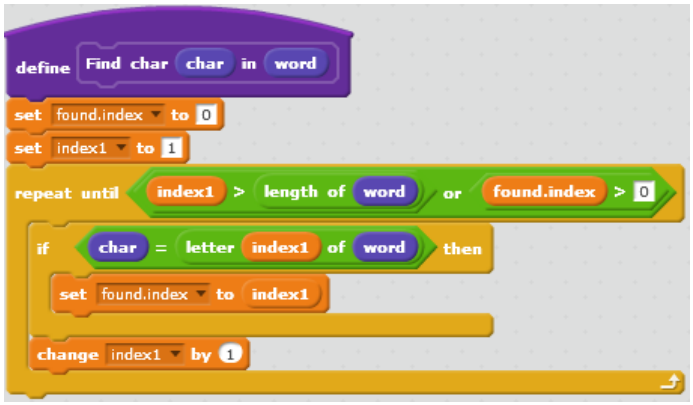
Xóa để làm lại từ đầu các dãy dữ liệu cần tìm.

Bắt đầu tìm từ vị trí index, lấy ra chữ cái **Ch** của vị trí index từ **MKey**.

Cách tìm: lấy ngẫu nhiên các từ từ kho dữ liệu **WList**, cho đến khi chứa chữ cái **Ch** và có độ dài <=11 thì dừng lại.

Tìm xong, gán các giá trị đã tìm được vào các dãy cần tìm.

Còn đây là thủ tục **Find char <ch> in <Word>**. Kết quả của thủ tục này sẽ lưu trong biến nhớ tổng thể **found.index**.



2) Thiết kế tọa độ màn hình ô chữ

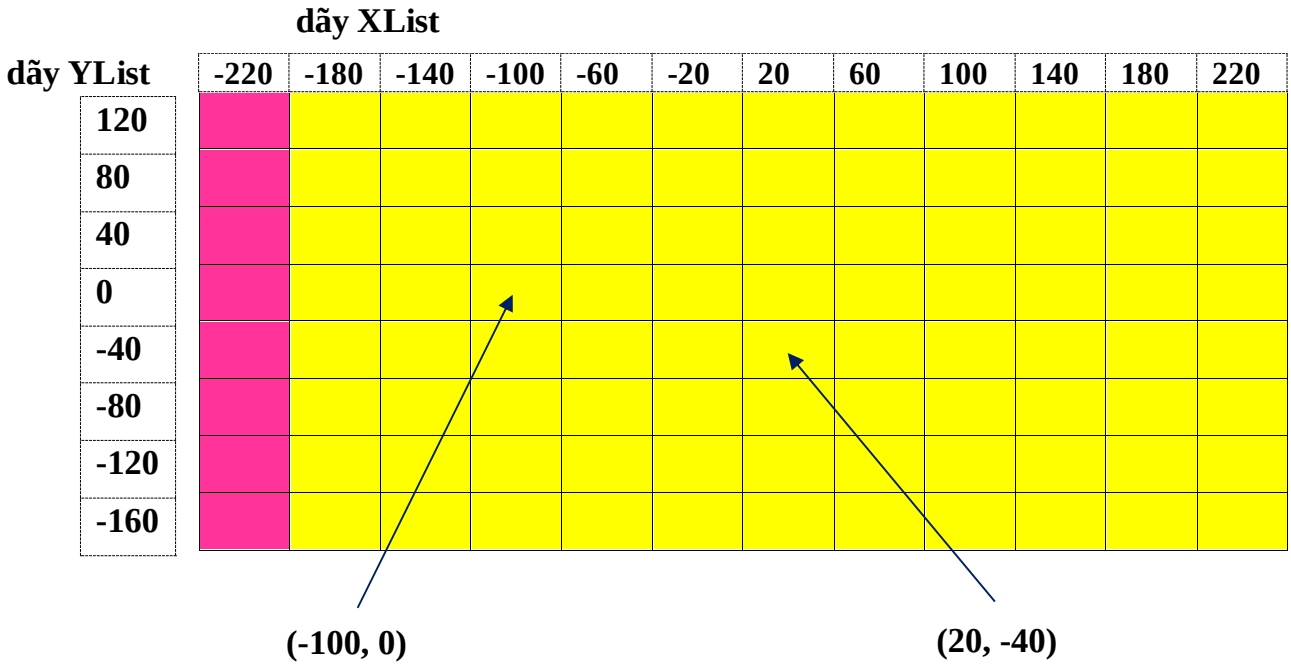
Bước tiếp theo là thiết kế giao diện để thể hiện ô chữ trên màn hình. Do hạn chế của kịch thước sân khấu Scratch nên chúng ta sẽ thiết kế mỗi ô vuông của các con chữ là 1 hình vuông **40 x 40**. Toàn bộ màn hình sẽ được chia thành 1 lưới ô vuông kích thước 12 x 8.

Do cột bên trái ngoài cùng phải dành cho dãy nút tròn nên vùng lưới chứa ô chữ sẽ chỉ là 11 x 8.

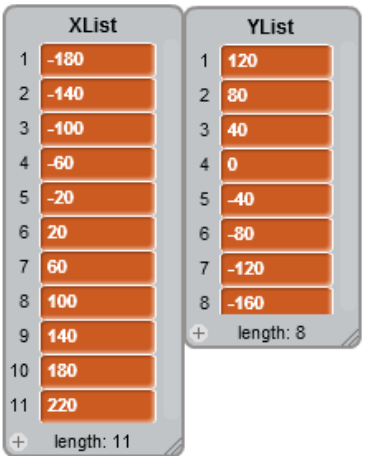
Tọa độ các tâm hình vuông theo chiều ngang (trục X), tính từ trái sang phải, được lưu trong dãy **XList**.

Tọa độ các tâm hình vuông theo chiều dọc (trục Y), từ trên xuống dưới, được lưu trong dãy **YList**.

Giá trị các bảng XList, YList thể hiện trong hình sau, đồng thời thể hiện thiết kế của lưới ô chữ của chương trình.



Hình ảnh 2 mảng số **XList** và **YList** trong chương trình.



Chú ý: dãy XList được thiết kế chỉ có 11 phần tử ứng với các ô vuông mù vàng của lưới ô chữ.

3) Thể hiện (paint) giao diện ô chữ: các hàng chữ hàng ngang

Nhân vật **Char** sẽ có nhiệm vụ thể hiện các từ hàng ngang của ô chữ.

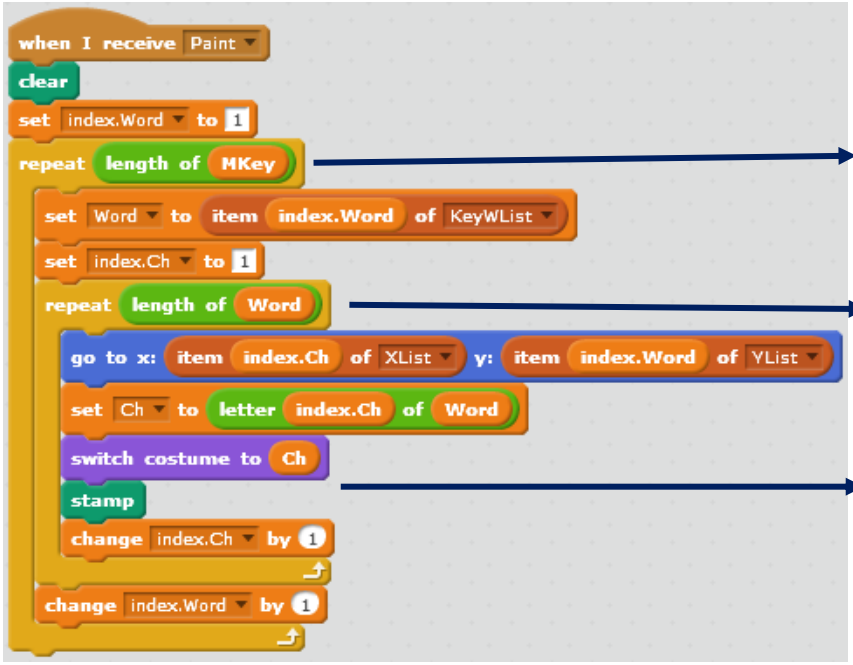
Chúng ta sử dụng kỹ thuật thể hiện chữ đã được đưa trong bài học 21. Nhân vật Char sẽ có các trang phục trùng với bảng chữ cái của ngôn ngữ hiện thời và có tên trang phục chính là chữ cái đó.

Trong chương trình này chúng ta sẽ thực hiện trên ngôn ngữ tiếng Anh với 26 chữ cái viết thường.

Chú ý: muốn mở rộng ra chữ hoa hoặc bổ sung thêm các ký tự khác hay ngôn ngữ khác thì chỉ cần mở rộng số lượng trang phục của nhân vật. Thuật toán sẽ không thay đổi.

Chú ý: Cần xác định tâm của trang phục nằm ở chính giữa hình ảnh chữ cái.

Đoạn chương trình thể hiện các từ hàng ngang trên lưới ô chữ của nhân vật Char như sau:



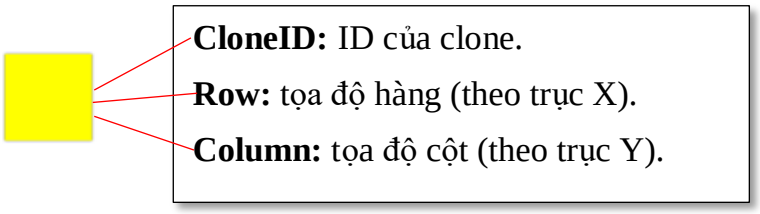
Vòng lặp ngoài, theo độ dài **MKey**.

Vòng lặp trong, theo độ dài của từng từ hàng ngang **KeyWList**.

Chuyển trang phục và vẽ bằng lệnh **stamp**.

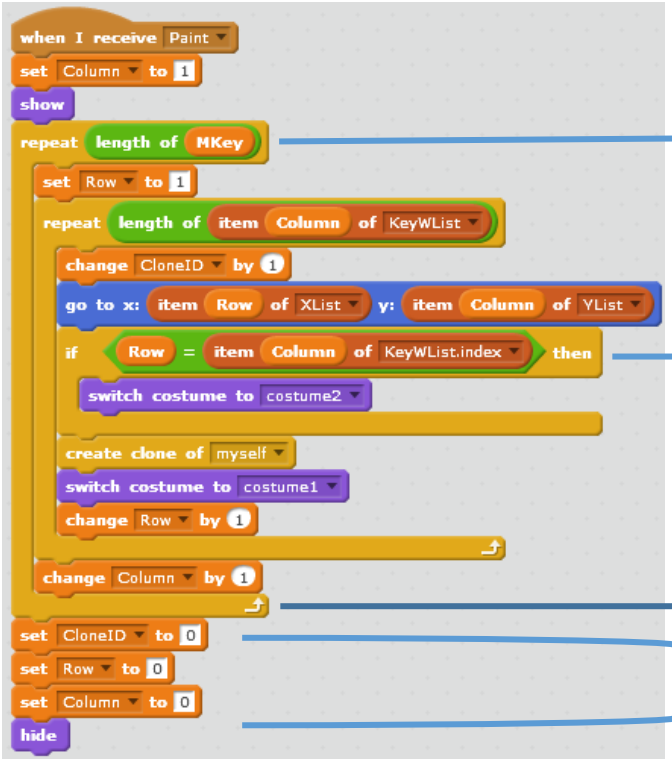
4) Thể hiện (paint) giao diện ô chữ: các hình vuông

Để thực hiện công việc này, chúng ta sẽ sử dụng kỹ thuật tạo clone quen thuộc của Scratch cho nhân vật Square (hình vuông). Để lưu trữ và tìm kiếm nhanh được vị trí của các hình vuông clone trên màn hình chúng ta bổ sung thêm 2 biến riêng cho hình vuông là **Row** - tọa độ theo X và **Column** - tọa độ theo Y.



Các biến nhớ riêng (local) của nhân vật Square.

Đoạn chương trình sinh clone của nhân vật Square phủ lưới ô vuông ô chữ như sau:



Đoạn chương trình sẽ ba gồm 2 vòng lặp, vòng ngoài có độ dài của từ khóa chính length(Mkey) bước, vòng lặp trong theo độ dài của từng từ hàng ngang.

Nếu ký tự thuộc từ khóa chính thì chuyển màu xanh.

Sau khi tạo xong clone thì trả lại các giá trị ban đầu của các biến riêng của nhân vật.

5) Thể hiện (paint) giao diện ô chữ: cột tròn bên trái đầu các hàng ngang

Thể hiện cột tròn bên trái các hàng ô chữ là nhân vật Circle. Cách thực hiện cũng dùng kỹ thuật clone. Nhân vật này sẽ có 8 trang phục tương ứng với các hình tròn có các chữ số từ 1 đến 8. Đoạn chương trình vẽ cột đầu tiên như sau.

Sử dụng biến nhớ riêng CloneID để đánh số các clone được khởi tạo.

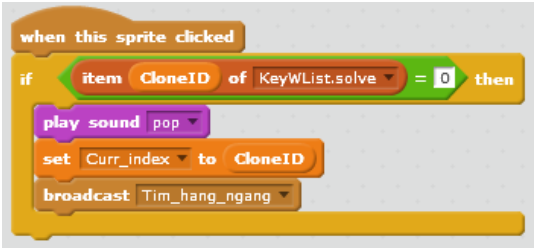
Với mỗi clone với CloneID, lệnh:

sẽ tạo 1 clone tại đúng vị trí của mình và có hình ảnh mang đúng số.

6) Tương tác giải từ hàng ngang

Khi người chơi nháy chuột lên 1 nút đỏ, biến nhớ tổng thể **Curr_index** sẽ lưu lại chỉ số của từ hàng ngang cần đoán.

Đoạn chương trình gửi thông điệp **Tim_hang_ngang** của **Circle** như sau.

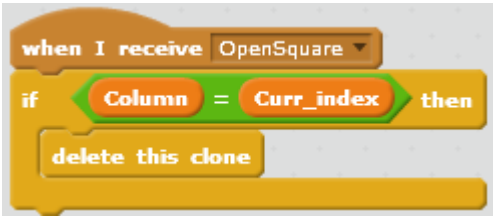


Kiểm tra xem từ hàng ngay này đã được tìm thấy trước đó hay chưa, thông tin này lưu trong dãy KeyWList.solve.

Biến nhớ **Curr_index** sẽ được gán trước khi gửi thông điệp.

GV sẽ thực hiện chương trình tạo tương tác để người chơi nhập từ hàng ngang tương ứng. Nếu người chơi nhập đúng thì gửi thông điệp OpenSquare cho nhân vật Square để làm ẩn (xóa) các clone có trên hàng tương ứng.

Đoạn chương trình xử lý của nhân vật hình vuông đơn giản, kiểm tra xem nếu giá trị Column = Curr_index thì xóa clone tương ứng:



7) Tương tác giải từ hàng dọc

Khi người dùng nháy lên nút tên của chương trình, nhân vật này sẽ kiểm tra xem từ khóa chính đã giải được chưa, nếu chưa thì gửi thông điệp Tim_hang_doc cho GV thực hiện các bước để người chơi thực hiện việc đoán từ khoa chính.

Đoạn chương trình này đơn giản như sau, chú ý rằng biến nhớ **main_key_solve** dùng để kiểm soát xem từ khóa chính đã đoán được hay chưa.

Em hãy hoàn thiện toàn bộ chương trình và thực hiện các bài tập mở rộng trong phần sau của bài học.

Câu hỏi, bài tập

1. Hoàn thiện trò chơi ghép từ theo thiết kế của hoạt động 1 của bài học.

2. Bổ sung trò chơi ghép từ trên 1 chức năng sau:

Bổ sung 1 nhân vật là con Rùa có chức năng hỗ trợ người chơi như sau:

Khi nháy lên con Rùa, Rùa sẽ hiển thị các gợi ý cho từ cần tìm, ví dụ:

- Chữ cái đầu tiên là chữ a.

- Chữ cái thứ 2 là chữ v, …. cứ như vậy.

Con Rùa này chỉ xuất hiện khi đang trong thời gian đoán chữ. Nếu người chơi đã đoán đúng từ thì con Rùa tạm ẩn đi cho đến lượt chơi sau.

3. Hoàn thiện trò chơi Tìm từ (hang man) đã thiết kế của bài học.

4. Bổ sung chức năng tính điểm số thông qua biến nhớ Score trong 2 chương trình Ghép chữ và Tìm từ.

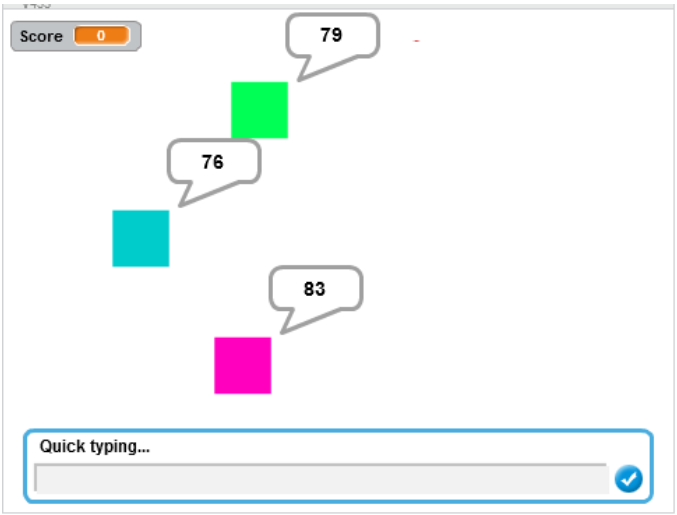
5. Trong thiết kế chương trình Hangman ở trên, từ và các ô chữ cần tìm được thể hiện từ bên trái, hay còn gọi là căn trái.

Em có thể mở rộng, thay đổi để từ này căn vào giữa được không. Hãy thực hiện các thay đổi cần thiết để làm được điều này.

6. Hoàn thiện chương trình Mưa từ theo thiết kế chi tiết của hoạt động 3.

7. Trong chương trình Mưa từ, em hãy thay đổi bộ dữ liệu WList thành bảng các chữ cái tiếng Anh (hoặc tiếng Việt). Khi đó chương trình trở thành phần mềm **Mưa chữ cái**, hỗ trợ các em nhỏ vừa chơi vừa học bảng chữ cái.

8. Viết 1 chương trình **"Mưa số"** có thiết kế tương tự như Mưa từ. Ví dụ giao diện của chương trình có dạng như sau.



9. Mở rộng trò chơi **Mưa chữ cái** trong bài tập 7 ở trên như sau:

Nếu học sinh nhập chính xác 1 chữ cái thì chương trình sẽ phát âm chữ cái đó trước khi hình tương ứng bị mất đi trên màn hình.

10. Mở rộng trò chơi **Mưa số** trong bài tập 8 ở trên như sau:

Yêu cầu người chơi nhập không phải tất cả các số đang rơi xuống, mà hạn chế theo các yêu cầu riêng. Các yêu cầu riêng này sẽ tự động thay đổi theo thời gian. Ví dụ các yêu cầu:

- Nhập các số chẵn có trên màn hình.

- Nhập các số lẻ có trên màn hình.

- Chỉ nhập các số < 10.

- ….

Em hãy viết chương trình mở rộng này.

11. Với các nhân vật Char được dùng nhiều trong các ví dụ của bài học này thì thủ tục sau có chức năng gì? Hãy giải thích ý nghĩa của thủ tục này.

Chú ý đến nhóm lệnh này.

12. Bài tập 11 trên đây chỉ ra 1 công cụ khác nữa để thể hiện chữ và số bằng hình ảnh trên màn hình: công cụ **clone**. Em hãy thiết kế 1 bộ công cụ hoàn chỉnh như vậy cho việc thể hiện chữ cái, từ, số trên màn hình.

13. Hoàn thiện chương trình **Luyện trí nhớ** trong mục 4 của bài học.

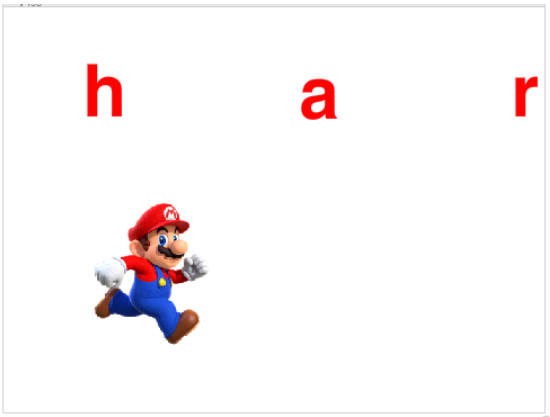
14. Hoàn thiện chương trình **Ô chữ** trong mục 5 của bài học.

15. Thay đổi 2 chương trình Luyện trí nhớ và Ô chữ, thay công cụ thể hiện nội dung (bên dưới các mảnh ghép), không dùng **Stamp**, mà thay vào đó dùng công cụ **Clone**.

Mở rộng

1. Thiết kế trò chơi luyện gõ bàn phím **Mario Typing** như sau.

Nhân vật chính là hình ảnh Mario quen thuộc. Từ bên phải phía trên các chữ cái sẽ lần lượt xuất hiện và chạy chậm sang trái. Người điều khiển Mario có thể cho chuyển động theo chiều ngang. Gõ 1 chữ cái sẽ làm cho Mario nhảy lên. Nếu gõ 1 chữ cái, Mario nhảy lên đúng chữ đó thì chữ đó biến mất và được thưởng điểm. Nếu nhảy lên không vào đúng chữ đã gõ thì bị trừ điểm.



2. Thiết kế trò chơi **Mưa chữ & số** như sau.

Từ phía trên của màn hình sẽ ngẫu nhiên rơi xuống các chữ cái hoặc số (thể hiện bằng hình ảnh to). Người chơi cần gõ nhanh từ bàn phím các chữ cái hoặc số này để chúng biến mất khỏi màn hình.

Nếu để có quá nhiều chữ cái hoặc số rơi xuống phía dưới màn hình thì bị thua.

Em hãy phát triển thiết kế này của chương trình để trở thành 1 chương trình trò chơi học tập hoàn chỉnh.

PHỤ LỤC 1: Vẽ đồ họa trong Scratch

Trong Scratch chúng ta phải thường xuyên sử dụng màn hình chỉnh sửa đồ họa để làm việc với hình ảnh nhân vật và sân khấu. Chức năng này được dùng trong các trường hợp sau:

- Chỉnh, sửa hình ảnh của các nhân vật hoặc hình nền đã có sẵn.

- Tạo mới nhân vật hoặc sân khấu sử dụng chức năng vẽ đồ họa để vẽ mới các hình nảy.

- Tạo mới trang phục của nhân vật.

Cửa sổ thiết kế đồ họa Scratch

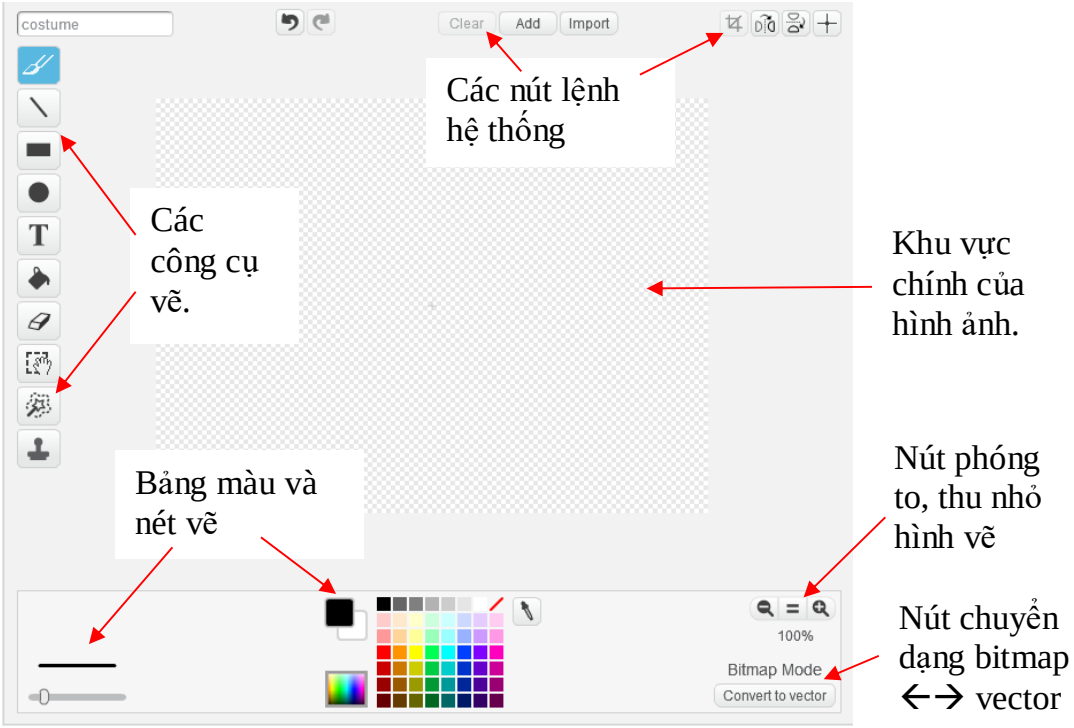
Chức năng thiết kế độ họa trong Scract có thể tạo các hình ảnh theo 2 dạng sau:

(1) Hình ảnh bitmap.

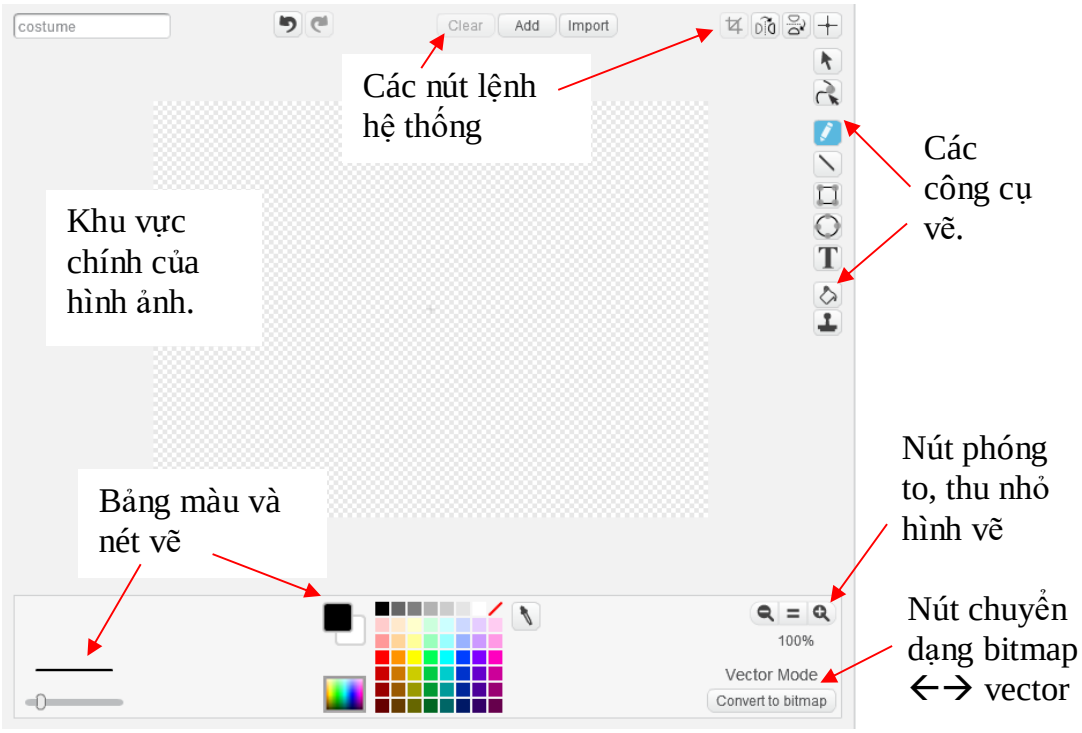
(2) Hình ảnh vector.

Em có thể chuyển đổi giữa 2 dạng hình ảnh bất cứ lúc nào.

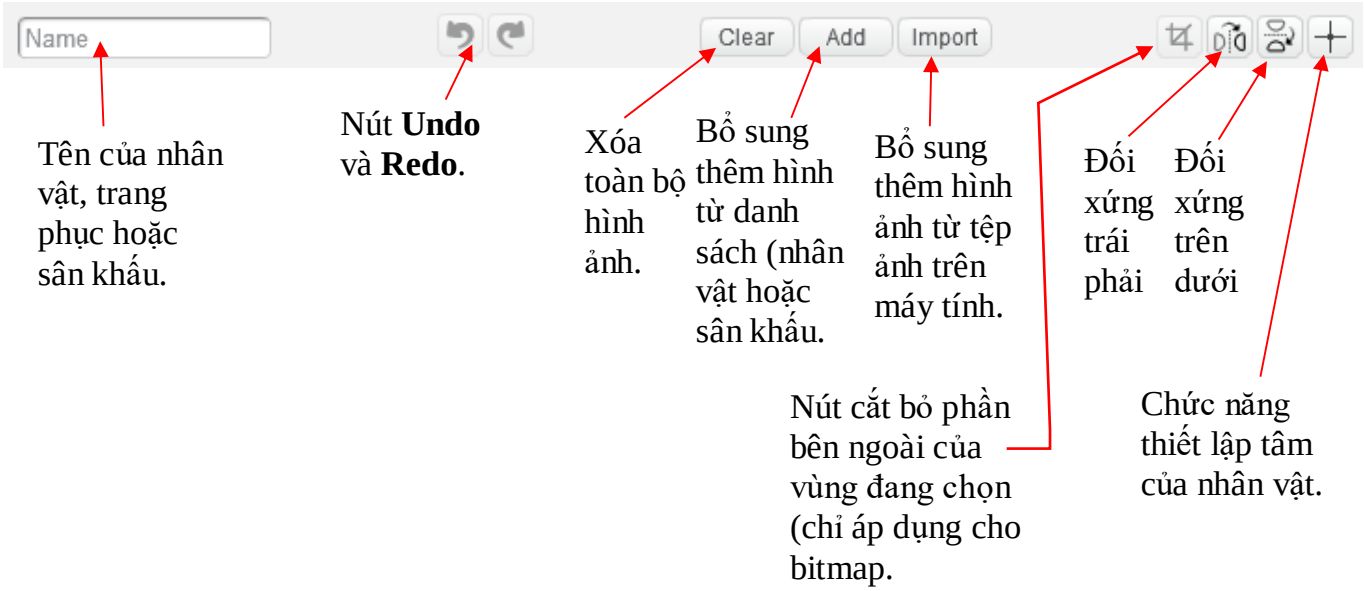
Hình dưới đây là cửa sổ soạn thảo hình ảnh dạng Bitmap, tức là hình ảnh được tạo ra từ các điểm có tô màu trên mặt phẳng.



Còn đây là giao diện của cửa sổ soạn thảo hình ảnh dạng vector, tức là hình sẽ bao gồm các hình hình học như đường thẳng, đường cong, đường tròn, hình hộp, ….



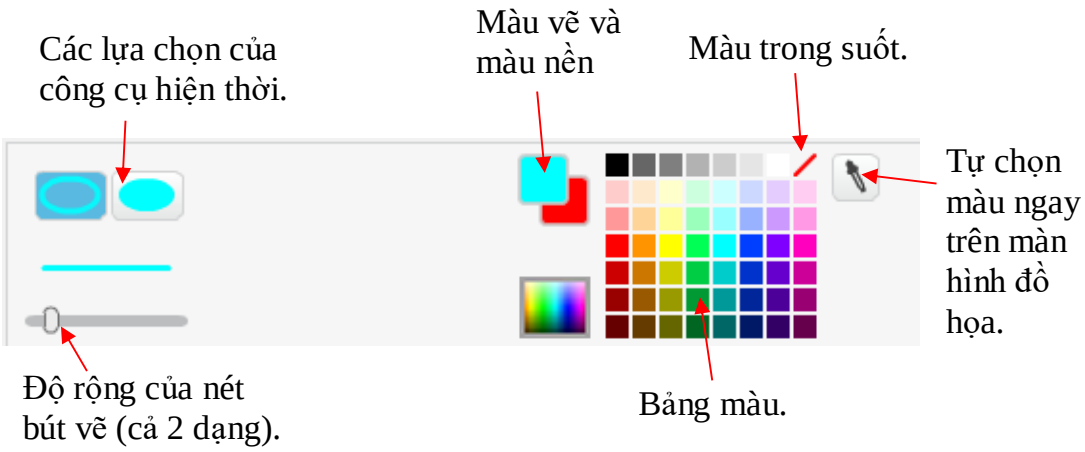
Các công cụ hệ thống chung



Bảng màu hệ thống

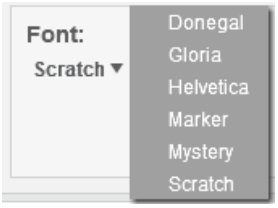
Vị trí bảng màu hệ thống phía dưới màn hình dùng để thiết lập các thông số:

- Màu sắc hệ thống bao gồm màu vẽ và màu nền.
- Thiết lập độ rộng nét vẽ.
- Thiết lập các lựa chọn đặc thù cho từng công cụ.



Các công cụ vẽ của hình dạng Bitmap

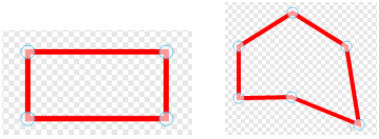
Các công cụ khi vẽ hình dạng Bitmap. Hình ảnh bitmap sẽ coi như nền của hình.

Biểu tượng	Công cụ	Chức năng, thao tác sử dụng
	Bút vẽ	Vẽ tự do theo màu vẽ và độ lớn bút vẽ hiện thời. Thao tác: rê chuột trong khi vẽ.
	Đoạn thẳng	Vẽ đoạn thẳng theo màu vẽ và độ lớn bút vẽ hiện thời. Thao tác: Nháy chuột tại 1 đầu đoạn thẳng, kéo thả chuột đến vị trí đích và nhả chuột.
	Hình hộp	Vẽ hình hộp (Shift - hình vuông) theo màu vẽ hiện thời. Thao tác: Nháy chuột tại 1 đỉnh, kéo thả chuột đến vị trí đỉnh đối diện và nhả chuột. Có 2 lựa chọn: 
	Hình ellipse	Vẽ hình oval (Shift - hình tròn) theo màu vẽ hiện thời. Thao tác: Nháy chuột tại 1 đỉnh, kéo thả chuột đến vị trí đỉnh đối diện và nhả chuột. Có 2 lựa chọn: 
T	Văn bản	Tạo, nhập văn bản với màu là màu vẽ hiện thời. Thao tác: Nháy chuột tại vị trí muốn viết chữ, nhập đoạn văn bản từ bàn phím, nháy chuột bên ngoài để kết thúc gõ phím, dùng chuột thay đổi kích thước đối tượng văn bản, nháy chuột lần nữa để kết thúc. Lựa chọn được font chữ sau: 

Biểu tượng	Công cụ	Chức năng, thao tác sử dụng
	Tô màu	Tô màu vùng màn hình với màu vẽ. Thao tác: nháy chuột lên vùng muốn tô màu. Các lựa chọn tô màu bao gồm tổ hợp kết hợp giữa màu vẽ và màu nền. 
	Tẩy	Công cụ tẩy, xóa hình. Xóa tất cả không phân biệt màu sắc nào, trừ ra màu trong suốt. Thao tác: rê chuột để xóa.
	Công cụ chọn	Chọn 1 vùng hình chữ nhật trên màn hình để xóa. Thao tác: dùng chuột kéo thả để xác định 1 vùng chữ nhật trên hình, bấm phím Delete để xóa vùng đã chọn.
	Công cụ siêu chọn	Công cụ này cho phép chọn 1 vùng đặc biệt, sau đó xóa toàn bộ các vùng bên ngoài vùng chọn này. Thao tác: rê chuột trên màn hình để tạo 1 hình khép kín (hình này có thể có hình dạng bất kỳ). Nháy chuột bên ngoài vùng đã chọn để xóa toàn bộ bên ngoài.
	Nhân đôi	Nhân đôi 1 vùng màn hình. Thao tác: dùng chuột đánh dấu 1 vùng chữ nhật trên màn hình. Sau đó kéo thả để kéo bản nhân đôi vùng màn hình này ra chỗ khác theo ý muốn.

Các công cụ vẽ của hình dạng Vector

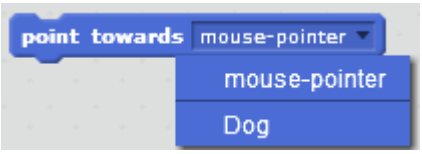
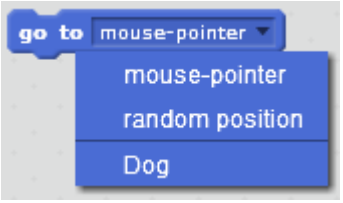
Các công cụ khi vẽ, thiết lập các hình vector.

Biểu tượng	Công cụ	Chức năng
	Công cụ chọn	Công cụ chọn đối tượng để điều chỉnh. Thao tác: nháy chuột lên 1 đối tượng đã có để bắt đầu điều chỉnh.
	Tinh chỉnh	Tinh chỉnh 1 đối tượng theo các điểm kết nối. Thao tác: - Nháy chuột lên nét vẽ để tạo điểm nối. - Rê chuột tại điểm nối để tinh chỉnh hình. 
	Bút vẽ	Công cụ vẽ tự do theo màu và độ rộng của bút vẽ. Thao tác: rê chuột để vẽ.
	Đoạn thẳng	Vẽ đoạn thẳng theo màu vẽ và độ lớn bút vẽ hiện thời. Thao tác: Nháy chuột tại 1 đầu đoạn thẳng, kéo thả chuột đến vị trí đích và nhả chuột.
	Hình hộp	Vẽ hình hộp (Shift - hình vuông) theo màu vẽ hiện thời. Thao tác: Nháy chuột tại 1 đỉnh, kéo thả chuột đến vị trí đỉnh đối diện và nhả chuột. Có 2 lựa chọn: 
	Ellipse	Vẽ hình oval (Shift - hình tròn) theo màu vẽ hiện thời. Thao tác: Nháy chuột tại 1 đỉnh, kéo thả chuột đến vị trí đỉnh đối diện và nhả chuột. Có 2 lựa chọn: 
	Văn bản	Công cụ tạo đối tượng văn bản. Thao tác: nháy chuột lên màn hình để bắt đầu gõ bàn phím tạo văn bản. Văn bản này có thể chỉnh sửa bất cứ lúc nào. Lựa chọn được font chữ sau: 
	Tô màu	Tô màu vùng bên trong của một đối tượng vector bất kỳ với màu vẽ. Thao tác: nháy chuột lên đối tượng muốn tô màu. Các lựa chọn tô màu bao gồm tổ hợp kết hợp giữa màu vẽ và màu nền.

Biểu tượng	Công cụ	Chức năng
		
	Nhân đôi	Nhân đôi 1 đối tượng. Thao tác: nháy chuột lên đối tượng muốn nhân đôi.
	Lên lớp trên	Đưa đối tượng lên trên 1 lớp (Shift → lên trên cùng).
	Xuống lớp dưới	Đưa đối tượng xuống dưới 1 lớp (Shift → xuống dưới cùng).
	Nhóm	Nhóm các đối tượng đã chọn thành 1 đối tượng mới.
	Hủy nhóm	Hủy lệnh nhóm đối tượng đã thực hiện trước đó.

PHỤ LỤC 2: CÁC TẬP LỆNH SCRATCH

Nhóm câu lệnh Move

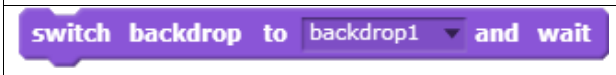
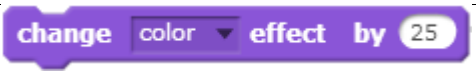
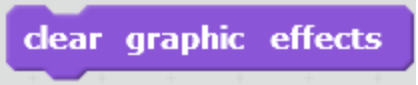
Câu lệnh	Ý nghĩa
	Dịch chuyển nhân vật <10> bước theo hướng hiện thời. Chú ý: mỗi bước = 1 pixel.
	Xoay nhân vật <15> độ theo chiều kim đồng hồ.
	Xoay nhân vật <15> độ ngược chiều kim đồng hồ.
	Quay nhân vật theo hướng được xác định (trong hình: 90 độ). Các hướng chính: Up (trên): 0º Right (phải): 90º Left (trái): -90º Down (xuống): 180º **Chú ý:** lệnh này sẽ xoay nhân vật theo tâm của mình. Mỗi nhân vật sẽ có 1 tâm duy nhất. Có thể dùng chức năng tinh chỉnh đồ họa để thay đổi vị trí tâm của nhân vật.
 	Xoay nhân vật theo hướng của 1 nhân vật khác. hoặc vị trí con trỏ chuột.
	Di chuyển nhân vật đến vị trí có tọa độ X, Y tương ứng. Lệnh này tương đương với việc thực hiện đồng thời 2 lệnh **Set X to** và **Set Y to**.  Chú ý: lệnh này có tác dụng tức thời.
 	Chuyển đến vị trí của con trỏ chuột, một nhân vật khác hoặc 1 ví trí ngẫu nhiên trên màn hình thông qua bảng chọn.

Câu lệnh	Ý nghĩa
	Di chuyển đến vị trí (X, Y) trong khoảng thời gian nhất định tính bằng giây. Thời gian có thể nhập là số thập phân, ví dụ 1.5 là 1 giây rưỡi.
	Thay đổi tọa độ X của nhân vật theo giá trị cho trong lệnh. Giá trị có thể là số dương hoặc số âm. Nếu giá trị > 0 thì nhân vật chuyển động sang phải, nếu giá trị < 0 thì nhân vật chuyển động sang trái.
	Thiết lập giá trị tọa độ X của nhân vật. Giá trị tham số của lệnh trong khoảng -240 đến 240.
	Thay đổi tọa độ Y của nhân vật theo giá trị cho trong lệnh. Giá trị có thể là số dương hoặc số âm. Nếu giá trị > 0 thì nhân vật chuyển động lên trên, nếu giá trị < 0 thì nhân vật chuyển động xuống dưới.
	Thiết lập giá trị tọa độ Y của nhân vật. Giá trị tham số của lệnh trong khoảng -180 đến 180.
	Nếu gặp cạnh màn hình, quay lại. Đây là lệnh rất quan trọng. Lệnh sẽ điều khiển nhân vật khi di chuyển gặp cạnh của màn hình thì nhân vật sẽ "bật" trở lại theo nguyên tắc đối xứng gương.
	Thiết lập kiểu quay của nhân vật (ví dụ khi gặp cạnh màn hình). Có 3 kiểu quay: trái - phải (right-left), không quay (do not rotate), quay tròn (all around).
	(hàm) trả lại giá trị tọa độ X của nhân vật.
	(hàm) trả lại giá trị tọa độ Y của nhân vật.
	(hàm) trả lại giá trị của hướng nhân vật tính bằng số đo góc.

Nhóm câu lệnh Look

Câu lệnh	Ý nghĩa
	Thể hiện dòng chữ "Hello" trong 1 khoảng thời gian <2> giây. Trong thời gian đó chương trình sẽ tạm dừng.
	Thể hiện dòng chữ "Hello" trong khi chương trình vẫn chạy bình thường.
	Thể hiện dòng chữ suy nghĩ "Hmm…" trong 1 khoảng thời gian

Câu lệnh	Ý nghĩa
	<2> giây. Trong thời gian đó chương trình sẽ tạm dừng.
think Hmm...	Thể hiện dòng chữ suy nghĩ "Hmm…" trong khi chương trình vẫn chạy bình thường.
show	Hiện nhân vật trên màn hình.
hide	Ẩn (không hiện) nhân vật trên màn hình.
switch costume to costume2	Thay đổi kiểu trang phục của nhân vật sang dạng <costume2> được hiện trong danh sách bên phải.
next costume	Thay đổi kiểu trang phục của nhân vật sang dạng tiếp theo (trong danh sách).
switch backdrop to backdrop1	Thay đổi sân khấu nền sang dạng <backdrop1> trong danh sách chọn bên phải.
change color effect by 25	Thay đổi hiệu ứng <màu sắc> bên ngoài của nhân vật theo giá trị <25> cho trước. Các hiệu ứng được chọn trong danh sách, như color, fisheye,.. 
set color effect to 0	Thiết lập giá trị của hiệu ứng bên ngoài của nhân vật theo giá trị cho trước. Các hiệu ứng được chọn trong danh sách, như color, fisheye,… 
clear graphic effects	Hủy tất cả hiệu ứng thể hiện của nhân vật, quay trở về trạng thái ban đầu.
change size by 10	Thay đổi kích thước của nhân vật theo giá trị <10> % cho trước so với

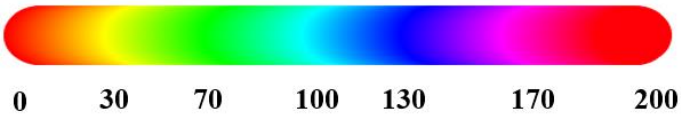
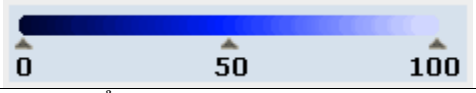
Câu lệnh	Ý nghĩa
	kích thước hiện thời. Giá trị có thể là số dương hoặc âm.
	Thiết lập kích thước của nhân vật là <100>% so với kích thước gốc.
	Chuyển nhân vật lên lớp trên cùng.
	Chuyển nhân vật xuống dưới <1> lớp.
	(hàm) trả lại số thứ tự của trang phục hiện thời của nhân vật.
	(hàm) trả lại tên của nền sân khấu hiện thời.
	(hàm) trả lại kích thước hiện thời của nhân vật.
Các lệnh riêng cho sân khấu	
	Thay đổi sân khấu nền sang dạng <backdrop1> trong danh sách chọn bên phải.
	Thay đổi sân khấu nền sang dạng trong danh sách chọn bên phải và chờ thực hiện các lệnh của lệnh điều khiển sau: 
	Chuyển nền sân khấu sang kiểu tiếp theo.
	Thay đổi hiệu ứng bên ngoài của sân khấu theo giá trị <25> cho trước. Các hiệu ứng được chọn trong danh sách, như color, fisheye, …
	Thiết lập giá trị của hiệu ứng bên ngoài của nền sân khấu theo giá trị cho trước. Các hiệu ứng được chọn trong danh sách, như color, fisheye, …
	Hủy tất cả hiệu ứng thể hiện của sân khấu, quay trở về trạng thái ban đầu.
	(hàm) trả lại số thứ tự của nền sân khấu hiện thời trong danh sách.
	(hàm) trả lại tên của nền sân khấu hiện thời.

Nhóm câu lệnh Sound

Câu lệnh	Ý nghĩa
play sound meow	Bật âm thanh <meow> theo âm thanh trong danh sách bên phải và không dừng chạy chương trình.
play sound meow until done	Bật âm thanh <meow> theo âm thanh trong danh sách bên phải, chương trình sẽ dừng lại đợi kết thúc âm thanh này.
stop all sounds	Tắt tất cả âm thanh.
play drum 1 for 0.25 beats	Đánh trống với tốc độ <0.25> (1/4) của nhịp trống. Lựa chọn kiểu trống trong danh sách được đánh số.
rest for 0.25 beats	Tạm nghỉ đánh trống trong thời gian <0.25> (1/4) nhịp trống.
play note 60 for 0.5 beats	Chơi nốt nhạc <60> (nốt C) trong danh sách trong thời gian <0.5> nhịp trống.
set instrument to 1	Chọn công cụ chơi nhạc <1> trong danh sách.
change volume by -10	Thay đổi độ lớn âm thanh theo tỉ lệ <-10> phần trăm so với hiện thời. Cho phép nhập số dương và âm.
set volume to 100 %	Thiết lập độ lớn âm thanh theo mức <100> phần trăm của loa hệ thống (máy tính).
volume	(hàm) trả lại độ lớn âm thanh hiện thời của loa hệ thống.
change tempo by 20	Thay đổi nhịp trống, tăng lên (hoặc giảm đi) <20> nhịp trống trong mỗi phút.
set tempo to 60 bpm	Thiết lập nhịp trống theo <60> nhịp đập của trống trong 1 phút.
tempo	(hàm) trả lại giá trị nhịp trống hiện thời là số lượng nhịp trống trong 1 phút.

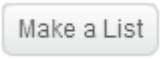
Nhóm câu lệnh Pen

Câu lệnh	Ý nghĩa
clear	Xóa tất cả các hình đồ họa được vẽ trước đó bởi bút (pen).
stamp	In (hiển thị) hình ảnh nhân vật trên màn hình (chú ý: hình ảnh này không nằm trong sân khấu).
pen down	Đặt chế độ hạ bút để bắt đầu vẽ theo vết chuyển động của nhân vật.
pen up	Đặt chế độ nhấc bút không vẽ nữa.

Câu lệnh	Ý nghĩa
set pen color to	Đặt màu sắc cho bút theo mẫu màu. Dùng chuột nháy chọn vị trí để chọn màu.
change pen color by 10	Thay đổi màu của bút theo giá trị <10> được nhập trực tiếp. Cho phép giá trị âm hoặc dương. Giá trị màu được thiết lập theo hiệu ứng màu sắc (color effect).
set pen color to 0	Thiết lập màu cho bút theo giá trị màu. Màu sắc có giá trị từ 0 đến 200 phân bổ như sau:  0 30 70 100 130 170 200
change pen shade by 10	Thay đổi độ mờ của bút theo giá trị <10>. Độ mờ được đo bởi các giá trị từ 0 (màu chuẩn, thẫm nhất, đến 100, mờ nhạt nhất).  0 50 100
set pen shade to 50	Thiết lập độ mờ của bút theo giá trị <50>. Độ mờ được đo bởi các giá trị từ 0 (màu chuẩn, thẫm nhất, đến 100, mờ nhạt nhất).  0 50 100
change pen size by 1	Thay đổi kích thước của bút theo <1> pixel. Giá trị thay đổi có thể dương hoặc âm.
set pen size to 1	Thiết lập kích thước bút theo giá trị <1> pixel.

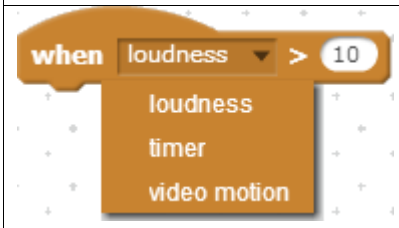
Nhóm câu lệnh Variable

Câu lệnh	Ý nghĩa
Make a Variable	Tạo một biến nhớ mới.
a	(hàm) trả lại giá trị hiện thời của biến nhớ.
a	Lựa chọn có hiển thị hay không giá trị biến nhớ trên màn hình. Thể hiện của biến nhớ có dạng sau:  a 0
set a to 0	Gán 1 giá trị cụ thể cho biến nhớ "a". Giá trị có thể là số hoặc chữ.
change a by 1	Thay đổi biến nhớ "a" theo giá trị số (1) được nhập trực tiếp tại ô phía phải.

Câu lệnh	Ý nghĩa
	Chú ý: giá trị bắt buộc phải là số, có thể dương hoặc âm. Nếu nhập sai thì lỗi sẽ được thông báo bởi "NaN". 
	Hiện giá trị biến nhớ "a" trên màn hình. 
	Ẩn không hiện giá trị biến nhớ "a" trên màn hình. 
	Tạo một danh sách (mảng) giá trị mới.
	(hàm) trả lại giá trị của biến nhớ "b" là một dãy số hoặc văn bản. 
	Bổ sung giá trị <"thing"> vào cuối của mảng "b". Giá trị có thể là số hoặc chữ.
	Xóa phần tử thứ <1> khỏi mảng "b".
	Chèn giá trị <"thing"> vào mảng "b" tại vị trí trước phần tử thứ <1>. Mảng "b" sẽ tăng thêm 1 phần tử. Giá trị có thể là số hoặc chữ.
	Thay thế phần tử thứ <1> của mảng "b" bằng giá trị mới <"thing">. Giá trị này có thể là số hoặc chữ.
	(hàm) trả lại giá trị của phần tử thứ <1> của mảng "b".
	(hàm) trả lại giá trị là số phần tử hiện có của mảng "b".
	(hàm logic) trả lại giá trị đúng nếu mảng "b" chứa phần tử có giá trị <"thing">.
	Thể hiện mảng "b" trên màn hình.

Câu lệnh	Ý nghĩa
	Ẩn không hiện mảng "b" trên màn hình.

Nhóm câu lệnh Event

Câu lệnh	Ý nghĩa
	Đây là lệnh "bắt đầu chương trình". Chương trình (nhóm lệnh) sẽ chạy khi nháy lên lá cờ xanh.
	Lệnh điều khiển bàn phím. Nhóm lệnh sẽ chạy khi phím tương ứng <Space> được bấm.
	Lệnh điều khiển chuột. Nhóm lệnh sẽ chạy khi nháy chuột lên nhân vật.
	Lệnh điều khiển theo sân khấu. Nhóm lệnh sẽ chạy khi nền sân khấu thay đổi sang <backdrop1>.
	Lệnh điều khiển phụ thuộc vào độ âm (loudness), thời gian (timer) và tốc độ của chuyển động (video motion). Khi các điều kiện trên thỏa mãn thì nhóm lệnh sẽ chạy.
	Lệnh điều khiển thông điệp. Nhóm lệnh sẽ được thực hiện khi nhân vật nhận được thông điệp <message1>.
	Lệnh gửi thông điệp cụ thể <message1>. Lệnh "when I receive the message" có tác dụng trả lời cho thông điệp này. Các lệnh khác vẫn chạy bình thường. Chú ý: trong lệnh này có chức năng tạo 1 thông điệp mới.
	Lệnh gửi thông điệp cụ thể <message1>. Khi gửi thông điệp chương trình sẽ tạm dừng và chờ cho đến khi tất cả các nhân vật khác thực hiện xong các lệnh khi nhận thông điệp này. Sau đó chương trình sẽ quay lại thực hiện các lệnh tiếp theo. Chú ý: trong lệnh này cũng có chức năng tạo 1 thông điệp mới. Ví dụ:

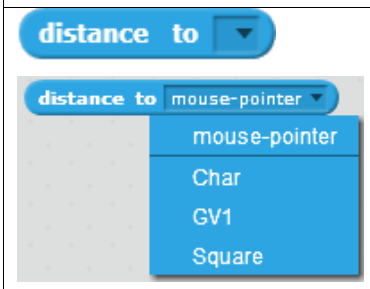
Câu lệnh	Ý nghĩa
	 Trong ví dụ trên: nếu người dùng nháy phím Space, nhóm trên sẽ gửi thông điệp <meow> và chờ cho đến khi nhóm thứ 2 nhận thông điệp và thể hiện <Meow!> trong 2 giây. Sau đó nhóm thứ nhất thực hiện tiếp lệnh "hide".

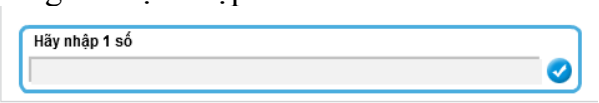
Nhóm câu lệnh Control

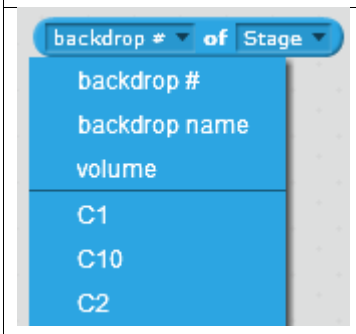
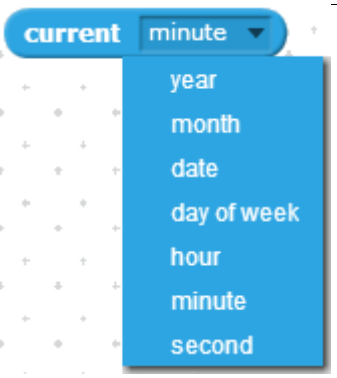
Câu lệnh	Ý nghĩa
	Chương trình sẽ tạm dừng trong <1> giây.
	Nhóm lệnh trong khung sẽ được thực hiện lặp lại <10> lần.
	Nhóm lệnh trong khung sẽ được thực hiện lặp lại vô hạn lần.
	Lệnh điều khiển có điều kiện If. Nếu điều kiện trong biểu thức lệnh là Đúng thì nhóm lệnh trong khung sẽ được chạy.
	Lệnh điều khiển rẽ nhánh If-then-else. Nếu điều kiện trong biểu thức lệnh là Đúng thì nhóm lệnh trong khung trên sẽ được chạy, ngược lại nếu điều kiện trong biểu thức lệnh là Sai thì nhóm lệnh trong khung dưới (else) sẽ được chạy.
	Lệnh "dừng cho đến khi". Chương trình sẽ tạm dừng cho đến khi điều kiện trong biểu thức lệnh trả lại giá trị Đúng.
	Lệnh "lặp cho đến khi". Nhóm lệnh trong khung sẽ được thực hiện lặp lại cho đến khi điều kiện trong biểu thức lệnh trả lại giá trị Đúng. Lệnh này kiểm tra điều kiện trước, nếu đúng mới thực hiện nhóm

Câu lệnh	Ý nghĩa
	lệnh. Lệnh này tương đương lệnh **While … do** của các ngôn ngữ bậc cao khác.
	Lệnh "dừng chạy". Lệnh này sẽ dừng chạy theo các khả năng lựa chọn từ ô bên phải. Có thể chọn "all" - dừng tất cả các chương trình; "this script" - chỉ dừng chương trình hiện tại; "other scripts in sprite" - dừng tất cả các chương trình tương ứng với nhân vật hiện thời.
	Lệnh "khi tôi bắt đầu là một bản sao". Các lệnh trong khối này sẽ được thực hiện khi nhân vật bản sao (phân thân) được tạo ra.
	Lệnh thực hiện việc tạo ra 1 bản sao (phân thân) của 1 nhân vật. Nhân vật sẽ được tạo bản sao (phân thân) có thể chọn từ danh sách. "itself" là tạo phân thân cho chính mình. Có thể tạo phân thân cho nhân vật khác.
	Lệnh xóa đi 1 bản sao (phân thân) hiện thời.

Nhóm câu lệnh Sensing

Câu lệnh	Ý nghĩa
	Điều kiện cảm biến va chạm. Có thể chọn vật thể va chạm từ danh sách là các nhân vật, con trỏ chuột hay cạnh màn hình. Nếu có va chạm thì hàm giá trị sẽ trả lời giá trị Đúng.
	Điều kiện cảm biến va chạm màu sắc. Màu được chọn bằng cách nháy chuột lên vị trí có màu muốn chọn. Hàm trả lại giá trị đúng khi nhân vật va chạm với màu sắc đã chọn.
	Điều kiện cảm biến màu sắc va chạm màu sắc. Màu sắc có thể chọn bằng cách nháy chuột lên vị trí màu. Hàm sẽ trả lại giá trị Đúng khi 2 màu này va chạm nhau.
	Hàm số trả lại khoảng cách từ (tâm) nhân vật hiện thời đến 1 nhân vật khác, hoặc đến vị trí con trỏ chuột.
	Lệnh "hỏi và chờ". Nhân vật sẽ thể hiện câu hỏi <What's your name> và chờ nhập

Câu lệnh	Ý nghĩa
	dữ liệu từ bàn phím. Giá trị được nhập từ bàn phím (số hoặc chữ) sau khi bấm Enter sẽ được lưu trong biến nhớ "answer". Hình ảnh thể hiện khi thực hiện lệnh này.  Nếu lệnh được thực hiện bởi nền sân khấu thì giao diện nhập như sau: 
answer	Biến nhớ "Answer" dùng để lưu giá trị nhập dữ liệu từ bàn phím của lệnh trên.
key space pressed?	Cảm biến bàn phím. Hàm này sẽ trả lại giá trị **true** (Đúng) nếu phím tương ứng <Space> được bấm, ngược lại trả lại giá trị **false** (Sai)
mouse down?	Cảm biến chuột. Hàm sẽ trả lại giá trị đúng nếu chuột được nhấn (chuột trái).
mouse x	(hàm) trả lại tọa độ X của vị trí chuột hiện thời.
mouse y	(hàm) trả lại tọa độ Y của vị trí chuột hiện thời.
loudness	(hàm) trả lại giá trị hiện thời của âm thanh loa hệ thống.
video motion on this sprite	(hàm) trả lại giá trị là các tham số liên quan đến video camera hiện thời tương ứng với nhân vật hoặc sân khấu. Các tham số có thể tính: - motion: số lượng chuyển động. - direction: hướng chuyển động
turn video on	Lệnh bật / tắt camera của máy tính. Camera sẽ dùng để điều khiển hoạt động và nhân vật trên màn hình.
set video transparency to 50 %	Lệnh thiết lập độ trong suốt của video camera trên màn hình. Giá trị có thể thiết lập từ 0% đến 100%. Vậy video có độ trong suốt = 100% sẽ không hiện trên màn hình.

Câu lệnh	Ý nghĩa
	(hàm) trả lại giá trị thời gian bằng giây tính từ lúc chương trình bắt đầu chạy hoặc từ khi thực hiện lệnh **reset timer**.
	Lệnh có tác dụng đặt lại tham số "timer" về 0.
	(hàm) trả lại giá trị các tham số là thuộc tính riêng của nhân vật được chọn tại ô bên phải lệnh. Các tham số có thể trả lại giá trị bao gồm: - x position: tọa độ X - y position: tọa độ Y - direction: hướng - costume #: số thứ tự của trang phục hiện thời. - costume name: tên của trang phục hiện thời. - size: kích thước nhân vật hiện thời trên màn hình. - volume: độ lớn của âm thanh hiện thời của nhân vật. **Chú ý:** các biến nhớ riêng của nhân vật sẽ nằm trong danh sách thuộc tính và có trong lệnh này. **Chú ý:** không có tên các biến nhớ dùng chung trong danh sách này. Các biến nhớ chung sẽ có trong thuộc tính của sân khấu (xem lệnh tiếp theo).
	(hàm) trả lại các giá trị tham số thuộc tính của sân khấu và các biến nhớ chung. backdrop #: số thứ tự của nền sân khấu hiện thời, backdrop name: tên của nền sân khấu hiện thời. - volume: độ lớn của âm thanh nền sân khấu hiện thời.
	(hàm) trả lại giá trị thời gian hiện thời được tính theo tham số chọn trong ô của lệnh: - year: năm (ví dụ: 2016) - month: tháng (ví dụ: 3, 1→12) - date: ngày của tháng (ví dụ: 20, 1→31) - day of week: ngày của tuần (ví dụ: 2, 1→7) - hour: giờ hiện thời (ví dụ: 14, 0→23) - minute: phút hiện thời (ví dụ: 56. 0→59) - second: giây hiện thời (ví dụ: 59. 0→59)

Câu lệnh	Ý nghĩa
	(hàm) trả lại số ngày của thế kỷ 21, tính từ năm 2000.
	(hàm) trả lại tên tài khoản người dùng trên trang scratch.mit.edu.

Nhóm câu lệnh Operators

Câu lệnh	Ý nghĩa
	(hàm) phép cộng 2 số
	(hàm) phép trừ 2 số
	(hàm) phép nhân 2 số
	(hàm) phép chia 2 số, kết quả thập phân. Chú ý: nếu 2 số là nguyên, muốn tính phép chia nguyên thì sử dụng biểu thức: 
	(hàm) trả lại số ngẫu nhiên giữa 2 giá trị số.
	Biểu thức logic mô tả quan hệ giữa 2 số. Biểu thức trả lại giá trị Đúng nếu số thứ nhất < số thứ 2.
	Biểu thức logic mô tả quan hệ giữa 2 số. Biểu thức trả lại giá trị Đúng nếu số thứ nhất > số thứ 2.
	Biểu thức logic mô tả quan hệ giữa 2 số. Biểu thức trả lại giá trị Đúng nếu số thứ nhất = số thứ 2.
	Biểu thức logic "and" (và).
	Biểu thức logic "or" (hoặc)
	Biểu thức logic phủ định "not".
	(hàm) toán tử nối 2 xâu ký tự.
	(hàm) trả lại ký tự thứ <1> của xâu ký tự <world>
	(hàm) trả lại độ dài của xâu ký tự <world>
	(hàm) phép toán tính số dư của phép chia số nguyên "mod".
	(hàm) phép tính làm tròn số số nguyên gần nhất có thể.
	(hàm) trả lại giá trị là kết quả của các hàm toán học của 1 số. Hàm toán học được chọn từ danh sách trên dòng lệnh.

Câu lệnh	Ý nghĩa
	**abs:** trị tuyệt đối **floor:** làm tròn xuống **celling:** làm tròn lên **sqrt:** căn bậc 2 **sin:** hàm lượng giác sin. **cos:** hàm lượng giác cosin. **tan:** hàm lượng giác tan. **asin: :** hàm lượng giác arcsin. **acos: :** hàm lượng giác arccos. **atan: :** hàm lượng giác arctang. **ln:** logarit cơ số e. **log:** logarit cơ số 10. **e^:** hàm mũ e. **10^:** hàm mũ 10.

INDEX

Tài liệu tham khảo

[1]. Learn to Program with Scratch. Majed Marji.

[2]. Scratch 2.0 Game Development. Sergio van Pul, Jessica Chiang.

[3]. Computer Science Concepts in Scratch. Michal Armoni and Moti BenAri, 2013.

http://stwww.weizmann.ac.il/g-cs/scratch/scratch-14-textbook-1-0-one-side.pdf

[4]. Program or be Programmed: Ten Commands for a Digital Age. Rushkoff, D. OR Books, 2009.

[5]. CS unplugged. Giảng dạy khoa học máy tính không cần máy tính. http://www.csunplugged.org

[6]. Chương trình, chuẩn kiến thức môn Tin học. NXBGD, Hà Nội, 2000.

[7]. Chương trình giáo dục phổ thông tổng thể trong chương trình giáo dục phổ thông mới. Tài liệu chính thức Bộ Giáo dục và Đào tạo, 2015.

[8]. Computing in the national curriculum A guide for primary teachers.
Tài liệu dành cho GV cấp Tiểu học Anh quốc dạy môn Tin học.
http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

[9]. Shut Down or Restart.
https://royalsociety.org/topics-policy/projects/computing-in-schools/report/
https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

[10]. Trang chính của Scratch.
https://scratch.mit.edu/

[11]. Computing at School, The Raspberry Pi Education Manual (CAS, 2012).
http://pi.cs.man.ac.uk/download/Raspberry_Pi_Education_Manual.pdf

[12]. Scratch Programming For Teens. Jerry Lee Ford, Jr.

[13]. Super Scratch Programming Adventure! The LEAD Project

[14]. Starting from Scratch.
An Introduction to Computing Science. Jeremy Scott.

[15]. Computational thinking. A guide for teachers. Andrew Csizmadia, Simon Humphreys, National Coordinator, Computing At School.

[16]. Computer Science: A curriculum for schools.

http://www.computingatschool.org.uk/

[17]. K12-Computer Science Framework.

https://k12cs.org/wp-content/uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf.