HƯỚNG DẪN HỌC TẬP **HOÀN THIỆN** KIẾN THỨC TRUNG HỌC

EVERYTHING YOU NEED TO ACE
COMPUTER SCIENCE & CODING
IN ONE BIG FAT NOTEBOOK

SỔ TAY KHOA HỌC MÁY TÍNH & MÃ HÓA

Workman Publishing & Contributor

Cassini *dịch*

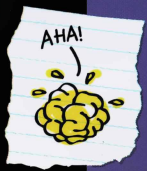
Những ghi chú đến từ **HỌC SINH THÔNG MINH** nhất lớp
(Đã được kiểm tra kỹ lưỡng bởi những **GIÁO VIÊN TÀI NĂNG**)

Achaubooks
BOOKS & STEM

NXB Lao Động

From the BRAINS behind BRAIN QUEST®!

Được tạo ra bởi ban biên tập của **BRAIN QUEST®**:
Bộ sách giáo dục bán chạy nhất nước Mỹ!



Cuốn sổ "**TO BỰ**" này giúp bạn "**THẤM**" tất cả kiến thức học ở trường:

BẰNG Khái niệm chính · Phương pháp ghi nhớ mnemonic · Định nghĩa · Biểu đồ · Và hình minh họa

Để giúp bạn hiểu **KHOA HỌC MÁY TÍNH**

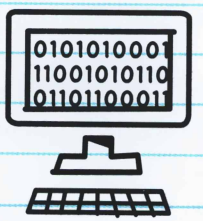
Bao gồm: **HỆ THỐNG MÁY TÍNH** Mã nhị phân Thuật toán Tư duy máy tính Vòng lặp, sự kiện thủ tục **LẬP TRÌNH** bằng **SCRATCH** và **PYTHON** **BIỂU THỨC BOOLEAN** Lập trình web **AN NINH MẠNG** **HTML** CSS Và hơn thế nữa

CHINH PHỤC **TẤT CẢ** CÁC MÔN Ở TRƯỜNG TRUNG HỌC
với tuyển tập sở hữu **NHIỀU TRIỆU BẢN!**

Achaubooks
BOOKS & STEM

SỔ TAY KHOA HỌC MÁY TÍNH VÀ MÃ HÓA
ISBN: 978-604-343-279-4
9 786043 432794
GIÁ: 495.000 VND

SỔ TAY KHOA HỌC MÁY TÍNH VÀ MÃ HÓA



ALL RIGHT RESERVED
Vietnam edition copyright © A Chau International Education Development and Investment Corporation, 2022.
All rights reserved. No part of this publicatioon may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the publishers.
ISBN: 978-604-343-279-4
Printed in Viet Nam

Bản quyền tiếng Việt thuộc về
Công ty Cổ phần Đầu tư và Phát triển Giáo dục Quốc tế Á Châu,
xuất bản theo hợp đồng chuyển nhượng bản quyền với Workman Publishing Co., Inc 2020.
Bản quyền tác phẩm đã được bảo hộ, mọi hình thức xuất bản, sao chụp, phân phối dưới dạng in ấn, văn bản điện tử, đặc biệt là phát tán trên mạng internet mà không được sự cho phép của đơn vị nắm giữ bản quyền là hành vi vi phạm bản quyền và làm tổn hại tới lợi ích của tác giả và đơn vị đang nắm giữ bản quyền.
Không ủng hộ những hành vi vi phạm bản quyền.
Chỉ mua bán bản in hợp pháp.

ĐƠN VỊ PHÁT HÀNH:
Công ty Cổ Phần Đầu Tư và Phát Triển Giáo Dục Quốc Tế Á Châu
Số 8, lô 2 Dự án nhà ở Phùng Khoang, Phường Trung Văn,
Quận Nam Từ Liêm, Thành phố Hà Nội
Điện thoại: 024 8582 5555
Hotline hỗ trợ: Tại Hà Nội: 0916 640 166
Tại Hồ Chí Minh: 0812 990 996
Website: http://achaubooks.vn
http://hocgioitoan.com.vn
Email: info@achaubooks.vn
Facebook: Fb.com/hocgioitoan.com.vn

HƯỚNG DẪN HỌC TẬP **HOÀN THIỆN** KIẾN THỨC TRUNG HỌC

EVERYTHING YOU NEED TO ACE
COMPUTER SCIENCE & CODING
IN ONE BIG FAT NOTEBOOK

SỔ TAY KHOA HỌC MÁY TÍNH & MÃ HÓA

Workman Publishing & Contributor

Cassini *dịch*

Những ghi chú đến từ **HỌC SINH THÔNG MINH** nhất lớp
(Đã được kiểm tra kỹ lưỡng bởi những **GIÁO VIÊN TÀI NĂNG**)

Achaubooks BOOKS & STEM

NXB Lao Động

MỌI THỨ BẠN CẦN BIẾT ĐỂ CHINH PHỤC

KHOA HỌC MÁY TÍNH VÀ MÃ HÓA

XIN CHÀO!

Trong cuốn sách này, bạn sẽ tìm thấy mọi thứ mình cần để chinh phục khoa học máy tính và lập trình: từ việc hiểu hệ thống máy tính tới việc đọc và viết bằng ngôn ngữ lập trình; từ việc sử dụng thuật toán cơ bản tới việc viết biểu thức Boolean; từ việc sử dụng Scratch và Python tới việc khám phá phát triển web. Đây đều là những điều vô cùng quan trọng bạn cần để học khoa học máy tính và xây dựng nền tảng kiến thức lập trình.

Những nội dung trong sách khá rõ ràng, và được thể hiện theo định dạng dưới đây để giúp bạn dễ nắm bắt:

- Từ vựng được tô **MÀU VÀNG**
- Định nghĩa được đặt trong khung
- Những người, địa điểm, thời gian và thuật ngữ quan trọng được tô màu xanh
- Hình phác thảo, hình minh họa và biểu đồ cho thấy ý chính
- Ý chính được gạch chân dưới

Nếu bạn gặp khó khăn trong lớp khoa học máy tính hoặc gặp vấn đề với dự án lập trình, cuốn sách này sẽ hữu ích cho bạn. Nó bao gồm những mục chính trong khoa học máy tính và hướng dẫn bạn ở mức cơ bản về Scratch, Python và phát triển web. Đây chính là một nguồn kiến thức tuyệt vời! Cho dù bạn muốn gợi nhớ lại kiến thức đã tiếp thu, củng cố bài học trên lớp, hoặc tự học kiến thức cơ bản, cuốn sách này đều có thể giúp.

MỤC LỤC

BÀI 1: HỆ THỐNG MÁY TÍNH 1

1. Khoa học máy tính là gì? 2
2. Máy tính là gì? 13
3. Chúng ta tương tác với máy tính như thế nào? 29

BÀI 2: DỮ LIỆU VÀ PHÂN TÍCH 39

4. Lưu trữ thông tin 40
5. Thu thập và sử dụng thông tin 57

BÀI 3: KỸ THUẬT PHẦN MỀM 67

6. Thiết kế hệ thống máy tính **68**
7. Thử nghiệm **73**
8. Ghi chú **81**
9. Tích hợp phản hồi **89**
10. Hợp tác **97**

BÀI 4: THUẬT TOÁN VÀ LẬP TRÌNH 109

11. Sử dụng thuật toán **110**
12. Ngôn ngữ lập trình **121**
13. Tư duy máy tính **139**

BÀI 5: NGUYÊN TẮC CHUNG CỦA LẬP TRÌNH 147

14. Biến 148
15. Câu lệnh điều kiện 163
16. Vòng lặp 179
17. Sự kiện 187
18. Thủ tục 191

BÀI 6: LẬP TRÌNH BẰNG SCRATCH 199

19. Khởi động 200
20. Thuật toán cơ bản 223
21. Dữ liệu và Toán tử 247
22. Khối điều khiển và khối sự kiện 273
23. Tái sử dụng kịch bản 299

BÀI 7: LẬP TRÌNH BẰNG PYTHON 309

24. Khởi động cùng Python 310
25. Biến trong Python 323
26. Chuỗi 333
27. Số làm biến 351
28. Danh sách và biểu thức Boolean 365
29. Vòng lặp *For* 381
30. Vòng lặp *While* và vòng lặp lồng ghép 393
31. Câu lệnh điều kiện 403
32. Hàm 423

BÀI 8: PHÁT TRIỂN WEB 443

33. Internet là gì? 444
34. An ninh mạng 463
35. Tạo một trang web cơ bản 481
36. Element văn bản của HTML 497
37. Element liên kết 511
38. Tạo phong cách với CSS 523
39. Tạo phong cách cho element riêng biệt với CSS 539

CHỈ MỤC 555

BÀI 1

Hệ thống máy tính

Chương 1

KHOA HỌC MÁY TÍNH LÀ GÌ?

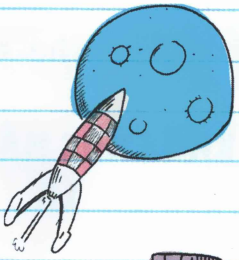
MÁY TÍNH là một thiết bị lưu trữ và xử lý thông tin (thay đổi, di chuyển và viết lại). Nó có thể thực hiện các phép tính phức tạp, sắp xếp và lưu trữ lượng thông tin khổng lồ. Khoa học máy tính là ngành nghiên cứu máy tính và cách ứng dụng công nghệ máy tính xử lý vấn đề. Nó liên quan đến **HỆ THỐNG MÁY TÍNH**, nguyên tắc lập trình, dữ liệu và phân tích, mạng, internet, và cách máy tính ảnh hưởng đến cuộc sống của con người. Máy tính vượt trội hơn con người trong việc lưu trữ và chia sẻ lượng lớn thông tin, giải nhanh các phép tính phức tạp, và học hỏi.

HỆ THỐNG MÁY TÍNH
Tất cả những phần cứng cơ bản (những phần bạn có thể thấy và sờ) và phần mềm (chương trình) cùng hoạt động để máy tính có thể chạy.

Các nhà khoa học máy tính sử dụng khả năng của máy tính để phát triển những công nghệ mới, nhằm giúp cuộc sống của ta dễ dàng hơn.

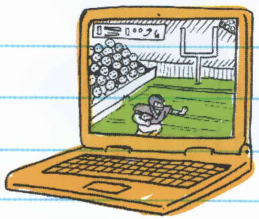
Công nghệ máy tính ảnh hưởng đến cách chúng ta sống và suy nghĩ. Nó có thể giúp chúng ta giải quyết các vấn đề, làm cuộc sống của ta dễ dàng hơn, vui hơn và an toàn hơn.

Ví dụ:



- Máy tính giúp điều hướng tàu vũ trụ tới mặt trăng và sao Hỏa.

- Rô-bốt giúp bác sĩ thực hiện phẫu thuật với độ chính xác cao.



- Máy tính tạo ra hiệu ứng âm thanh và hình ảnh thực tế cho phim ảnh và trò chơi điện tử.

Các nhà khoa học máy tính sử dụng và tạo ra công nghệ để giải quyết vấn đề. Họ lập trình máy tính để hoàn thành công việc tốt hơn và nhanh hơn, đồng thời dạy chúng cách xử lý những công việc mới. Trở thành một nhà khoa học máy tính, bạn sẽ chuyển từ một **người tiêu dùng** đơn thuần (người sử dụng một thứ gì đó) sang một **người sáng tạo.**

Khoa học máy tính là một cách giải quyết vấn đề. Bạn sẽ cần nghiên cứu máy tính, thiết kế của chúng, và cách chúng xử lý thông tin.

Ví dụ về khoa học máy tính là gì và không là gì:

KHOA HỌC MÁY TÍNH LÀ	KHOA HỌC MÁY TÍNH KHÔNG LÀ
Tạo ra một ứng dụng điện thoại (ứng dụng hoặc chương trình) để bạn bè chia sẻ cho nhau những video mèo hài hước.	Xem video mèo lái xe đạp trên mạng internet.
Tự lập trình trò chơi điện tử.	Chơi trò chơi điện tử yêu thích.

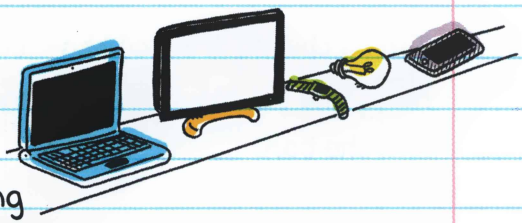
KHOA HỌC MÁY TÍNH LÀ	KHOA HỌC MÁY TÍNH KHÔNG LÀ
Tạo ra chương trình có khả năng giải bài tập toán về nhà cho bạn.	Làm bài tập toán trên máy tính của trường.
Tạo ra chương trình tự động bình luận tốt về ảnh của bạn bè.	Bình luận bức ảnh mới nhất mà bạn bè vừa đăng tải.

Tất cả những ví dụ bên **TAY PHẢI** đều chỉ *sử dụng* công nghệ, còn những ví dụ bên **TAY TRÁI** là *tạo ra* công nghệ mới.

5 LĨNH VỰC TRONG KHOA HỌC MÁY TÍNH

Khoa học máy tính có thể được chia thành 5 lĩnh vực (chính):

1. Hệ thống máy tính



Hệ thống máy tính là những thiết bị chạy chương trình và xử lý thông tin. Ví dụ như máy tính để bàn, máy tính xách tay và điện thoại. Nhiều thiết bị điện tử được chạy bởi **CHƯƠNG TRÌNH**[1] và có máy tính nhỏ cài bên trong. Ví dụ, máy rửa bát, TV, đồng hồ thông minh, và thậm chí là một số bóng đèn.

2. Thuật toán và lập trình

THUẬT TOÁN[3] và lập trình bao gồm việc tạo chương trình (code) để máy tính thực hiện. Chương trình có thể rất phức tạp - ví dụ như iTunes, Safari, Chrome, hoặc Roblox - và tuân theo nhiều chỉ dẫn khác nhau.

CHƯƠNG TRÌNH
Tập hợp các chỉ thị (hoặc thuật toán) được dịch sang các lệnh mà máy tính có thể hiểu được (code)[2].

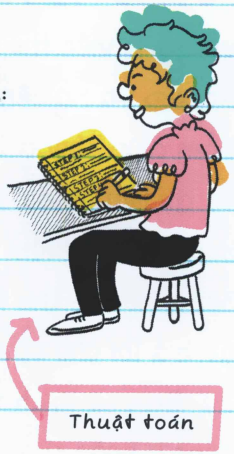
THUẬT TOÁN
Một danh sách các bước chỉ dẫn được viết bằng ngôn ngữ của con người, có mục đích hướng dẫn một người cách hoàn thành công việc.

[1] Program.
[2] Code (danh từ): đoạn mã.
[3] Algorithm.

Chúng cũng có thể rất đơn giản, như một chương trình chỉ hiển thị thông điệp "Hello, World!"[1]

Đôi khi viết chương trình gồm 2 bước:

BƯỚC 1: Nhà khoa học máy tính lập một danh sách chỉ dẫn từng bước về điều cô ấy muốn chương trình làm.



BƯỚC 2: Nhà khoa học máy tính dịch chỉ thị thành ngôn ngữ máy tính có thể hiểu (code). Bây giờ thuật toán đã là một chương trình.

Máy tính rất **mạnh**, nhưng chúng không thông minh đến thế. Chúng chỉ có thể làm điều được dạy. Chương trình chỉ dẫn cho máy tính cách thực hiện.

[1] Xin chào, thế giới!

3. Dữ liệu và Phân tích

DỮ LIỆU
Những thông tin thô, chưa sắp xếp.

PHÂN TÍCH
Tổ chức, mô tả, và hiểu về dữ liệu.

DỮ LIỆU[1] và **PHÂN TÍCH**[2] liên quan đến việc thu thập, lưu trữ, sắp xếp và nghiên cứu thông tin nhằm giúp chúng ta hiểu ý nghĩa của dữ liệu. Ví dụ, trong một chương trình kiểm tra tính cách, chúng ta cần thu thập những câu trả lời cho câu hỏi (giả dụ như "Bạn có thích nhiều bạn bè sang nhà chơi điện tử không?" hoặc "Bạn muốn tận hưởng thời gian rảnh rỗi của mình như thế nào?"), lưu trữ câu trả lời, và nghiên cứu chúng để hiểu về tính cách một người. Một cách sử dụng dữ liệu đơn giản khác là thu thập và lưu trữ số điểm trong trò chơi điện tử, như vậy chúng ta có thể biết liệu người chơi đã qua màn hoặc ai có số điểm cao nhất.

Máy tính vô cùng hữu ích trong việc lưu trữ, chia sẻ, và tính toán một lượng lớn dữ liệu. Vì vậy chúng hoàn hảo để phân tích dữ liệu.

4. Mạng và Internet

MẠNG[3] là một tổ hợp các thiết bị được kết nối với nhau (ví dụ như máy tính, máy tính xách tay, và máy chủ).

[1] Data.
[2] Analysis.
[3] Network.

Chúng chia sẻ thông tin, và có thể dùng chung các thiết bị như máy in. **INTERNET** là mạng toàn cầu kết nối hàng triệu máy tính. Một phần của việc nghiên cứu mạng và internet nhằm tìm ra cách tốt hơn và an toàn hơn để chia sẻ thông tin giữa các thiết bị.

5. Ảnh hưởng của máy tính

Để hiểu được sự ảnh hưởng của máy tính, chúng ta cần nghiên cứu cách máy tính tác động lên cuộc sống, văn hóa, sự an toàn, luật pháp và hành vi. Chúng ta có thể tạo ra công nghệ mới không có nghĩa là chúng ta *nên* làm vậy.

Ví dụ, chuyện gì sẽ xảy ra nếu bạn tạo ra một chương trình có thể xâm nhập vào điện thoại của mọi người và xóa bức ảnh yêu thích của họ?

Những câu hỏi cần suy ngẫm:

- Chuyện đó sẽ ảnh hưởng tới cuộc sống người bị hại như thế nào?
- Hành động của bạn sẽ ảnh hưởng tới tình bạn như thế nào?
- Luật pháp có cấm sử dụng chương trình đó không?
- Có sai không nếu chúng ta chỉ chia sẻ chương trình chứ không đích thân sử dụng?
- Công nghệ này sẽ khiến cuộc sống mọi người tốt hơn hay tệ hơn?

KIỂM TRA KIẾN THỨC

1. Khoa học máy tính là gì?

2. Đồ vật nào sau đây KHÔNG thuộc hệ thống máy tính?

A. Điện thoại

B. TV mới

C. Sách

D. Sách điện tử

3. Chương trình là gì? Và tại sao máy tính cần chương trình?

4. Khoa học máy tính KHÔNG PHẢI là:

A. Nghiên cứu những vấn đề thú vị, chẳng hạn như cách sử dụng thuật toán để khiến việc giao hàng nhanh hơn.

B. Xây một hệ thống máy tính để giải quyết vấn đề.

C. Chơi điện tử.

D. Hiểu rằng một công nghệ mới có thể ảnh hưởng xấu tới con người.

5. Hệ thống máy tính là gì? Đưa ví dụ.

6. Các nhà khoa học máy tính đã tạo ra hệ thống giúp người nông dân sản xuất nhiều thức ăn hơn cho xã hội. Sự cải tiến này thuộc về lĩnh vực nào trong ngành khoa học máy tính?

7. Sự khác biệt giữa người tiêu dùng và người sáng tạo?

8. Một nhóm các thiết bị máy tính được kết nối gọi là gì?

9. Đánh giá tổng hợp điểm kiểm tra của học sinh để biết được trình độ của lớp học. Đây là hành động thuộc lĩnh vực nào trong ngành khoa học máy tính?

10. Giải thích thuật toán là gì?

ĐÁP ÁN

1. Khoa học máy tính là ngành sử dụng máy tính để giải quyết vấn đề.
2. C
3. Chương trình là tập các chỉ thị được mã hóa (code) thành ngôn ngữ mà máy tính có thể hiểu được. Không có chương trình, máy tính sẽ không hiểu chúng ta muốn nó làm gì.
4. C
5. Hệ thống máy tính là một hoặc nhiều thiết bị điện tử. Bao gồm máy tính, máy tính bảng, và những thiết bị khác được chạy bởi chương trình.
6. Ảnh hưởng của máy tính. Những lĩnh vực khác trong ngành khoa học máy tính được ứng dụng để phát triển công cụ tốt hơn cho nông dân, tuy nhiên ví dụ chỉ ra cách công cụ giúp ích cho xã hội - một ví dụ cho thấy ảnh hưởng của máy tính.
7. Người tiêu dùng chỉ sử dụng chương trình và sản phẩm. Người sáng tạo làm ra những sản phẩm người tiêu dùng sử dụng.
8. Mạng hoặc mạng máy tính.
9. Dữ liệu và phân tích.
10. Thuật toán là các chỉ dẫn cách làm điều gì đó.

Chương 2

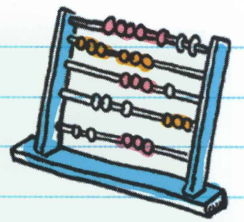
MÁY TÍNH LÀ GÌ?

MÁY TÍNH lưu trữ, chia sẻ và phân tích lượng lớn dữ liệu. Nó cũng có thể xử lý các phép toán phức tạp.

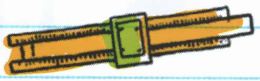
> Dù CD (đĩa nén) có thể lưu trữ thông tin như âm nhạc, nó không được coi là máy tính vì bản thân CD không thay đổi được thông tin.

MÁY TÍNH QUA CÁC THỜI KỲ

Hàng ngàn năm trước, con người tạo ra các thiết bị như **BÀN TÍNH** để tính các phép toán phức tạp.

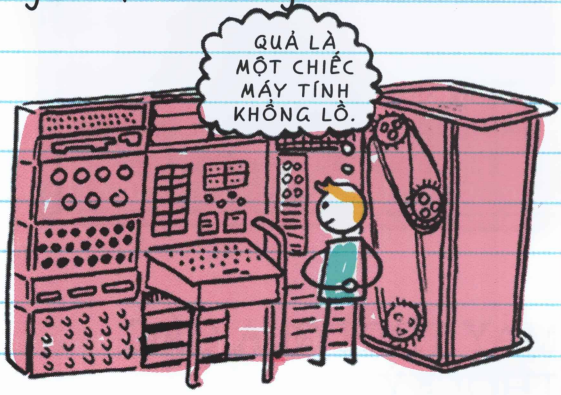


Vào những năm 1600, tính toán trở thành một nghề và từ ngữ "máy tính" nghĩa là "người thực hiện phép tính". Những máy tính (con người) sử dụng một công cụ gọi là **THƯỚC KẺ TRƯỢT** *(Slide Rule)* để tính tốt hơn.



Vào năm 1944, giữa Thế Chiến II, chiếc máy tính điện tử đầu tiên được chế tạo. Nó tên là COLOSSUS MARK 1 và được chính phủ Anh sử dụng để giải mật mã của quân Đức.

Năm 1945, người Mỹ chế tạo chiếc máy tính nhanh hơn có tên là Eniac.



ENIAC

Viết tắt của "Electronic Numerical Integrator and Computer". Đây là chiếc máy tính đa năng, có thể lập trình được đầu tiên trên thế giới.

Những chiếc máy tính đời đầu rất khổng lồ – chúng chiếm cả căn phòng! Những chiếc máy tính này được sử dụng bởi các doanh nghiệp, chính phủ, hoặc nhà nghiên cứu.

Chiếc máy tính cá nhân được sản xuất hàng loạt đầu tiên, APPLE II, ra đời năm 1977. Người dùng có thể sử dụng máy tính để chạy chương trình đơn giản hoặc chơi điện tử.

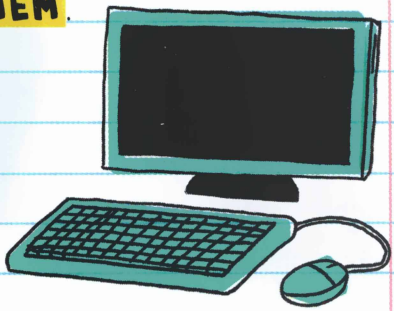
Ngày nay, máy tính nhỏ xíu và ở mọi nơi, bao gồm điện thoại. Điện thoại thông minh được coi là máy tính bởi vì:

- Chúng chạy chương trình (như nhắn tin, chỉnh sửa ảnh, và bản đồ).
- Chúng lưu trữ thông tin (như ảnh, thông tin liên lạc, và tài liệu).
- Chúng xử lý các phép tính (bằng máy tính, đồng hồ/đồng hồ bấm giờ, bản đồ).

Điện thoại thông minh có tính năng cao hơn máy tính được sử dụng trên chiếc tên lửa đưa các phi hành gia lên mặt trăng vào năm 1969.

CÁC PHẦN CỦA TỔNG THỂ

Máy tính được tạo nên bởi 2 phần: **PHẦN CỨNG** và **PHẦN MỀM**.



Phần cứng

Phần cứng là các phần vật chất của máy tính, như bàn phím, chuột và màn hình. Có một cách đơn giản để xác định phần nào của máy tính là phần cứng: Nếu bạn nhìn thấy nó bằng mắt, đó là phần cứng.

Phần mềm

Phần mềm là những chương trình (chỉ dẫn) cho máy tính biết phải làm gì, ví dụ như ứng dụng điện thoại, ứng dụng chỉnh sửa, và dịch vụ giải trí. Có nhiều loại phần mềm: từ trò chơi điện tử tới phần mềm chạy lò vi sóng, và cả trình duyệt web trên điện thoại. Phần mềm được lưu ở các thiết bị lưu trữ (phần cứng) như ổ đĩa cứng, ổ USB flash, và CD.

Ứng dụng (app) là một chương trình được thiết kế để thực hiện một nhiệm vụ – ví dụ như chụp ảnh.

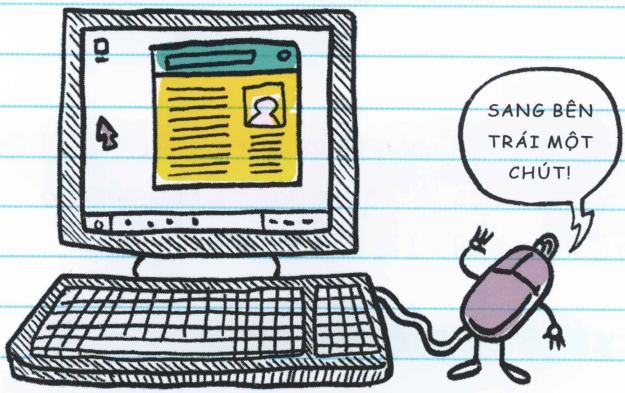
Tất cả ứng dụng đều là chương trình, nhưng không phải tất cả chương trình là ứng dụng. Một số chương trình chỉ cho máy tính sử dụng – không phải cho người dùng sử dụng.

TÌM HIỂU KỸ HƠN VỀ PHẦN CỨNG

Phần cứng có thể được chia thành các loại:

Đầu vào

Đầu vào chỉ những phần nạp thông tin vào máy tính, ví dụ như bàn phím, chuột, điều khiển trò chơi điện tử, và webcam.

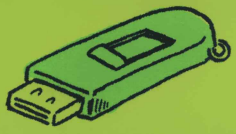


Ổ lưu trữ

Ổ lưu trữ chỉ những phần lưu trữ thông tin cho máy tính, bao gồm ổ đĩa cứng, **Ổ USB**, CD và **RAM**.

Ổ USB (UNIVERSAL SERIAL BUS)

Còn được gọi là thumb drive – ổ đĩa ngón tay cái, hoặc USB *stick* – thanh USB; một thiết bị di động có thể cắm vào, dùng để lưu trữ thông tin.



RAM (RANDOM ACCESS MEMORY)

Một dạng bộ nhớ máy tính – nơi thông tin được lưu trữ.

Bộ xử lý

Bộ xử lý là phần chạy chương trình, diễn giải thông tin đầu vào, và đưa thông tin đầu ra, bao gồm bộ xử lý trung tâm (CPU) và bộ xử lý đồ họa (GPU).

BỘ XỬ LÝ TRUNG TÂM (CPU)

Đây là bộ não của máy tính. Nó nhận thông tin đầu vào và thực hiện mệnh lệnh từ phần cứng và phần mềm khác. Nó bao gồm bộ nhớ chính, bộ điều khiển và bộ logic số học.

BỘ XỬ LÝ ĐỒ HỌA (GPU)

Đây là phần máy tính tập trung vào đồ họa. Đồ họa là hình ảnh ở mọi loại tệp (ảnh chụp, hoạt ảnh và video). GPU đưa hình ảnh, hoạt ảnh và video có thể xem được lên trên màn hình máy tính. Nếu bạn muốn trò chơi điện tử yêu thích có chất lượng hình ảnh tốt nhất, bạn cần GPU xịn để có được đồ họa đẹp.

Hãy coi CPU là thợ đa nghề và GPU là thợ lành nghề. CPU phù hợp để hoàn thành tốt những công việc đa dạng hàng ngày; GPU tập trung vào xử lý nhanh chóng thông tin hiển thị. Bạn sẽ không muốn thợ đa nghề thay hết ống nước trong nhà. Bạn sẽ muốn một người thợ lành nghề với kỹ năng chuyên nghiệp xử lý công việc đó.

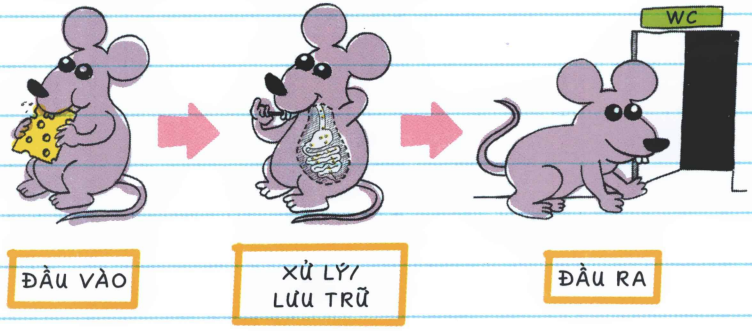
Đầu ra

Đầu ra chỉ phần máy tính dùng để gửi thông tin tới bạn. Nó bao gồm máy chiếu, màn hình, máy in, loa, và tai nghe.

Màn hình cảm ứng là đầu vào hay đầu ra? Nó là cả hai!

Các phần cứng làm việc cùng nhau. Dữ liệu (thông tin) từ người dùng (có thể là người hoặc máy tính) đi qua thiết bị đầu vào để xử lý. Dữ liệu có thể được lưu vào ổ nhớ (lưu trữ), hoặc xuất từ ổ nhớ và gửi tới người dùng.

Quá trình này giống như tiêu hóa: Chúng ta ăn, tiêu hóa và lưu trữ nó, sau đó thải nó ra ngoài.



TÌM HIỂU KỸ HƠN VỀ PHẦN MỀM

Phần mềm ứng dụng (App)

Phần mềm ứng dụng (app) là chương trình giúp người dùng hoàn thành công việc.

"App" là viết tắt của "application" (ứng dụng).

Bộ xử lý văn bản, trình duyệt internet, và trò chơi đều là ví dụ về phần mềm ứng dụng. Mọi ứng dụng trong cửa hàng ứng dụng trên điện thoại đều là phần mềm ứng dụng. Có nhiều dạng phần mềm ứng dụng, và lập trình viên có thể chuyên sâu vào một lĩnh vực.

VÍ DỤ:

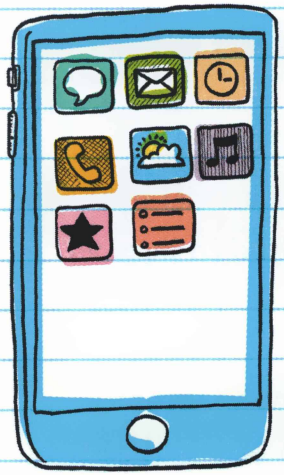
- **LẬP TRÌNH VIÊN WEB** là người viết phần mềm ứng dụng chạy trên web.
- **LẬP TRÌNH VIÊN DI ĐỘNG** là người viết phần mềm ứng dụng cho thiết bị di động như điện thoại và máy tính bảng.
- **LẬP TRÌNH VIÊN TRÒ CHƠI** là người viết phần mềm ứng dụng cho trò chơi điện tử.

Phần mềm hệ thống

Phần mềm hệ thống là chương trình đảm bảo các phần cứng độc lập hoạt động với nhau và hoạt động với các chương trình khác một cách trơn tru. **HỆ ĐIỀU HÀNH** như Window trên máy tính xách tay hay iOS (hệ điều hành điện thoại) trên iPhone là ví dụ của phần mềm hệ thống. Hệ điều hành chạy ở nền, bạn không thể thấy chúng hoạt động trên màn hình. Hệ điều hành cho phép máy cài đặt trò chơi, mạng xã hội, và những ứng dụng khác trên điện thoại.

Ứng dụng không thể hoạt động nếu thiếu phần mềm hệ thống.

HỆ ĐIỀU HÀNH
Chương trình tương tác với phần cứng, cho phép các chương trình khác chạy ở nền.



Phần cứng và phần mềm khá vô dụng nếu đứng độc lập. Chỉ khi hoạt động cùng nhau trong một hệ thống hoàn chỉnh, chúng mới trở thành công cụ.

Ví dụ:

PHẦN CỨNG		PHẦN MỀM		KẾT QUẢ CUỐI CÙNG
MÁY ẢNH	+	ỨNG DỤNG ẢNH	=	ĐIỆN THOẠI CHỤP HÌNH
BÀN PHÍM + MÁY IN	+	BỘ XỬ LÝ VĂN BẢN	=	IN THIỆP MỜI TIỆC
Ô TÔ + CẢM BIẾN + MÁY QUAY	+	CHƯƠNG TRÌNH ĐỊNH VỊ ĐƯỜNG	=	Ô TÔ TỰ LÁI

KIỂM TRA KIẾN THỨC

1. Sự khác biệt giữa phần mềm và phần cứng?

2. Máy tính là gì?

3. Máy tính đã thay đổi qua thời gian như thế nào?

4. Các vật sau đây là dạng phần cứng nào?
   Chọn từ: **Đầu vào, đầu ra, xử lý** hoặc **lưu trữ**
   A. Màn hình TV: _____
   B. CPU: _____
   C. Bàn phím: _____
   D. Tai nghe: _____
   E. USB: _____

5. Phần cứng lưu trữ và phần cứng xử lý khác nhau như thế nào? Có cần cả hai để máy tính hoạt động không?

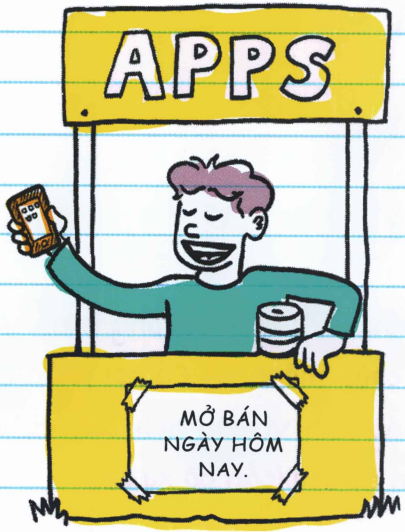
6. Sự khác biệt giữa CPU và GPU?

7. Tại sao Apple II đặc biệt?

8. Chiếc điện thoại thông minh đời mới giống chiếc máy tính đầu tiên được chế tạo như thế nào? Khác như thế nào?

9. Công cụ nào tồn tại trước khi máy tính có thể lập trình được sáng chế?

10. Sự khác biệt giữa phần mềm hệ thống và phần mềm ứng dụng?



ĐÁP ÁN

ĐÁP ÁN

1. Phần cứng là phần vật chất của máy tính, và phần mềm là code hoặc chương trình mà máy tính chạy.

2. Máy tính là thiết bị lưu trữ và xử lý thông tin.

3. Máy tính đã nhỏ hơn, nhiều tính năng hơn rất nhiều qua thời gian.

4. Phần cứng:
   A. Đầu ra
   B. Xử lý
   C. Đầu vào
   D. Đầu ra
   E. Lưu trữ

5. Bộ xử lý như CPU chạy chương trình, và phần lưu trữ như RAM chứa thông tin. Cả hai phần đều quan trọng đối với máy tính, vì máy tính là thiết bị lưu trữ và điều khiển dữ liệu (cần bộ xử lý).

6. CPU là bộ não chính của máy tính và chạy hầu hết chương trình. GPU chủ yếu xử lý thông tin hiển thị trên màn hình.

7. Apple II là máy tính cá nhân được sản xuất hàng loạt đầu tiên.

8. Giống như chiếc máy tính đầu tiên, điện thoại thông minh sử dụng cả phần mềm và phần cứng để chạy. Cả hai đều có đầu vào, đầu ra, bộ nhớ và bộ xử lý. Nhưng điện thoại thông minh mạnh mẽ và nhỏ hơn nhiều lần so với chiếc máy tính đầu tiên.

9. Trước khi máy tính lập trình được phát minh, con người sử dụng vài công cụ để giải toán, ví dụ như bàn tính và thước loga.

10. Phần mềm hệ thống đảm bảo các phần cứng độc lập hoạt động trơn tru với nhau và với các chương trình khác. Phần mềm ứng dụng là chương trình cho phép người dùng thực hiện những công việc cụ thể.

TÔI MÁY TÍNH

Chương 3

CHÚNG TA TƯƠNG TÁC VỚI MÁY TÍNH NHƯ THẾ NÀO?

Càng ngày máy tính càng dễ sử dụng. Tuy nhiên nó vẫn có thể có sai sót. Khi xảy ra vấn đề, có vài điều bạn có thể làm.

SỰ TƯƠNG TÁC NGƯỜI - MÁY TÍNH

GIAO DIỆN NGƯỜI DÙNG (UI) bao gồm tất cả các phần của hệ thống máy tính bạn dùng để vận hành máy tính. Ví dụ, giao diện người dùng khi bạn chơi trò chơi trên điện thoại bao gồm màn hình cảm ứng, loa/tai nghe, màn hình chính của trò chơi, nút bấm và đồ họa.

GIAO DIỆN ĐỒ HỌA NGƯỜI DÙNG (GUI, phát âm là "Goo-ey", nghe như một loại bánh brownie mềm) là loại giao diện người dùng mà sử dụng biểu tượng trên màn hình thay vì chỉ có văn bản. GUI giúp mọi người dùng máy tính dễ hơn. Hầu hết giao diện người dùng là GUI.

GIAO DIỆN DÒNG LỆNH (CLI) chỉ dùng văn bản để điều khiển máy tính. Loại giao diện này khó sử dụng hơn vì người dùng phải gõ chính xác lệnh, hoặc cụm từ để sử dụng. Hầu hết người dùng máy tính không bao giờ phải dùng giao diện dòng lệnh, tuy nhiên bạn nên học nếu bạn là một nhà khoa học máy tính.

Sự tương tác người - máy tính (HCI) là một phần khoa học máy tính, nghiên cứu và cải tiến cách người dùng sử dụng máy tính. Các UI đều dễ sử dụng cho mọi người, bao gồm cả những người mù, thị lực kém, điếc hoặc thính giác kém.

Bạn có thể trải nghiệm một ví dụ về sự tương tác người - máy tính bằng cách tìm phần cài đặt "Truy cập" trong điện thoại hoặc máy tính, bật tính năng đọc màn hình để nghe máy đọc to những thông tin hiển thị trên màn hình.

VÍ DỤ, nhiều điện thoại có cài đặt tính năng đọc màn hình. Điều này giúp người mù hoặc người có thị giác kém sử dụng điện thoại.

Tính năng đọc thành tiếng những chữ trên màn hình.

KHẮC PHỤC SỰ CỐ

Khắc phục sự cố là cách tiếp cận có hệ thống, hay từng bước, để xử lý lỗi sai trong hệ thống máy tính hoặc phần mềm hoặc sửa lỗi chương trình trong phần mềm.

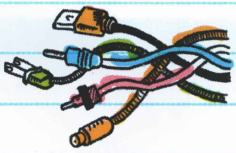
Cách tiếp cận có hệ thống nghĩa là tuân theo quy trình hoặc kế hoạch cố định, từng bước một. Ví dụ như hướng dẫn làm bánh mì. Nếu máy tính của bạn không khởi động, đầu tiên bạn cần đảm bảo nó đã được cắm điện, sau đó bạn cần kiểm tra xem các dây điện trong máy đã được kết nối cẩn thận hay chưa. Lợi thế của cách tiếp cận có hệ thống là bạn có thể lặp lại các bước một cách chính xác và chắc chắn bạn không bỏ sót thứ gì.

Sửa lỗi là tìm và sửa chữa *"bug"*[1] trong chương trình. Sửa lỗi là một kiểu khắc phục sự cố cụ thể, được dùng khi lập trình. Lỗi có thể do đánh máy hoặc do không tuân theo nguyên tắc hoặc hình thức của ngôn ngữ lập trình.

Troubleshooting

Người khắc phục sự cố sử dụng một số phương pháp sau để tìm kiếm và sửa chữa vấn đề:

- Đổi các bộ phận hoạt động được từ máy tính khác sang, ví dụ như màn hình, bàn phím, thậm chí là ổ cứng, mạch chủ hoặc dây cắm điện.



- Tạo ra một biểu đồ để lần theo vấn đề. Ví dụ, tạo biểu đồ của tất cả các thành phần trong hệ thống máy tính và cách chúng kết nối với nhau. Bằng cách này, người khắc phục sự cố sẽ không bị rối khi lần theo vấn đề.

[1] Bug: lỗi trong các đoạn mã lập trình.

- Thay đổi phần mềm để xem phần cứng có hoạt động. Đôi khi phần mềm cập nhật bị lỗi và bạn cần cài lại phiên bản trước đó. Hoặc có lẽ bạn cần cài đặt bộ điều khiển mới (chương trình giúp máy tính biết cách tương tác với phần cứng).

- Kiểm tra cài đặt phần mềm và độ tương thích của các thành phần. Một số phần mềm được chế tạo cho phần cứng đặc biệt. Ví dụ, bạn không thể cài đặt ứng dụng iPhone trên điện thoại Android.

- Hỏi sự giúp đỡ từ người khắc phục sự cố có kinh nghiệm hơn. Đôi khi vấn đề quá lớn để một người tự giải quyết. Có nhiều trang web hỗ trợ mọi người xử lý mọi vấn đề máy tính - đó là những nguồn thông tin đáng tìm kiếm.

KIỂM TRA KIẾN THỨC

1. Khắc phục sự cố là gì?

2. Dòng nào dưới đây KHÔNG PHẢI LÀ kỹ thuật khắc phục sự cố?

   A. Lập danh sách

   B. Nhờ giúp đỡ

   C. Thay máy tính

   D. Lần lượt đổi từng bộ phận máy tính

3. Cách tiếp cận có hệ thống là gì? Tại sao nó hữu ích?

4. Nếu máy tính không khởi động, bạn nên làm gì đầu tiên?

   A. Tháo dỡ nó

   B. Đảm bảo dây điện đã cắm vào ổ điện

   C. Coi nó bị hỏng và trả lại cửa hàng

   D. Ấn mạnh các nút

5. Sửa lỗi là gì?

6. Khắc phục sự cố khác gì với sửa lỗi?

7. Lĩnh vực nào trong khoa học máy tính nghiên cứu giao diện người dùng của máy tính?

8. Khi bạn chơi điện tử trên iPad, giao diện người dùng nào được sử dụng?

9. Điều gì tạo nên một giao diện người dùng tốt?

ĐÁP ÁN

1. Giải quyết vấn đề bằng cách tiếp cận có hệ thống, từng bước một.

2. C

3. Cách tiếp cận có hệ thống là quy trình từng bước, có thể lặp lại và cho kết quả tương tự.

4. B

5. Sửa lỗi là tìm và sửa lỗi sai trong chương trình.

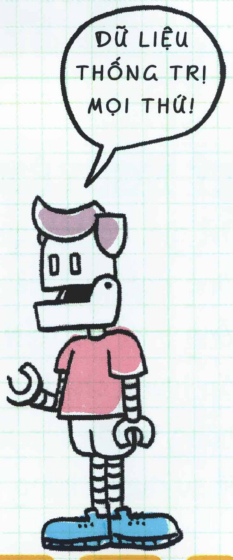
6. Khắc phục sự cố là quá trình sửa chữa vấn đề nói chung, nhưng sửa lỗi chỉ nói đến tìm kiếm và sửa chữa lỗi trong chương trình.

7. Sự tương tác người - máy tính.

8. Giao diện đồ họa người dùng (GUI).

9. Một giao diện người dùng tốt là giao diện mọi người đều dễ dàng sử dụng.

BÀI 2



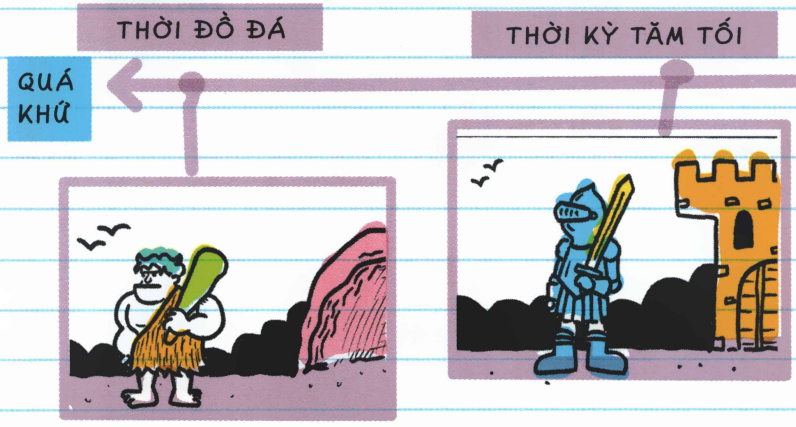
Dữ liệu và phân tích

Chương 4

LƯU TRỮ THÔNG TIN

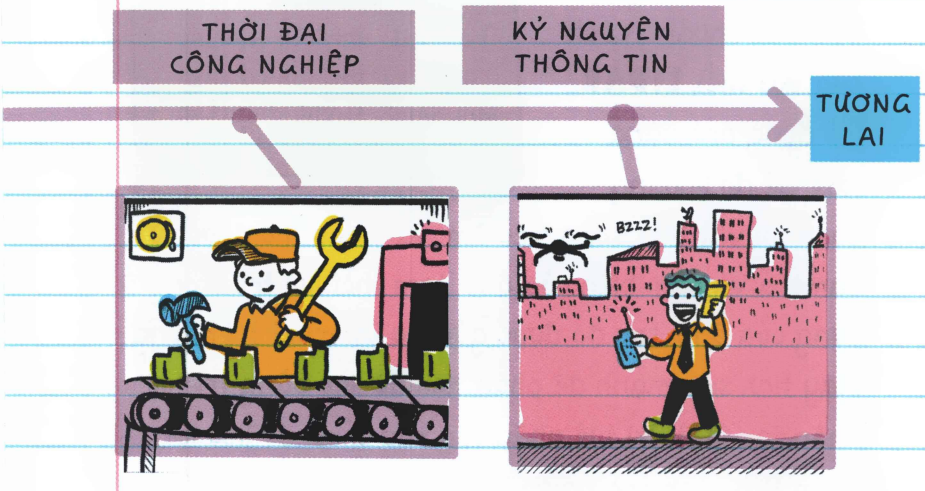
KỶ NGUYÊN THÔNG TIN

Thế giới đã trải qua Thời Đồ Đá, Thời Kỳ Tăm Tối, Thời đại Công nghiệp. Nhiều người cho rằng chúng ta hiện đang sống ở **KỶ NGUYÊN THÔNG TIN**. Vì chúng ta đã chuyển từ tập trung chế tạo đồ thủ công sang sử dụng công nghệ thông tin để sản xuất. Chúng ta đã tìm ra những cách mới để sử dụng thông tin vào mục đích chế tạo.



Máy tính là **CÔNG CỤ XỬ LÝ THÔNG TIN**. Máy tính nạp thông tin và xử lý chúng để chúng ta dễ sử dụng hơn. Ví dụ:

- **Ứng dụng bản đồ** xử lý thông tin từ vệ tinh để giúp chúng ta tìm ra đường nhanh nhất tới đích, thậm chí là tránh đường tắc.
- **Ứng dụng mạng xã hội** (ví dụ như Snapchat, Twitter, Instagram, hoặc Facebook) xử lý thông tin cá nhân (lượt thích, bài đăng,...), nhằm tạo ra hồ sơ người dùng. Các doanh nghiệp có thể sử dụng thông tin đó để bán hàng cho họ.



Lưu ý: Bạn không trả tiền để sử dụng mạng xã hội, không có nghĩa là nó miễn phí. Cái giá của việc sử dụng chúng là bạn đưa ra lượng lớn thông tin về bản thân. Đó là lý do tại sao bạn luôn cần xác nhận của người bảo hộ và đọc hướng dẫn đầy đủ trước khi cung cấp BẤT CỨ thông tin cá nhân nào trên mạng.

- Ô tô tự lái thu thập và phân tích lượng lớn thông tin để đưa ra dự đoán và quyết định. **RADAR** (sử dụng sóng vô tuyến để xác định vị trí và khoảng cách của vật thể), **LIDAR** (sử dụng ánh sáng lase để đo khoảng cách tới vật thể), sóng siêu âm, và video thu được từ camera và cảm biến của ô tô, tạm thời lưu trữ trong ổ cứng của ô tô, sau đó được gửi qua internet tới những máy tính siêu mạnh để phân tích dữ liệu bằng trí tuệ nhân tạo (AI). Trí tuệ nhân tạo xâu chuỗi dữ liệu lại, phân tích cách con người lái xe an toàn để học hỏi.

RADAR
Viết tắt của "Đo khoảng cách và xác định bằng sóng vô tuyến" (radio detection and ranging).

LIDAR
Viết tắt của "Đo khoảng cách và xác định bằng ánh sáng" (light detection and ranging).

Chúng ta càng có nhiều thông tin và công cụ mạnh hữu ích hơn để giải mã thông tin, chúng ta càng đưa ra quyết định tốt hơn. Ví dụ, bạn mua quà sinh nhật cho bạn của bạn sẽ dễ hơn mua quà cho một đứa nhóc mới chuyển đến khu phố. Vì bạn có nhiều thông tin về bạn mình hơn, nên bạn có thể đưa ra quyết định tốt hơn về món quà họ sẽ thích.

MÁY TÍNH SỬ DỤNG DỮ LIỆU

DỮ LIỆU là tổng hợp của những số liệu, từ ngữ chưa được sắp xếp và chưa có ý nghĩa. Dữ liệu có nhiều dạng, phụ thuộc vào dạng thông tin nó chứa đựng.

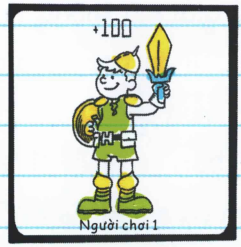
THÔNG TIN là dữ liệu đã được sắp xếp và có ý nghĩa. Dữ liệu được xem xét trong tình huống cụ thể.

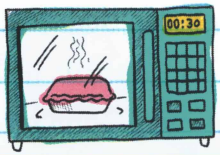
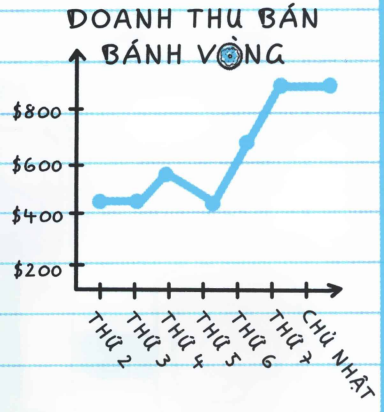
Ví dụ, tất cả những câu trả lời trong bài khảo sát là dữ liệu, và kết quả trung bình là thông tin.

Dữ liệu đầu vào là thông tin chưa sắp xếp được nạp vào máy tính.

Dữ liệu đầu ra là thông tin sau khi máy tính đã xử lý .

Quá trình xử lý của máy tính.

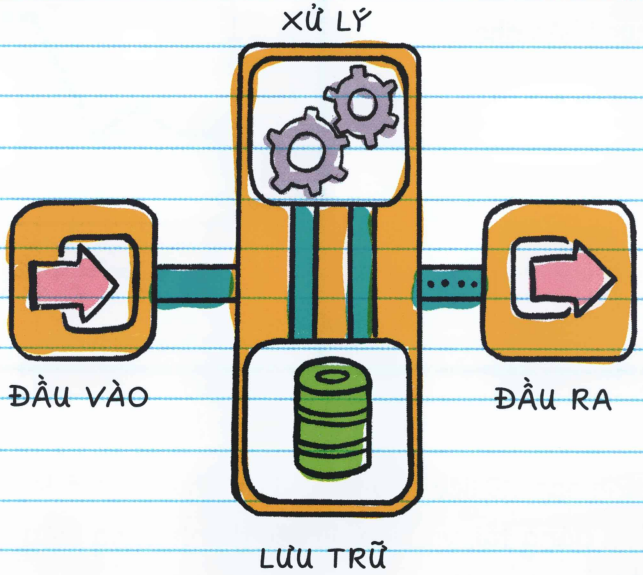
DỮ LIỆU ĐẦU VÀO	DỮ LIỆU ĐẦU RA
Đối với trò chơi điện tử: Nhấn nút trên điều khiển cầm tay.	Nhân vật di chuyển trên màn hình. 
Đối với mạng xã hội: Gõ bình luận hoặc đăng tải ảnh và video.	Video và ảnh đã qua xử lý đính kèm với trang cá nhân.

DỮ LIỆU ĐẦU VÀO	DỮ LIỆU ĐẦU RA
Đối với lò vi sóng: Chọn mức độ trên phím điều khiển.	Thức ăn được nấu trong một khoảng thời gian xác định. 
Đối với chương trình lập biểu đồ: Nhập số liệu và các mục khác nhau.	Bảng biểu với số liệu 

Máy tính nạp dữ liệu đầu vào, ví dụ như câu trả lời, số liệu, ảnh đăng tải, và biến dữ liệu thành dạng máy tính có thể hiểu và xử lý. Ví dụ, máy tính phân nhỏ ảnh thành hàng triệu mảnh, sau đó màu của mỗi mảnh được ghi lại

trên một đoạn code dài. Máy tính xử lý đoạn code và gửi lại dữ liệu đầu ra cho người dùng. Để hiển thị bức ảnh, máy tính đọc đoạn code dài và tái cấu trúc hình ảnh trên màn hình bằng cách hiển thị từng mảnh màu tương ứng, giống như một bức tranh siêu chi tiết được vẽ bằng số.

MÃ (CODE)
Hệ thống biểu tượng, chữ, và số được sử dụng để biểu thị thứ khác.



MÃ HÓA DỮ LIỆU

MÃ MORSE
Sử dụng gạch ngang và chấm để biểu thị chữ cái.

Dữ liệu cần được viết ở dạng máy tính hiểu được. Bạn cần mã hóa dữ liệu để máy tính xử lý. Khi bạn **MÃ HÓA** hóa thông tin, bạn thay đổi hình ảnh, video, từ ngữ... thành mã. Ví dụ, máy mật mã morse hồi Thế Chiến II sẽ mã hóa thông điệp thành gạch ngang và chấm, sau đó được gửi qua dây điện báo.

Một hệ thống liên lạc lỗi thời sử dụng tín hiệu điện tử.

Khi bạn **GIẢI MÃ** thông tin, bạn chuyển đổi mã thành dạng thông điệp có thể hiểu được. Ví dụ, sau khi nhận thông điệp được mã hóa ở dạng gạch ngang và chấm, máy mật mã morse sẽ giải mã chúng thành Tiếng Anh.

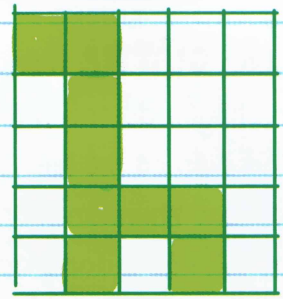
Máy mật mã morse.

Có một số **HỆ THỐNG MÃ HÓA DỮ LIỆU** mà chúng ta có thể dùng để giúp máy tính hiểu dữ liệu đầu vào. Hệ thống mã hóa dữ liệu là cách biểu thị mọi loại thông tin để máy tính có thể hiểu được. Máy tính chỉ có thể hiểu được dãy số 1 và 0.

Mọi mã sử dụng 1 và 0 được gọi là **MÃ NHỊ PHÂN** (chúng ta sẽ đề cập kỹ hơn ở phần sau). Hệ thống mã hóa ra đời để giúp máy tính hiểu, diễn giải, và tạo ra dữ liệu bằng cách phân nhỏ dữ liệu – hình ảnh, âm thanh, video, màu sắc, biểu tượng, và các dạng thông tin khác – thành số 1 và 0.

Bạn có thể bắt chước cách máy tính lưu trữ hình ảnh bằng cách sử dụng số 0 để biểu thị ô trắng và số 1 để biểu thị ô xanh. Dưới đây là ví dụ cho thấy cách dữ liệu được biểu thị để vẽ một chú hươu cao cổ đơn giản trên bảng 5x5.

DÃY 1:	1, 1, 0, 0, 0
DÃY 2:	0, 1, 0, 0, 0
DÃY 3:	0, 1, 0, 0, 0
DÃY 4:	0, 1, 1, 1, 0
DÃY 5:	0, 1, 0, 1, 0



Mã hóa màu sắc

Các dạng dữ liệu khác nhau, ví dụ như màu sắc, con số, và hình ảnh có thể được mã hóa theo những cách khác nhau. Cách phổ biến nhất để mã hóa màu sắc bằng hệ thống mã hóa là **RGB** và **HEXADECIMAL**.

Bạn có thể dùng máy scan để mã hóa một bức ảnh cũ thành dữ liệu kỹ thuật số – điều máy tính có thể hiểu. Máy scan sử dụng cảm biến để nhận biết màu sắc và mã hóa

cả bức ảnh thành một loạt các giá trị số. Máy scan sắp xếp tất cả những giá trị số này theo một thứ tự mà máy tính có thể đọc và dựng lại hình ảnh trên màn hình.

RGB là hệ màu sử dụng ba số được ngăn cách bởi dấu phẩy, mỗi số nằm ở khoảng giữa 0 và 255 (máy tính mã hóa từng giá trị thành số nhị phân). Mỗi số trong nhóm biểu thị một sắc thái khác nhau của **màu đỏ, màu xanh lục** và **xanh da trời** (Đó là lý do họ gọi nó là hệ màu RGB!). Khi các sắc thái khác nhau của màu đỏ, xanh lục và xanh da trời kết hợp lại, chúng sẽ tạo ra các màu khác.

Có 16 triệu cách kết hợp 3 số trong khoảng từ 0 đến 255. Nghĩa là có 16 triệu số có thể được biểu thị bằng hệ màu mã hóa RGB.

Hệ màu **hexadecimal** (viết tắt là "hex") sử dụng 6 ký tự để biểu thị 16 triệu màu, tương đương với hệ màu RGB. Hexadecimal khác RGB vì nó dùng chữ và số. Hệ màu sử dụng sự kết hợp giữa số từ 0-9 và chữ cái từ A-F. Ví dụ, #FF00B4 sẽ tạo thành màu hồng bằng cách dùng chữ cái F và B cùng số 0 và 4. Mỗi nhóm 2 ký tự (ví dụ như FF, 00, và B4) đại diện cho một màu khác: Hai ký tự đầu đại diện

cho màu đỏ, hai ký tự tiếp theo đại diện cho màu xanh lục, và hai ký tự cuối đại diện cho màu xanh da trời - giống như hệ màu RGB, sự kết hợp những chữ cái/số khác nhau tạo nên sắc thái của mỗi màu riêng.

Ngôn ngữ máy tính CSS dùng ký hiệu này để biểu diễn hệ màu hexadecimal.

Mặc dù dữ liệu như màu sắc có thể được thể hiện theo nhiều cách khác nhau, nhưng phần quan trọng là mỗi màu sử dụng một sơ đồ mã hóa để máy tính có thể hiểu được thông tin.

Biểu diễn màu sắc

Trong bảng dưới đây, hai cột đầu là chương trình (cách) máy tính đọc màu sắc. Mỗi hệ thống mã hóa được dịch sang mã nhị phân. Ba cột còn lại là ví dụ về cách con người đọc và nhận diện màu sắc.

MÁY TÍNH		CON NGƯỜI		
MÃ HEX	RGB	TIẾNG ANH	TIẾNG TÂY BAN NHA	HÌNH ẢNH
FF0000	255,0,0	Red	Rojo	
0000FF	0,0,255	Blue	Azul	
FFFF00	255,255,0	Yellow	Amarillo	
008000	0,128,0	Green	Verde	
000000	0,0,0	Black	Negro	

Mã hóa màu sắc thành số và chữ cái chỉ là một trong những cách xử lý thông tin để máy tính đọc được. Âm thanh, bàn phím, màn hình cảm ứng, dấu vân tay, địa điểm GPS, và mọi dạng dữ liệu đều được thu thập và mã hóa thành những dạng mà máy tính có thể hiểu.

KIỂM TRA KIẾN THỨC

Đối với câu hỏi 1-3, điền vào chỗ trống.

1. Chúng ta hiện đang sống ở Kỷ nguyên _____.

2. Hành động chuyển đổi dữ liệu thành mã được gọi là _____.

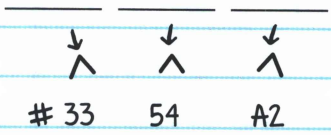
3. Hành động chuyển đổi dữ liệu từ mã sang dạng đọc được gọi là _____.

4. Sự khác biệt giữa dữ liệu và thông tin?

5. Những ví dụ sau là mã hóa hay giải mã?
   A. Xanh → 008000
   B. 000000 → Đen
   C. Amarillo → 255,255,0
   D. FF0000 →

6. Hệ màu hexadecimal hoặc RGB có thể tạo nên bao nhiêu màu?

7. Mỗi cặp ký tự dưới đây đại diện cho màu sắc nào trong hệ màu hexadecimal?



8. Mã hóa dữ liệu là gì? Tại sao máy tính cần nó?

9. Các công ty mạng xã hội sử dụng thông tin bạn chia sẻ để kiếm tiền như thế nào?

10. Nêu tên 1 hệ thống mã hóa dữ liệu.

ĐÁP ÁN

1. Thông tin

2. Mã hóa

3. Giải mã

4. Dữ liệu là nguồn đầu vào thô chưa xử lý, và thông tin là dữ liệu đã được xử lý thành nguồn đầu ra có thể sử dụng.

5. **A.** Mã hóa
   B. Giải mã
   C. Mã hóa
   D. Giải mã

6. Khoảng 16 triệu

7. Đỏ Lục Xanh da trời
   ↓ ↓ ↓
   ^ ^ ^
   # 33 54 A2

8. Mã hóa dữ liệu là chuyển đổi dữ liệu thành dạng máy tính có thể hiểu. Máy tính chỉ có thể hiểu dữ liệu được mã hóa sang hệ nhị phân. Vì vậy hình ảnh, âm thanh, tin nhắn, và những dạng dữ liệu khác phải được mã hóa để máy tính đọc được.

9. Mọi bài đăng, hình ảnh, video, hoặc mọi mảnh thông tin nhỏ bạn chia sẻ trên mạng đều cho doanh nghiệp gợi ý về bạn và về thứ bạn thích mua. Công ty có thể bán thông tin này cho doanh nghiệp quảng cáo.

10. Có nhiều hệ thống mã hóa dữ liệu. Câu trả lời đúng là những cách biểu thị thông tin dưới dạng máy tính có thể hiểu. Ví dụ: Hệ nhị phân, hexadecimal, RGB, hoặc tương tự.

ĐỎ
FFOOOO

Chương 5

THU THẬP VÀ SỬ DỤNG THÔNG TIN

Có nhiều cách để thu thập thông tin. Bạn có thể:

- Tiến hành phỏng vấn.
- Phát tờ khảo sát.
- Quan sát.

HIỆU QUẢ
Mất ít thời gian và sức lực hơn.

Máy tính có thể giúp các phương pháp trên **HIỆU QUẢ** hơn. Một bài khảo sát có thể là một loại cảm biến nạp thông tin từ con người. Máy tính có thể thu thập và phân tích dữ liệu nhanh và chính xác hơn con người.

PHỎNG VẤN

PHỎNG VẤN là cách thu thập thông tin thông qua đặt câu hỏi. Đây là cách tuyệt vời để lấy thông tin chi tiết từ mọi người. Khi phỏng vấn, chúng ta có thể mở rộng câu hỏi để có thêm thông tin và hiểu hơn về người được phỏng vấn. Phỏng vấn rất hiệu quả nếu bạn chỉ cần thu thập dữ liệu từ một nhóm nhỏ.

Mặt hạn chế là dữ liệu lấy được không dễ **TÍNH TOÁN**. Điều này nghĩa là chúng ta không thể sử dụng công thức toán học hoặc thuật toán máy tính để tìm khuôn mẫu trong dữ liệu. Ví dụ, ta không thể tìm được trung bình của câu trả lời dài một trang giấy. Nhưng ta có thể tìm được trung bình của những con số.

KHẢO SÁT

TỜ KHẢO SÁT là một danh sách câu hỏi. Khảo sát có thể được hoàn thành và gửi qua thư điện tử, trang web, hoặc các chương trình khác. Khảo sát cũng có thể được thực hiện trên giấy. Phản hồi của cuộc khảo sát sẽ cho ta thông tin chung về một nhóm lớn.

Máy tính sử dụng thông tin trong cuộc khảo sát để so sánh lượng lớn thông tin một cách hiệu quả.

Khảo sát hữu ích vì ta có thể gửi bảng khảo sát và mọi người sẽ tự hoàn thiện chúng. Khảo sát được sử dụng vì nó tiết kiệm thời gian.

Hạn chế của khảo sát là mọi người có thể sẽ không nghiêm túc với nó, hoặc họ sẽ không hoàn thành nó. Một hạn chế khác là phương pháp này khó thu thập thông tin chi tiết hơn, vì câu hỏi mở rộng không thể cá nhân hóa.

Các loại câu hỏi khảo sát khác nhau cung cấp các loại dữ liệu khác nhau:

Câu hỏi mở gợi ra câu trả lời chi tiết hơn, nhưng giống như phương pháp phỏng vấn, máy tính sẽ khó phân tích dữ liệu hơn.

Câu hỏi đa lựa chọn rất phù hợp để nhìn ra xu hướng và khuôn mẫu. Phiếu bầu cử là câu hỏi đa lựa chọn, các cử tri chọn một ứng viên cho mỗi vị trí. Điều này giúp máy tính phân tích dễ dàng hơn: Ứng viên có số phiếu nhiều nhất sẽ chiến thắng! Điểm trừ của câu hỏi đa lựa chọn là các câu trả lời bị hạn chế, và chúng ta sẽ không nhận được câu trả lời chi tiết như câu hỏi mở.

CÂU HỎI MỨC ĐỘ cho phép người trả lời đánh giá mức độ câu trả lời. Ví dụ, người thử vị kem đánh giá họ thích vị kem mới tới mức nào trên thang đo từ 1 đến 5, 1 có nghĩa là không thích và 5 nghĩa là rất thích.

Máy tính sử dụng câu hỏi đa lựa chọn và câu hỏi mức độ để so sánh dữ liệu hiệu quả hơn. Máy tính có thể sử dụng dữ liệu được cung cấp để tính toán và xác định xu hướng.

QUAN SÁT

QUAN SÁT đòi hỏi nhìn vào sự kiện và ghi chép lại.

Dữ liệu ở các trận đấu thể thao được thu thập bằng cách quan sát. Điểm, lỗi, phạt, thời gian, và khoảng cách được ghi lại để biết người chơi đã thắng ở mức nào.

Máy tính có thể so sánh và dự đoán dựa trên quan sát.

SỬ DỤNG DỮ LIỆU

Máy tính đầu tiên được chế tạo để tổng hợp thông tin **ĐIỀU TRA DÂN SỐ**. Dân số ở Mỹ đã lớn tới mức chính phủ cần một cách để tổng hợp tất cả dữ liệu khảo sát. HOLLERITH TABULATING MACHINE (Máy Lập bảng Hollerith) là chiếc máy được dùng để tổng hợp kết quả khảo sát, từ đó cung cấp thông tin hữu ích cho chính phủ. Máy tính vẫn luôn giúp con người hiểu được lượng lớn dữ liệu.

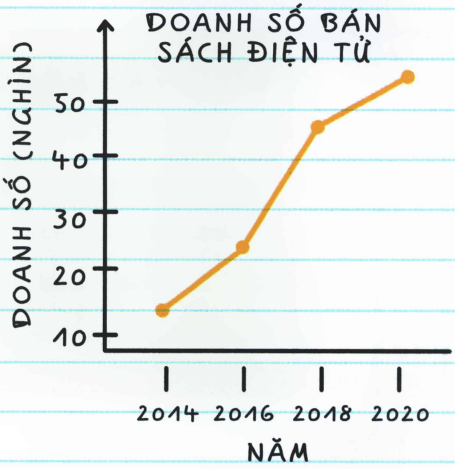
ĐIỀU TRA DÂN SỐ
Một cuộc khảo sát hoặc tính toán chính thức nhằm mục đích thu thập thông tin về dân số (ví dụ, nơi sống, tuổi tác, số người trong một hộ gia đình).

1890 – Hollerith Tabulating Machine
Một bước lớn tới máy tính hiện đại. Chiếc máy này đọc và tóm tắt thông tin lưu trên thẻ đục lỗ.

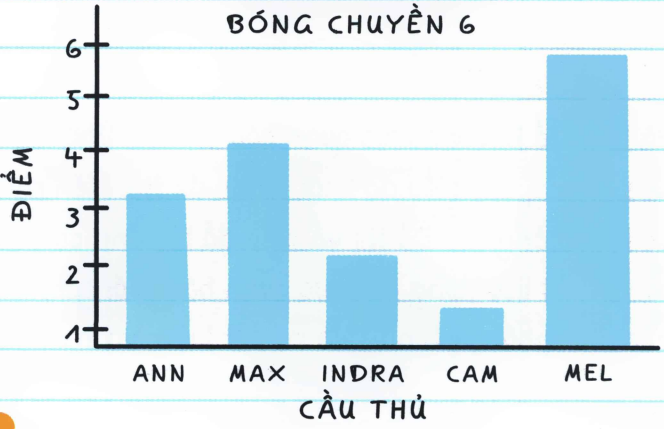
Đồ thị và Biểu đồ

Đồ thị và biểu đồ là cách trực quan hóa dữ liệu. Ứng dụng máy tính sử dụng các biểu đồ khác nhau để thể hiện rõ ràng các loại thông tin. Đồ thị và biểu đồ là công cụ các nhà khoa học dữ liệu dùng để trực quan hóa thông tin nhằm mục đích phân tích nó, đưa ra kết luận, và trao đổi thông tin.

Biểu đồ đường dùng để biểu thị sự thay đổi qua thời gian của thứ gì đó. Ví dụ, một công ty nên dùng biểu đồ đường để thể hiện sự tăng trưởng của doanh số bán sách điện tử.

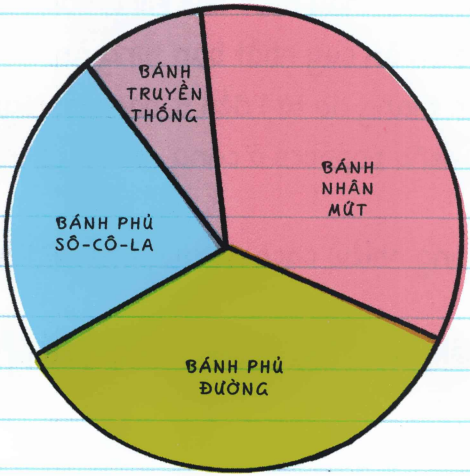


Biểu đồ cột dùng để so sánh các giá trị khác nhau trong cùng một hạng mục. Ví dụ, ta có thể so sánh số điểm được ghi bởi mỗi cầu thủ trong đội bóng.



Biểu đồ tròn dùng để biểu thị phần trăm của các phần. Ví dụ, ta có thể dùng biểu đồ tròn để thể hiện mức độ được yêu thích của các loại bánh vòng trong cửa hàng.

DOANH SỐ BÁN BÁNH VÒNG



KIỂM TRA KIẾN THỨC

1. Những câu hỏi nào sau đây sẽ thu thập được dữ liệu dễ tổng hợp? (Có thể chọn nhiều đáp án)
   A. Bạn thích ăn đồ cay không?
   B. Bạn có bị chảy nước mũi khi ăn đồ cay không?
   C. Vẽ món ăn cay nhất bạn từng ăn.
   D. Trên thang đo từ 1 đến 5, bạn đánh giá món jalapeño poppers ở căng-tin cay đến mức nào?

2. Hai (trong nhiều) cách để thu thập dữ liệu là _____.

3. Một điểm cộng của phương pháp phỏng vấn so với khảo sát?

4. Nếu bạn muốn biết trong 100 đứa trẻ xung quanh bạn, đội bóng nào được chúng yêu thích nhất, bạn sẽ dùng công cụ thu thập dữ liệu nào?

5. Nếu bạn muốn thu thập dữ liệu dễ tổng hợp, bạn nên tránh hai dạng câu hỏi nào?

6. Đưa ra một ví dụ về máy tính được sử dụng để thu thập dữ liệu.

7. Bạn của bạn đang tranh cử chức lớp trưởng. Họ muốn thu thập thông tin từ các học sinh khác một cách nhanh chóng và dễ dàng, để biết được học sinh muốn lớp trưởng mới thay đổi điều gì? Bạn sẽ gợi ý phương pháp thu thập dữ liệu nào và tại sao?

8. Bạn dùng đồ thị hoặc biểu đồ nào để thể hiện mức độ tăng trưởng của cây hàng tháng trong khoảng thời gian 2 năm?

9. Bạn dùng đồ thị hoặc biểu đồ nào để thể hiện mật độ chó và mèo trong trạm cứu hộ?

ĐÁP ÁN

ĐÁP ÁN

1. A, B, D
2. Tiến hành phỏng vấn, phát tờ khảo sát, hoặc quan sát.
3. Tiến hành phỏng vấn cho bạn nhiều quyền kiểm soát đối với quá trình. Bạn có thể tương tác với đối tượng và điều hướng tới câu trả lời liên quan hơn nếu bạn thấy mình không nhận được dữ liệu mong muốn.
4. Phỏng vấn 100 người sẽ mất rất nhiều thời gian. Vì vậy, lựa chọn tốt hơn là một khảo sát đơn giản.
5. Ta nên tránh câu hỏi mở và câu hỏi phỏng vấn.
6. Máy tính được dùng để thu thập mọi loại dữ liệu. Ví dụ, dữ liệu về người sử dụng mạng xã hội, bản đồ để ô tô định vị, bầu cử, và nhiều hơn.
7. Để thu thập dữ liệu từ nhiều người, khảo sát là cách dễ dàng nhất. Câu trả lời cho khảo sát là dữ liệu dễ tổng hợp hơn.
8. Biểu đồ đường dùng để thể hiện xu hướng qua thời gian.
9. Biểu đồ tròn dùng để thể hiện phần trăm các hạng mục trên tổng số.

BÀI 3

Kỹ thuật phần mềm

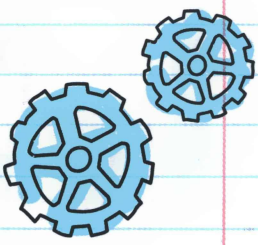
Chương 6

THIẾT KẾ HỆ THỐNG MÁY TÍNH

QUY TRÌNH THIẾT KẾ KỸ THUẬT

Khi thiết kế máy tính hoặc chương trình, các nhà khoa học máy tính làm theo **QUY TRÌNH THIẾT KẾ KỸ THUẬT** để hiểu và cải tiến thiết kế của họ. Họ muốn tạo ra chương trình hiệu quả và thân thiện với người dùng nhất có thể.

KỸ THUẬT là một nhánh của khoa học tập trung vào nghiên cứu thiết kế, và công dụng của máy móc và kiến trúc để giải quyết vấn đề. **Kỹ thuật máy tính** là ứng dụng nghiên cứu khoa học máy tính và kỹ thuật thực tiễn vào phần cứng và phần mềm. **Kỹ thuật phần mềm** là ứng dụng của khoa học máy tính.



Có nhiều **quy trình thiết kế kỹ thuật** hơi khác nhau, nhưng quy trình cơ bản nhất là:

1. Xác định vấn đề
2. Đề ra giải pháp
3. Lập kế hoạch
4. Thử nghiệm
5. Cải thiện

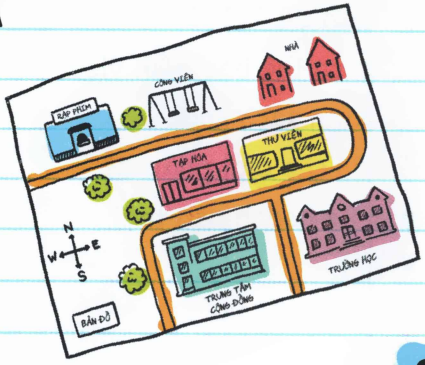
Và lặp lại quy trình.

Bước 1: Xác định vấn đề

Kỹ sư phần mềm xác định vấn đề và điều họ cần để đưa ra giải pháp. Họ tự hỏi:

- Vấn đề tôi muốn giải quyết là gì?

  - Ví dụ: Tìm đường ngắn nhất từ trường về nhà mà đi qua thư viện và tiệm tạp hóa.



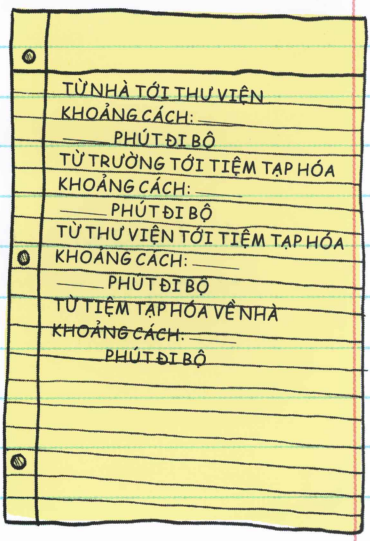
- Tôi có thể dùng gì để hỗ trợ giải quyết vấn đề?
  - Bản đồ cho thấy các địa điểm.
  - Khoảng cách giữa các địa điểm.
  - Loại phương tiện được sử dụng.

- Tôi cần kỹ năng gì để giải quyết vấn đề?
  - Biết tính toán.
  - Biết vẽ bản đồ.

Bước này có thể bao gồm việc thu thập và phân tích dữ liệu từ những người có chung vấn đề. Bạn có thể phỏng vấn, gửi khảo sát, hoặc quan sát mọi người. Bạn cũng có thể dùng cảm ứng hoặc theo dõi GPS.

Bước 2: Lập kế hoạch

Kỹ sư phần mềm đưa ra kế hoạch cho giải pháp của vấn đề. Việc lên kế hoạch có thể bao gồm việc lập danh sách hoặc vẽ bản đồ cho giải pháp đề xuất. Kế hoạch nên thật chi tiết.



Bước 3: Thực hiện

Bạn đã viết code và thêm các phần của giải pháp vào chương trình. Bước thực hiện kết thúc khi sản phẩm hoàn thành.

Bước 4: Thử nghiệm

Xác định vấn đề bằng cách thử nghiệm code nguồn hoặc giải pháp. Kỹ sư phần mềm thường nhờ người khác dùng thử sản phẩm và phản hồi. Có nhiều cấp độ thử nghiệm khác nhau. Ở giai đoạn đầu, lập trình viên sẽ tự thử nghiệm, sau đó họ sẽ nhờ đồng nghiệp thử nghiệm sản phẩm của mình. Sau đó lập trình viên sẽ nhờ một nhóm người dùng thử nghiệm sản phẩm và phản hồi.

Bước 5: Cải thiện

Sửa vấn đề và cải thiện sản phẩm. Ở bước này, các lỗi hoặc vấn đề xác định được ở bước thử nghiệm đều đã được sửa chữa.

KỸ THUẬT MỖI NGÀY

Những công nghệ tân tiến ngày càng trở nên nhỏ hơn, rẻ hơn, và dễ sử dụng hơn. Điều này nghĩa là ai cũng có thể thiết kế và tạo ra những thứ tuyệt vời. Với robot Lego, máy bay điều khiển từ xa, máy in 3D, máy cắt lase để bàn, micro:bit, Arduinos, Makey Makeys, bạn có thể trở thành kỹ sư mỗi ngày.

Micro:bits và **Arduinos** là máy tính siêu nhỏ trên mạch đơn, có thể được lập trình để thực hiện một số thứ, ví dụ như theo dõi dữ liệu giống như thiết bị theo dõi tập luyện, hoặc như một thiết bị điều khiển trò chơi. Chúng được dùng để dạy học sinh viết code và lập trình.

Makey Makeys (một dạng điều khiển siêu nhỏ như micrco:bit hoặc Arduino) là bộ phát minh điện tử cho trẻ em, cho phép người dùng kết nối các vật dụng hàng ngày với chương trình máy tính bằng kẹp cá sấu. Từ đó người dùng có thể điều khiển vật thể theo cách khác.

Chương 7

THỬ NGHIỆM

Thử nghiệm là một bước quan trọng trong quy trình kỹ thuật. Là bước 4 trong quy trình thiết kế kỹ thuật, công việc này tốn nhiều thời gian nhất.

Chương trình có thể gặp nhiều sai sót. Có một số cách lập trình viên dùng để hạn chế vấn đề trong chương trình, để chúng chạy trơn tru.

LẬP TRÌNH PHÒNG NGỪA

LẬP TRÌNH PHÒNG NGỪA[1] là thiết kế chương trình sao cho nó vẫn hoạt động ngay cả khi xảy ra sự cố ngoài ý muốn. Một cách thực hiện lập trình phòng ngừa là cố ý để chương trình xảy ra sai sót và xem bạn nên giải quyết nó như thế nào.

Ví dụ, nếu bạn viết một chương trình để người dùng nhập địa chỉ thư điện tử, bạn có thể thử tạo ra lỗi sai bằng cách nhập văn bản không phải là địa chỉ thư điện tử.

[1] Defensive Programming.

Sau đó hãy đặt câu hỏi:

- Liệu chương trình vẫn chạy mượt nếu người dùng nhập số điện thoại hoặc tên vào phần địa chỉ thư điện tử?
- Bạn có thể làm gì để ngăn người dùng nhập thông tin sai?

TRƯỜNG HỢP SỬ DỤNG[1] và **TRƯỜNG HỢP THỬ NGHIỆM**[2] được dùng trong lập trình phòng ngừa, nhằm mục đích giúp bạn biết chương trình cần điều chỉnh ở đâu.

Trường hợp sử dụng

Trường hợp sử dụng là **một danh sách hành động** người dùng có thể thực hiện. Nó miêu tả cách sử dụng chương trình.

[1] Use cases.
[2] Test cases.

Danh sách có thể là một sơ đồ rất chi tiết, hoặc có thể chỉ là vài ý tưởng bạn nảy ra.

Quan trọng là người dùng có thể chạy chương trình mà không gặp lỗi. Sau khi bạn lập trình xong chương trình cho trò chơi, bạn nên chơi thử từ đầu tới cuối để chắc chắn nó hoạt động theo đúng ý bạn.

Trường hợp thử nghiệm

Trường hợp thử nghiệm kiểm tra xem người dùng có thể hoàn thành mục tiêu trong chương trình hay không. Đối với mỗi trường hợp sử dụng, sẽ có nhiều trường hợp thử nghiệm. Trường hợp thử nghiệm là bài kiểm tra cụ thể, chi tiết hơn. Chúng sử dụng các điều kiện mà chương trình có thể chạy. Trường hợp thử nghiệm dùng để xác định chức năng cần thiết. Kiểm tra trường hợp sử dụng là kiểm tra chức năng đó.

Trường hợp thử nghiệm tập trung vào một điều kiện (hoặc biến) một lúc để đảm bảo chương trình chạy trong mọi tình huống.

Ví dụ, nếu bạn đang tạo ra một trò chơi, bạn có thể tạo ra một trường hợp thử nghiệm để xem nhân vật của bạn có nhảy khi bạn ấn phím cách không.

Lập trình phòng ngừa nghĩa là đảm bảo rằng dù bạn ấn phím cách bao lâu thì nhân vật chỉ nhảy một lần.

Nếu trường hợp thử nghiệm thành công, bạn nên kiểm tra xem chuyện gì sẽ xảy ra nếu bạn giữ phím cách thay vì chỉ ấn một lần.

Nếu bạn áp dụng lập trình phòng ngừa cho trò chơi, nhân vật sẽ chỉ nhảy một lần.

Nếu bạn không lập trình phòng ngừa, hoặc quên kiểm tra chuyện gì xảy ra khi người chơi giữ phím cách, có thể nhân vật sẽ làm một vài động tác bạn không muốn.

Một ví dụ khác: Nếu bạn lập trình ứng dụng máy tính có chức năng chia một số cho 10, trường hợp thử nghiệm đảm bảo chương trình hoạt động với các dạng số khác nhau. Liệu chương trình có bị đơ nếu người dùng nhập số 1 hoặc 0?

Quan trọng là chọn các trường hợp thử nghiệm với dữ liệu đầu vào khác nhau. Ví dụ, đối với ứng dụng máy tính, thay vì chỉ thử với số từ 0-9, hãy chọn số nguyên dương và số nguyên âm, số hữu tỉ và số không.

GHI CHÚ TRONG LỚP TOÁN

Số nguyên dương: một số nguyên lớn hơn 0, không phải là phân số hay số thập phân.
Ví dụ: 2, 5, 9, 50.

Số nguyên âm: một số nguyên nhỏ hơn 0, không phải là phân số hay số thập phân.
Ví dụ: -1, -5, -150.

Số hữu tỉ: bất kỳ số nào có thể được viết dưới dạng một phân số hoặc một tỷ lệ.
Ví dụ: -5 có thể được viết thành $\frac{-5}{1}$ và 2,15 có thể được viết thành $\frac{215}{100}$.

KIỂM TRA KIẾN THỨC

1. Giải thích sự khác nhau giữa trường hợp sử dụng và trường hợp thử nghiệm.

2. NASA đang thử nghiệm một chương trình bằng cách xem phi hành gia có thể sử dụng chương trình hoàn thiện các bước trong nhiệm vụ, từ phóng tên lửa cho tới đi vào quỹ đạo Trái Đất. Đây là trường hợp sử dụng hay trường hợp thử nghiệm?

3. NASA đang thử nghiệm một phần nhỏ của chương trình phóng – phần kiểm soát động cơ – để xem liệu chương trình có chạy trơn tru với dữ liệu đầu vào khác không. Đây là trường hợp sử dụng hay trường hợp thử nghiệm?

4. Bạn là lập trình viên phòng ngừa nếu:
   A. Bạn nghĩ thử nghiệm thật tốn thời gian.
   B. Chương trình của bạn vẫn chạy dù có lỗi.
   C. Chương trình của bạn được bảo vệ khỏi virus.
   D. Tất cả ý trên.

5. Đối với một chương trình yêu cầu người dùng nhập địa điểm của hình vẽ graffiti để tình nguyện viên xóa đi, đáp án nào sau đây là trường hợp thử nghiệm? (Có thể chọn nhiều đáp án)

A. Liệu người dùng có thể tạo tài khoản và đăng địa điểm của hình vẽ graffiti?

B. Chuyện gì xảy ra nếu người dùng gõ tên thành phố vào ô mã bưu điện?

C. Chuyện gì xảy ra nếu người dùng gõ số âm vào ô mã bưu điện?

D. Liệu người dùng có thể chụp ảnh hình vẽ graffiti, đăng tải và đính kèm vào bài viết?

6. Tại sao kiểm tra với các dữ liệu đầu vào khác nhau lại quan trọng đối với trường hợp thử nghiệm?

7. Trường hợp thử nghiệm và trường hợp sử dụng được dùng để tìm _____ trong chương trình của bạn.

8. Tổ hợp số nào dưới đây là lựa chọn tốt nhất để điền vào trường hợp thử nghiệm cho chương trình máy tính?

A. 1, 2, 3, 4

B. -2, -4, -6, -8

C. 0,4, 0,5, 0,2, 0,6

D. -2, 0, 0,4, 16

ĐÁP ÁN

ĐÁP ÁN

1. Trường hợp sử dụng là bài kiểm tra tổng thể để xem người dùng có thể hoàn thiện một công việc. Trường hợp thử nghiệm là bài kiểm tra cụ thể, dùng các dữ liệu đầu vào đa dạng.

2. Trường hợp sử dụng.

3. Trường hợp thử nghiệm.

4. **B**

5. **B, C**

6. Nó quan trọng vì bạn cần đảm bảo chương trình chạy bất kể người dùng nhập dữ liệu nào. Ví dụ, nếu chương trình của bạn chỉ cho người dùng gõ chữ vào ô văn bản, có thể nó sẽ bị lỗi nếu người dùng gõ số hoặc # hoặc ký tự khác. Bạn cần chắc chắn rằng dữ liệu sai không làm hỏng chương trình.

7. Lỗi sai (bug)

8. **D**

Chương 8

GHI CHÚ

ĐOẠN GHI CHÚ là thông tin về code của chương trình. Có hai dạng ghi chú chính: **bình luận** và tài liệu **README**.

BÌNH LUẬN

Tất cả các ngôn ngữ lập trình đều có cách để thêm bình luận vào code. Bình luận không phải là một phần chương trình, chương trình sẽ bỏ qua và không hiển thị chúng khi chạy. Bình luận là lời nhắn do lập trình viên viết về code của chương trình. Chúng có thể là lời giải thích cho nhiệm vụ của code, hoặc lời nhắc nhở về thứ cần thêm hoặc thay đổi trong code, hoặc câu hỏi cho người lập trình viên xem code.

Ví dụ:

Thêm đồng hồ đếm ngược vào phần trên màn hình.

Bình luận về code giúp bạn nhớ nhiệm vụ của mỗi phần. Nó rất hữu ích đối với các chương trình lớn vì bạn sẽ không nhớ mỗi dòng code được viết để làm gì. Bình luận cũng hữu ích với các chương trình ngắn vì một thời gian dài sau bạn sẽ muốn nhìn lại code.

Đôi khi bạn muốn dùng lại code từ chương trình khác, hay code do bạn hoặc người khác viết. Bình luận giúp bạn nhanh chóng tìm được đoạn code bạn muốn dùng lại. Ví dụ, nếu bạn đang tạo ra một trò chơi, thay vì tự viết code để di chuyển nhân vật, bạn có thể sao chép đoạn code tương tự từ chương trình khác. Bạn có thể viết bình luận để biết đó là đoạn code nào trong chương trình của bạn.

Bình luận cũng được dùng để hỗ trợ sửa lỗi chương trình. Vì bình luận không đọc được khi chương trình chạy, bạn có khiến máy tính bỏ qua đoạn code (hoặc lỗi) một cách dễ dàng mà không cần xóa chúng bằng cách **BIẾN ĐOẠN CODE ĐÓ THÀNH BÌNH LUẬN**.

BIẾN ĐOẠN CODE THÀNH BÌNH LUẬN

Đánh dấu một phần code là bình luận để chương trình không chạy nó cùng với phần code còn lại.

Bình luận có thể là lời nhắc cho bản thân khi bạn viết code.

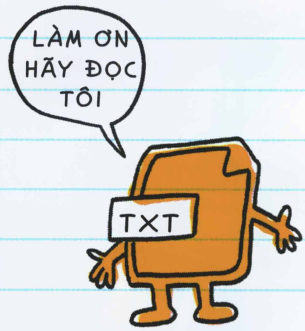
Ví dụ, bạn có thể để lại lời nhắn cho bản thân:

#BUG (#lỗi) hoặc **#FIXME** (#Sửa_tôi) - một lỗi sai đã được phát hiện mà bạn sẽ quay lại sửa sau.

#TODO (#Cần_làm) - lời nhắn nhắc bạn về đoạn code cần viết mà bạn làm sau.

README

Tài liệu **README**[1] (thường được viết hoa để khiến bạn chú ý) đưa thông tin về chương trình, bao gồm những tài liệu bên trong, cách cài đặt, cách sử dụng, dẫn nguồn, ghi nhận đóng góp, hoặc các lỗi đã phát hiện.



Tài liệu README có thể dành cho người dùng, lập trình viên khác, hoặc cả hai. Thường mỗi chương trình chỉ có một tài liệu README, và nó thường là một **ĐOẠN VĂN BẢN ĐƠN THUẦN**.

VĂN BẢN ĐƠN THUẦN
Đoạn văn bản không có định dạng (như in đậm, thụt lề, khoảng cách giữa các đoạn văn) chỉ có chữ và dấu câu.

[1] Hãy đọc tôi.

Mặc dù tài liệu README có thể bao gồm thông tin đến người dùng, hướng dẫn người dùng (user guide) là cách phổ biến hơn khi hướng dẫn người dùng sử dụng chương trình. Phương pháp này thông thườngchỉ có ở các chương trình lớn và phức tạp như Microsoft Word hay Excel.

ĐỌC HƯỚNG DẪN SỬ DỤNG = Ý KIẾN HAY

KIỂM TRA KIẾN THỨC

1. Bạn có thể làm 2 loại ghi chú nào để giúp người khác hiểu chương trình của mình?

2. Loại ghi chú nào giúp bạn nhớ nhiệm vụ của những đoạn code?

3. Tại sao biến đoạn code thành bình luận lại giúp sửa lỗi?

4. Tại sao bình luận hữu ích với bạn và lập trình viên khác?

5. Đáp án nào sau đây là lý do để dùng ghi chú? (có thể chọn nhiều đáp án)

   A. Giúp bạn dễ dàng tìm được đoạn code mình cần.
   B. Giúp bạn dễ dàng hợp tác cùng người khác.
   C. Bạn có thể viết lời nhắn cho bản thân khi lập trình.
   D. Giúp bạn tìm lỗi.
   E. Ghi danh.
   F. Giải thích cách sử dụng chương trình.

6. Nếu bạn muốn thêm lời nhắn vào đoạn code để quay lại và sửa lỗi sai, bạn có thể thêm bình luận _____.

7. Thứ nào sau đây giúp người dùng cài đặt chương trình: tài liệu README hay bình luận trong code?

8. Hướng dẫn sử dụng là gì?

ĐÁP ÁN

1. Bình luận và tài liệu README

2. Bình luận

3. Bạn có thể ngăn một đoạn code chạy mà không cần xóa chúng.

4. Nếu bạn đang làm chung dự án với người khác, đọc bình luận giải thích nhiệm vụ của đoạn code sẽ dễ dàng hơn đọc đoạn code và diễn giải ý nghĩa của nó.

5. **A**, **B**, **C**, **D**, **E**, **F** (tất cả câu trả lời đều đúng.)

6. #lỗi hoặc #sửa_tôi

7. Tài liệu README

8. Hướng dẫn sử dụng là tài liệu giúp người dùng hiểu về công dụng của chương trình.

Chương 9

TÍCH HỢP PHẢN HỒI

THIẾT KẾ TẬP TRUNG VÀO NGƯỜI DÙNG

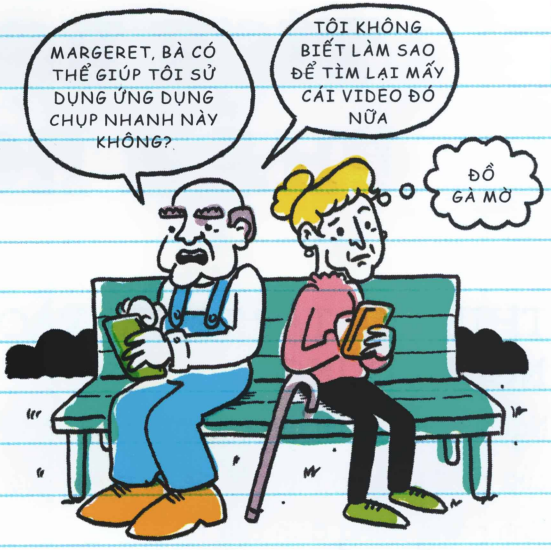
Người dùng đánh giá chương trình có tốt không. Ý kiến của họ dựa trên mức độ dễ sử dụng của chương trình và tính năng đúng đắn của nó.

THIẾT KẾ TẬP TRUNG VÀO NGƯỜI DÙNG là cân nhắc nhu cầu và mong muốn của người dùng trong mọi giai đoạn tạo ra chương trình. Để khiến chương trình dễ sử dụng đối với nhiều người nhất có thể, bạn nên hỏi ý kiến các kiểu người dùng khác nhau về chương trình.

Tích hợp phản hồi là sử dụng thông tin bạn thu thập được từ người dùng để chỉnh sửa thiết kế.

Khi tạo ra một thiết kế tập trung vào người dùng, bạn nên tập trung vào vài thứ:

- **KHẢ NĂNG SỬ DỤNG** - Cách dùng có dễ hiểu không?



- **KHẢ NĂNG TIẾP CẬN** - Những người có khả năng khác nhau có thể sử dụng nó không?

- **NỘI DUNG** - chương trình của bạn có hợp lý đối với khán giả mục tiêu?

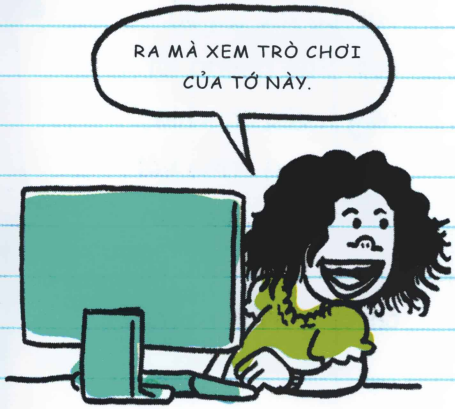
đối tượng chương trình hướng tới

THU THẬP PHẢN HỒI

Trước khi ra mắt chương trình với thế giới, lập trình viên sử dụng thử nghiệm **ALPHA** và **BETA** để nhận phản hồi từ người dùng. Những người dùng này chỉ ra lỗi hoặc vấn đề trong chương trình.

Thử nghiệm Alpha là vòng đầu tiên của thử nghiệm người dùng, thường được thực hiện trước khi chương trình hoàn thành. Người thử nghiệm alpha là bạn bè hoặc những người bạn tin tưởng. Tại thời điểm này, code của bạn chưa hoàn thiện, nên họ sẽ tìm được rất nhiều lỗi. Trong lớp viết code, thử nghiệm alpha được thực hiện bằng cách nhờ bạn cùng lớp xem chương trình của bạn, cho dù nó chỉ xong một nửa.

Alpha (α) và beta (β) là hai chữ cái đầu tiên của bảng chữ cái Hy Lạp.



Thử nghiệm beta là vòng hai của thử nghiệm người dùng, thường được thực hiện sau khi chương trình hoàn thành. Người thử nghiệm beta là một nhóm người dùng tiềm năng được chọn lọc.

VÍ DỤ: Nếu bạn tạo ra ứng dụng giúp bạn bè ôn tập cho bài kiểm tra toán sắp tới, người thử nghiệm beta sẽ là một nhóm bạn cùng lớp toán của bạn. Người thử nghiệm beta thường tìm thấy ít lỗi hơn người thử nghiệm alpha.

Để người dùng tiềm năng xem chương trình của bạn giúp xác định những lỗi bạn có thể đã bỏ qua. Sửa những lỗi này trước khi chia sẻ chương trình với mọi người giúp bạn sửa hết lỗi trước khi chương trình ra mắt.

KIỂM TRA KIẾN THỨC

1. Thiết kế tập trung vào người dùng là gì?

2. Đáp án nào sau đây không phải là quy tắc của thiết kế tập trung vào người dùng?

   A. Giải trí

   B. Khả năng sử dụng

   C. Khả năng tiếp cận

   D. Nội dung

3. Phản hồi là gì?

4. Thử nghiệm alpha khác như thế nào với thử nghiệm beta?

5. Người thử nghiệm alpha hay người thử nghiệm beta sẽ tìm được nhiều lỗi hơn?

6. Một chương trình có khả năng tiếp cận là gì, và tại sao chương trình có khả năng tiếp cận lại quan trọng?

7. Bạn thiết kế một chương trình theo dõi thời lượng ngủ và chất lượng giấc ngủ. Khán giả mục tiêu của bạn là gì?

A. Người mua sắm ở Target

B. Người mệt mỏi

C. Bác sĩ giải phẫu thần kinh

D. Lập trình viên

8. Chuyện gì xảy ra nếu bạn ra mắt chương trình trước khi thử nghiệm nó?

ĐÁP ÁN

1. Thiết kế tập trung vào người dùng là tạo ra chương trình dựa trên điều người dùng muốn.

2. A

3. Phản hồi là thông tin thu thập từ người thử nghiệm chương trình.

4. Thử nghiệm alpha là vòng đầu của thử nghiệm, nó được thực hiện với một nhóm nhỏ những người đáng tin. Thử nghiệm beta là vòng hai của thử nghiệm, nó được thực hiện với một nhóm người thuộc khán giả mục tiêu.

5. Người thử nghiệm alpha sẽ tìm thấy nhiều lỗi hơn vì họ là người đầu tiên sử dụng chương trình, và dự án thậm chí có thể chưa hoàn thiện. Khi người thử nghiệm beta xem chương trình, hầu hết lỗi đã được khắc phục.

6. Một chương trình được coi là có khả năng tiếp cận nếu những người có khả năng khác nhau đều có thể sử dụng. Là một lập trình viên, bạn không muốn loại trừ nhóm khách hàng nào dựa trên khả năng của họ.

7. B

8. Mặc dù chương trình của bạn có vẻ hoàn hảo, có khả năng rằng nó vẫn còn lỗi hoặc người dùng sẽ đề xuất chỉnh sửa.

Chương 10

HỢP TÁC

LÀM VIỆC NHÓM

Việc hợp tác, hay làm việc với người khác, trong một dự án lập trình rất hữu ích vì mỗi người sẽ có thế mạnh riêng. Vài người cùng làm một chương trình lớn (chỉ dẫn cho máy tính) sẽ hoàn thành dự án nhanh hơn. Mỗi lập trình viên sẽ được giảm nhẹ gánh nặng.

Tàu Con Thoi chạy 400.000 dòng code. Ô tô cao cấp chạy 100 triệu dòng code. Facebook được tạo nên bởi khoảng 60 triệu dòng code. Bạn sẽ mất hơn một đời người để tự viết hàng triệu dòng code.

Hợp tác sẽ chia nhỏ công việc, tập trung vào thế mạnh bản thân giúp bạn nhanh chóng xây dựng chương trình hiệu quả.

Mẹo để giúp các thành viên trong nhóm hợp tác hiệu quả:

Đặt ra kỳ vọng rõ ràng. Đảm bảo mọi người biết rõ họ cần làm gì cho dự án. Đừng bỏ quên bất kỳ phần nào của dự án.

Tập trung vào thế mạnh bản thân. Chia công việc sao cho mỗi người được làm phần họ thích và có thể làm tốt.

Đặt ra mốc thời gian hợp lý. Cho bản thân thời gian để hoàn thành công việc, đồng thời thúc đẩy bản thân làm việc chăm chỉ để hoàn thành dự án.

Phân công đồng đều. Dù mỗi thành viên làm phần việc khác nhau, hãy đảm bảo rằng công việc được phân chia công bằng để không ai cảm thấy họ phải làm tất.

Phân vai rõ ràng. Quyết định ai nên là người ghi chép, thử nghiệm, sửa lỗi, hoặc bất kỳ vai trò nào bạn cần. Vai trò rõ ràng sẽ giúp bạn theo dõi tiến độ và tập trung vào công việc của mình.

THIẾT KẾ CHO ỨNG DỤNG

VIẾT CODE CHO ỨNG DỤNG

VẼ CHO ỨNG DỤNG

LẬP TRÌNH ĐÔI

Khi dự án rất nhỏ, giống như dự án trong lớp học, lập trình cùng một đội lớn là không cần thiết. **LẬP TRÌNH ĐÔI** là hai người cùng làm việc trên một máy tính để hoàn thành dự án. Lập trình viên sử dụng cách này vì họ có thể làm việc nhanh hơn và ít mắc lỗi hơn.

Khi lập trình đôi, mỗi lập trình viên phải có vai trò cụ thể. Một người là **TÀI XẾ**, người kia là **HOA TIÊU**. Tài xế điều khiển bàn phím. Họ có trách nhiệm viết code và rà soát lại công việc. Hoa tiêu có trách nhiệm nói cho tài xế biết nên viết code nào và tránh mắc lỗi khi lập trình.

Cả người tài xế và hoa tiêu nên giao tiếp với nhau khi viết code. Cả hai nên thường xuyên đổi vai trò để mỗi người đều có cơ hội làm tài xế hoặc hoa tiêu.

TÌM KIẾM SỰ TRỢ GIÚP TỪ CỘNG ĐỒNG

Hầu hết lập trình viên dành rất nhiều thời gian tìm kiếm giải pháp cho vấn đề lập trình của họ. Khi bạn tạo ra chương trình, bạn có thể sử dụng nhiều nguồn để hoàn thành dự án. Ngay cả những lập trình viên thạo nghề vẫn nhờ những lập trình viên khác giúp đỡ.

Khi bạn lập trình, bạn nên tìm cơ hội vận dụng sản phẩm của người khác. Điều này KHÔNG có nghĩa là bạn được sao chép code của người khác và nhận là của mình. Thay vào đó, bạn nên tìm ý tưởng từ chương trình của người khác, hoặc chỉ sao chép một đoạn code ngắn.

Bạn luôn phải **GHI NHẬN** những công sức của lập trình viên. Nếu chương trình hoặc một phần chương trình của bạn được viết bởi bạn nhưng được lấy cảm hứng từ người khác, bạn cần ghi công họ trong tài liệu README. Nếu bạn muốn sao chép một phần chương trình của người khác, hãy hỏi ý kiến tác giả hoặc hỏi xin bản quyền. Sau đó, làm rõ phần nào là code của họ và phần nào là của bạn. Khi bạn sao chép

code, bạn nên thêm phần attribution (người đóng góp) vào cả trong bình luận và tài liệu README.

> Tôi sử dụng dự án mẫu của @the_coding_master, và tôi điều chỉnh nó để thêm hiệu ứng ánh sáng, thêm vào bộ đồ hóa trang con rùa, và thêm nhạc của @music_lover. Cảm ơn @gamer_pro vì nguồn cảm hứng!

Khi bạn ghi nhận đóng góp, bạn cho thấy mình trân trọng thành quả lao động chăm chỉ của người khác và bạn đang tạo ra một cộng đồng biết sẻ chia. Đồng thời, ghi nhận đóng góp của người khác là một hành động đẹp - trong một số trường hợp, nó còn là luật phát nữa.

Ghi nhận đóng góp khi lập trình giống như trích dẫn hoặc diễn giải tác giả khác khi bạn viết bài trong lớp Ngữ văn.

TRÍCH DẪN
Nhắc đến nguồn thông tin.

DIỄN GIẢI
Sử dụng ý tưởng của người khác nhưng nói lại theo cách của bản thân.

SỬ DỤNG THƯ VIỆN

THƯ VIỆN (hay còn gọi là mô-đun hoặc phần mở rộng) cho bạn truy cập vào những **TÍNH NĂNG SẴN CÓ** hữu ích. Ví dụ, nếu bạn đang tạo ra một trò chơi, bạn có thể gõ hàng trăm dòng code để điều khiển nhân vật nhảy xung quanh và tương tác với nhân vật khác trên màn hình. Hoặc bạn có thể sử dụng một thư viện có sẵn code để nhân vật nhảy xung quanh và tương tác với nhân vật khác. Một số thư viện yêu cầu bạn ghi nhận đóng góp trong chương trình, hoặc ở bình luận trong đoạn code.

TÍNH NĂNG SẴN CÓ
Những dòng code đã được viết cho một hành động cụ thể.

Bạn luôn nên tìm kiếm sự giúp đỡ hoặc một thư viện có thể hỗ trợ dự án của bạn.

ĐÂY KHÔNG PHẢI LÀ THƯ VIỆN MÌNH NGHĨ ĐẾN.

KIỂM TRA KIẾN THỨC

1. Nêu hai lợi ích khi lập trình theo nhóm.

2. Điều này là đúng hay sai: Hầu hết lập trình viên không nhờ giúp đỡ.

3. Hai hành động bạn phải làm khi dùng code của người khác?

4. Nếu bạn không sao chép hoàn toàn code của người khác, nhưng lấy cảm hứng từ nó, bạn có cần dẫn nguồn?

5. Hoa tiêu sẽ giúp đỡ tài xế như thế nào?

6. Tại sao việc đổi vai thường xuyên khi lập trình theo đôi lại quan trọng?

7. Tại sao dùng thư viện vật lý lại có ích đối với lập trình viên trò chơi? (thư viện có những tính năng khiến nhân vật hoặc vật thể của bạn trông như đang chuyển động hoặc tương tác thật)

8. Bạn sẽ cảm thấy thế nào nếu có người dùng code của bạn mà không xin phép, không ghi nhận, và nhận đó là code của họ?

9. Ghi nhận đóng góp là (được chọn nhiều đáp án):

A. Là một hành động đẹp

B. Luật pháp

C. Cách thể hiện sự tôn trọng

D. Cách thúc đẩy một cộng đồng biết sẻ chia

ĐÁP ÁN

1. Bất kỳ hai đáp án nào sau đây đều là lợi ích:
   - Bạn có thể hoàn thành khối lượng công việc lớn.
   - Bạn có thể tận dụng nhiều thế mạnh.
   - Bạn có đồng đội cùng quan sát để tránh mắc và phát hiện lỗi.
   - Bạn có thể học hỏi từ người khác.

2. Sai

3. Hỏi xin quyền sử dụng hoặc ghi nhận đóng góp.

4. Đúng

5. Có người nhìn tổng quan sẽ giúp tài xế tập trung viết code và mắc ít lỗi hơn.

6. Đảo vai cho cả hai cơ hội nhìn tổng quan và tìm lỗi sai từ góc nhìn khác.

7. Sử dụng thư viện tiết kiệm thời gian và tránh sai sót. Với thư viện vật lý, lập trình viên trò chơi tốn ít thời gian viết code hoạt động của nhân vật và có thêm thời gian lập trình phần còn lại của trò chơi.

8. Nhiều lập trình viên sẽ buồn hoặc giận dữ nếu như ai đó tuyên bố quyền tác giả đối với code của họ.

9. A, B, C, D

BÀI 4

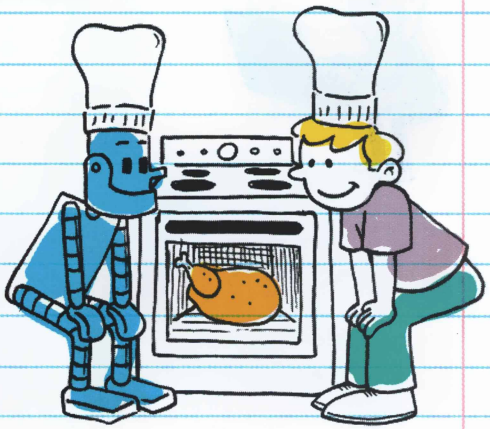
Thuật toán và lập trình

Chương 11

SỬ DỤNG THUẬT TOÁN

THUẬT TOÁN

Cả con người và máy tính đều sử dụng thuật toán. Thuật toán là quy trình hoặc các bước có thể thực hiện. Công thức nấu ăn là ví dụ về thuật toán cả con người và máy tính đều sử dụng.

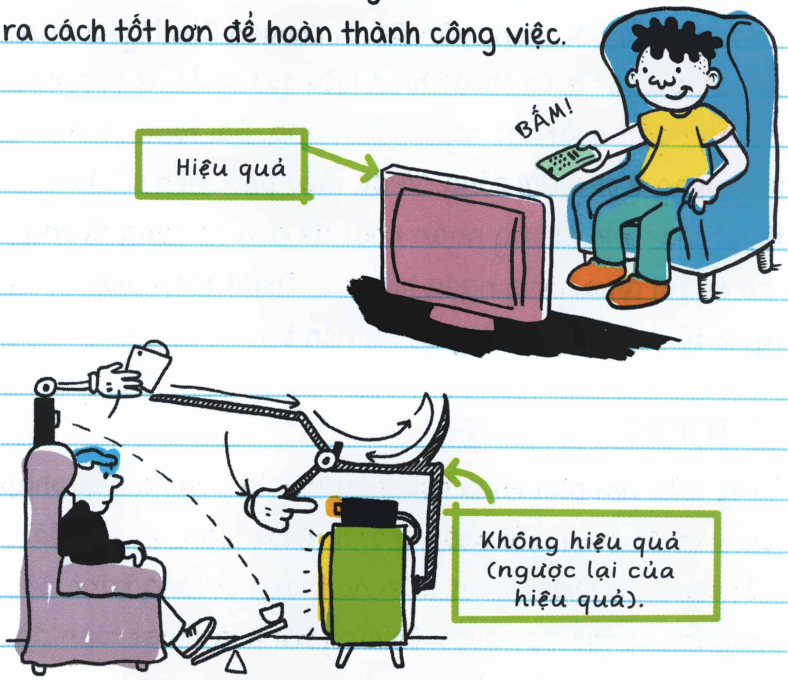


Thuật toán đưa ra chỉ dẫn rõ ràng cho những công việc lặp lại. Các thuật toán khác nhau có thể được sử dụng để hoàn thành cùng một công việc. Ví dụ, có nhiều cách để chia hai số, ta có thể sử dụng phép chia dài hoặc phép trừ lặp lại. Cả hai cách đều ra cùng một kết quả,

nhưng có thể có một cách tốt hơn. Đối với khoa học máy tính, chúng ta luôn tìm kiếm thuật toán tốt hơn.

Thuật toán tốt hơn sẽ nhanh, đơn giản và hiệu quả hơn.

Thuật toán hữu ích vì chúng đưa ra cách tốt hơn để hoàn thành công việc.



Nghiên cứu thuật toán

Phát triển thuật toàn là một ngành riêng trong khoa học máy tính, và nó rất giống với nghiên cứu toán học. Ví dụ, nhà toán học và nhà khoa học máy tính đều đã nghiên cứu VẤN ĐỀ DI CHUYỂN CỦA NGƯỜI BÁN HÀNG trong gần 100 năm.

Vấn Đề Di Chuyển Của Người Bán Hàng đặt ra câu hỏi: "Cho một danh sách các thành phố và khoảng cách giữa các thành phố, đâu là đường ngắn nhất mà người bán có thể tới mỗi thành phố và về nhà?"

Mục tiêu của bài toán "Vấn Đề Di Chuyển Của Người Bán Hàng" là phát triển thuật toán hiệu quả nhất để đưa ra đường ngắn nhất cho người bán hàng. Bài toán có thể ứng dụng trong đời sống. Ví dụ, điều phối viên sử dụng thuật toán tìm đường ngắn nhất để đưa ra cung đường cho xe tải giao hàng ngày hôm đó. Thuật toán giúp công ty xe tải tiết kiệm thời gian và tiền bạc.

CHƯƠNG TRÌNH

Công việc của nhà khoa học máy tính là đưa ra giải pháp cho vấn đề, biến giải pháp thành thuật toán, và sau đó viết code cho thuật toán, biến nó thành chương trình để máy tính có thể đọc được. Chương trình là thuật toán đã được dịch thành chỉ dẫn cho máy tính (viết code).

Thuật toán thường được viết để hầu hết mọi người đều hiểu. Ít người có thể đọc chương trình máy tính và hiểu nó, vì nó sử dụng tổ hợp số, chữ cái, và biểu tượng được

sắp xếp theo cách không quen thuộc với chúng ta - nhưng máy tính sẽ hiểu được.

Lập trình viên đưa ra chỉ dẫn cụ thể cho máy tính bằng cách sử dụng **NGÔN NGỮ LẬP TRÌNH**. Ngôn ngữ lập trình sử dụng sự kết hợp số, chữ cái, biểu tượng, và định dạng để cho máy tính biết cần phải làm gì.

VÍ DỤ: Giả dụ bạn muốn biết hôm nay có phải là sinh nhật bạn bè. Bạn có thể viết thuật toán hoặc chương trình sau:

THUẬT TOÁN	CHƯƠNG TRÌNH (PYTHON)
1. Tìm ngày hôm nay.	#Import datetime library from datetime import *

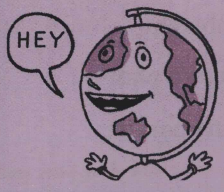
2. Hỏi bạn bè ngày sinh nhật của họ.	#Get Today's Date
	today = date.today()

3. Xác định xem hôm nay có phải sinh nhật bạn bè.	#Get User's Birthday
	dob_str = input("What is your Date of Birth? dd/mm/yyyy")

THUẬT TOÁN	CHƯƠNG TRÌNH (PYTHON)

4. Nếu hôm nay là sinh nhật bạn bè, gửi họ lời chúc "Chúc mừng sinh nhật!" Nếu không phải, gửi họ "Hôm nay không phải sinh nhật bạn"	#Convert user input into a date
	dob_data = dob_str.split("/")
	dobDay = int(dob_data[0])
	dobMonth = int(dob_data[1])
	dobYear = int(dob_data[2])
	dob = date(dobYear,dobMonth,dobDay)#Determine if today is the user's birthday
	thisYear = today.year
	nextBirthday = date(thisYear,dobMonth,dobDay)
	
	if today == nextBirthday:
	print("Happy Birthday!")
	else:
	print("Today is not your birthday.")

Có một truyền thống khi bạn bắt đầu học lập trình, đó là viết một chương trình đơn giản để máy tính hiển thị "Xin chào, thế giới!".

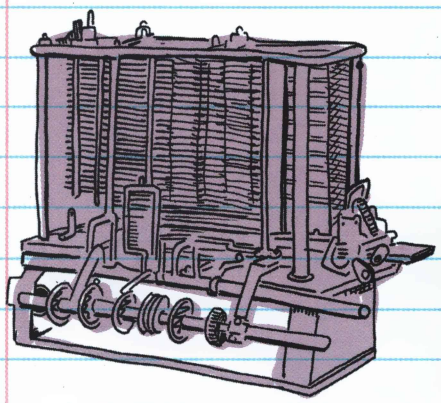


THỜI KỲ ĐẦU CỦA LẬP TRÌNH

Lập trình máy tính còn gọi là viết code. Viết code được cho là xuất hiện vào thế kỷ XIX.

Ada Lovelace được cho là một trong những **LẬP TRÌNH VIÊN ĐẦU TIÊN**. Năm **1843**, Lovelace viết thuật toán cho một chiếc máy có tên là **ĐỘNG CƠ PHÂN TÍCH** - một chiếc máy tính cơ khí.

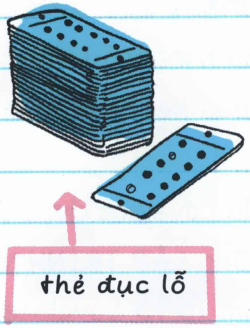
Augusta Ada King,
Nữ bá tước Lovelace



Động Cơ Phân Tích.

Động Cơ Phân Tích được thiết kế bởi **CHARLES BABBAGE**. Thay vì chạy bằng điện như máy tính hiện đại, chiếc máy tính này sử dụng bánh răng và động cơ hơi nước. Nó được dùng để đếm và xử lý các phép toán đơn giản.

Động Cơ Phân Tích chạy chương trình đơn giản tạo nên bởi thẻ đục lỗ. Máy tính sẽ đọc những lỗ trên thẻ như chỉ dẫn. Thẻ đục lỗ được dùng để lập trình máy tính.



Lovelace tin máy tính có thể được chỉ dẫn để làm những việc khác. Bà biết nó có thể được lập trình. Lovelace là người đầu tiên nhận ra tiềm năng vượt xa công dụng tính toán của Động Cơ Phân Tích.

KIỂM TRA KIẾN THỨC

1. Giải thích sự khác biệt giữa thuật toán và chương trình.

2. Đáp án nào sau đây KHÔNG PHẢI là thuật toán?
   A. Các bước làm bánh burrito.
   B. Hướng dẫn tết tóc kiểu Pháp.
   C. Hướng dẫn xây dựng mô hình máy bay.
   D. Lời nhắn gửi yêu thương ẩn danh tới người bạn mến.

3. Tại sao những thuật toán tốt nhất lại hiệu quả? Tại sao những nhà khoa học máy tính luôn tìm kiếm thuật toán hiệu quả hơn?

4. Chọn thuật toán đồng thời là chương trình.
   A. Hướng dẫn chơi nhạc cụ
   B. Microsoft Word
   C. Snapchat
   D. Hướng dẫn chế tạo máy bay không người lái

5. Điều này là đúng hay sai: Những thuật toán khác nhau có thể được dùng để hoàn thành một công việc theo nhiều cách khác nhau.

6. Máy tính hiểu dạng ngôn ngữ nào?

7. Tại sao thuật toán cần được viết theo cách riêng để máy tính hiểu?

8. Điều này là đúng hay sai: Ada Lovelace viết chương trình máy tính đầu tiên cho máy tính Apple.

9. Ai là người thiết kế Động Cơ Phân Tích?

10. Động Cơ Phân Tích được thiết kế để làm gì?

11. Tại sao Ada Lovelace là người đi trước thời đại?

ĐÁP ÁN

1. Thuật toán là những chỉ dẫn để hoàn thành công việc, và chương trình là những chỉ dẫn chỉ có máy tính hiểu.

2. D

3. Thuật toán hiệu quả chạy nhanh hơn và tốn ít bộ nhớ hơn.

4. B, C

5. Đúng

6. Ngôn ngữ lập trình

7. Máy tính cần chỉ dẫn cụ thể và chỉ hiểu ngôn ngữ lập trình.

8. Sai. Ada Lovelace viết chương trình đầu tiên cho Động Cơ Phân Tích.

9. Charles Babbage

10. Động Cơ Phân Tích được thiết kế để thực hiện chức năng tính toán, như một chiếc máy tính.

11. Bà thấy tiềm năng không ai thấy. Bà tin rằng có thể lập trình máy móc cho nhiều mục đích ngoài mục đích tính toán.

Chương 12

NGÔN NGỮ LẬP TRÌNH

Có *hàng trăm* ngôn ngữ lập trình, và lập trình viên sử dụng các ngôn ngữ khác nhau cho các dạng công việc khác nhau. Ngôn ngữ như JavaScript, PHP, và Python được dùng để lập trình **TRANG WEB** - dạng tài liệu hiển thị trên trình duyệt web (như Chrome, Safari, hoặc Firefox). Tổ hợp các trang web gọi là **WEBSITE**.

ỨNG DỤNG ĐIỆN THOẠI được lập trình bởi ngôn ngữ Swift, Java, C, và các ngôn ngữ khác.

ỨNG DỤNG ĐIỆN THOẠI
Dạng phần mềm được thiết kế để chạy trên thiết bị di động như điện thoại và máy tính bảng.

SỬ DỤNG NGÔN NGỮ LẬP TRÌNH

Vì mỗi ngôn ngữ lập trình có thế mạnh riêng (ví dụ, một số nhanh hơn, một số dễ viết hơn, hoặc một số khác hiển thị đồ họa tốt hơn), lập trình viên có thể sử dụng vài ngôn ngữ cho một chương trình lớn. Ví dụ:

- Facebook sử dụng Python, JavaScript, PHP, và những ngôn ngữ khác.
- YouTube sử dụng Python, JavaScript, C++ và những ngôn ngữ khác.
- Twitter sử dụng JavaScript, C++, Ruby và những ngôn ngữ khác.

Tất cả những website trên đều sử dụng JavaScript. Vì JavaScript (cùng với HTML và CSS) là **NGÔN NGỮ FRONT-END** - loại ngôn ngữ dùng để xây dựng phần website hiển thị trên màn hình.

Website như Facebook, Youtube và Twitter lưu trữ lượng thông tin khổng lồ (bài đăng, bình luận, hồ sơ cá nhân, video,v.v.). Vì vậy họ cần dạng ngôn ngữ lập trình khác (ngôn ngữ **BACK-END**) để sắp xếp và kết nối tất cả thông tin với phần hiển thị.

Các ngôn ngữ back-end khác nhau được dùng cho các mục đích khác nhau. Ví dụ, C++ chạy nhanh hơn các ngôn ngữ lập trình khác nhưng lại khó viết hơn. Vì vậy công ty có thể sẽ lập trình sản phẩm của họ bằng ngôn ngữ dễ hơn như Python và sau đó thay thế một số phần của sản phẩm với ngôn ngữ C++ để tăng tốc độ xử lý thông tin.

Không lập trình viên máy tính nào biết mọi ngôn ngữ, cũng như không ai biết mọi ngôn ngữ nói.

Tuy nhiên, bạn nên biết một vài ngôn ngữ. Ví dụ, bạn có thể kết hợp ngôn ngữ tốt cho đồ họa với ngôn ngữ tốt để xử lý đoạn code dài.

CÁCH MÁY TÍNH ĐỌC CODE

CODE = **NGÔN NGỮ MÁY TÍNH HIỂU ĐƯỢC**

Mã nhị phân

Máy tính xử lý thông tin bằng cách sử dụng các **mạch điện** có thể bật và tắt. Các nhà khoa học máy tính sử dụng số 1 và 0 để thể hiện trạng thái bật và tắt. 1 và 0 là một dạng mã hóa gọi là **MÃ NHỊ PHÂN**, hoặc **CODE MÁY** (code). Số nhị phân chỉ có thể là 1 hoặc 0.

Đường dẫn hoặc tổ hợp đường dẫn truyền điện.

Đối với mã nhị phân, 1 và 0 được sử dụng thường xuyên, nhưng bạn cũng có thể biểu diễn hệ nhị phân bằng "Bật" và "Tắt" hoặc "Đúng" và "Sai". Tất cả từ ngữ, số, chữ cái, biểu tượng, hình ảnh, video, chương trình, biểu tượng cảm xúc và âm nhạc đều có thể được biểu diễn dưới dạng nhị phân.

MÃ NHỊ PHÂN
Mã nhị phân là cách máy tính giao tiếp và biểu diễn thông tin.

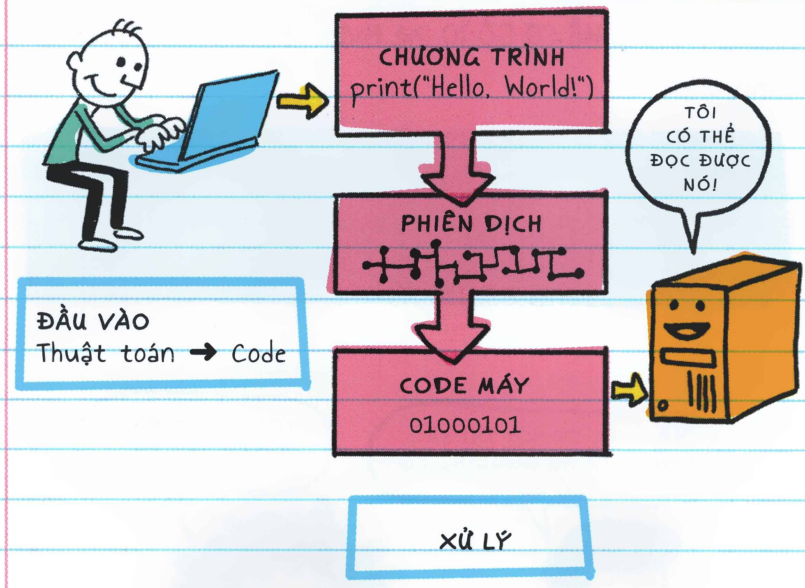
1 = BẬT
0 = TẮT

CODE MÁY
Ngôn ngữ đơn giản nhất. Mọi ngôn ngữ lập trình đều được chuyển thành ngôn ngữ này trước khi máy tính chạy nó.

CPU của máy tính chỉ có thể đọc chỉ dẫn được viết bằng code máy. Vì vậy, sau khi con người đưa chương trình vào máy tính, một chương trình máy tính sẽ biến nó thành mã nhị phân. Quá trình dịch (thay đổi) chương trình thành Code máy gọi là **PHIÊN DỊCH.**

PHIÊN DỊCH
Dịch ngôn ngữ lập trình thành Code máy.

Dù bao nhiêu ngôn ngữ lập trình khác nhau được sử dụng để viết cụm từ "Xin chào, Thế giới!", code máy của mỗi phiên bản đều giống nhau.



MÃ NHỊ PHÂN = CODE MÁY

Chương trình phiên dịch thay đổi chỉ dẫn từ ngôn ngữ lập trình sang code máy. Trước khi bộ phiên dịch được phát minh, lập trình viên phải viết chương trình theo ngôn ngữ gần với code máy. Năm 1952, GRACE HOPPER viết chương trình phiên dịch đầu tiên. Chương trình phiên dịch của cô giúp lập trình tốn ít thời gian viết code máy và có thêm thời gian lập trình.

Biểu thị số bằng hệ nhị phân

Hệ nhị phân là một cách sử dụng số để đếm hoặc biểu thị số. Hệ nhị phân có nghĩa là **Hệ cơ số 2**, vì nó sử dụng 2 số (0 và 1), trong khi cách đếm thông thường của chúng ta được gọi là **Hệ cơ số 10** (hệ thập phân) vì nó sử dụng hàng chục.

> Chữ "nhị" trong hệ nhị phân nghĩa là hai. Mặc dù hệ nhị phân chỉ dùng 2 số, nó có thể biểu diễn các giá trị tương đương hệ cơ số 10.

Đối với hệ cơ số 10, vị trí của mỗi số có giá trị gấp 10 lần vị trí số bên phải nó. Khi số càng lớn, ta thêm số vào vị trí bên trái.

VÍ DỤ, số 237 nghĩa là hai trăm ba mươi bảy. Giá trị của mỗi số phụ thuộc vào vị trí của nó trong số 237.

100s	10s	1s
2	3	7

nhị = 2

Đối với hệ nhị phân, các số sau có giá trị gấp đôi số trước. Từ phải qua trái, số của hệ nhị phân lần lượt có giá trị theo hàng là 1, 2, 4, 8, 16, 32, 64 và cứ thế nhân đôi.

Giá trị theo hàng của hệ nhị phân: Từ trái qua phải, mỗi hàng có giá trị gấp đôi hàng trước.

Để biểu diễn số 1 trong hệ nhị phân, viết 1 vào vị trí đầu tiên.

Vị trí 128	Vị trí 64	Vị trí 32	Vị trí 16	Vị trí 8	Vị trí 4	Vị trí 2	Vị trí 1
							1

Không có số "2" trong hệ nhị phân. Để biểu diễn số 2, đặt số 0 vào vị trí đầu tiên, sau đó đặt số 1 vào vị trí thứ 2.

Vị trí 128	Vị trí 64	Vị trí 32	Vị trí 16	Vị trí 8	Vị trí 4	Vị trí 2	Vị trí 1
						1	0

BẬT TẮT

0 ở vị trí đầu tiên, vậy nên giá trị của nó là 0 x 1 hoặc 0.

1 ở vị trí thứ hai, vậy nến giá trị của nó là 1 x 2 hoặc 2.

Khi bạn cộng hai giá trị lại (0 + 2), bạn được 2.

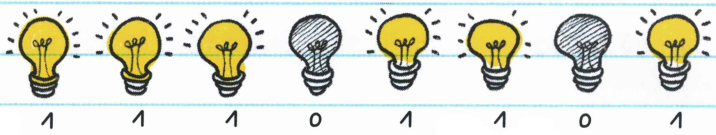
Bạn có thể sử dụng phép cộng cơ bản để biểu diễn các giá trị.

GIÁ TRỊ	Vị trí 128	Vị trí 64	Vị trí 32	Vị trí 16	Vị trí 8	Vị trí 4	Vị trí 2	Vị trí 1	
3							1	1	2 + 1
4						1	0	0	4 + 0
5						1	0	1	4 + 1
6						1	1	0	4 + 2

Một cách khác để hiểu số hệ nhị phân là cách máy tính làm: "bật" và "tắt".

VÍ DỤ. Nếu chúng ta có 8 bóng đèn xếp thẳng hàng (để biểu diễn 8 vị trí trong hệ nhị phân), chúng ta có thể sử dụng chúng để biểu diễn số theo hệ nhị phân bằng cách bật và tắt từng bóng đèn.

Bạn có thể biểu diễn số 237 bằng 1 hàng 8 bóng đèn. Đèn sáng thể hiện số 1, đèn tắt thể hiện số 0 trong bảng vị trí.



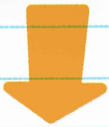
Tổng giá trị của bóng đèn là 237.

128 + 64 + 32 + 8 + 4 + 1

Biểu thị chữ cái bằng hệ nhị phân

Ngày trước, máy tính được dùng để thực hiện phép toán. Sau này, người ta phát triển thêm cách sử dụng hệ nhị phân để biểu diễn chữ cái và biểu tượng.

MÃ TRAO ĐỔI THÔNG TIN TIÊU CHUẨN MỸ (ASCII)[1] quy ước mỗi chữ cái trong bảng chữ cái và mỗi biểu tượng là một mã nhị phân riêng. Mỗi chữ cái được quy ước là một số, sau đó được mã hóa thành mã nhị phân.



Ví dụ, chữ A được quy ước là số 65, viết theo hệ nhị phân là 01000001.

Dù số 65 và chữ A có cùng mã nhị phân, chương trình sẽ cho máy tính biết nên diễn giải nó là chữ cái hay chữ số.

[1] American Standard Code for Information Interchange.

Bit và Byte

Bit là viết tắt của binary digit. (số nhị phân).

Thuật ngữ như "bộ nhớ 16 gigabyte" hoặc "ổ cứng 1 terabyte" được dùng để nói về khả năng lưu trữ của máy tính. Thuật ngữ này nói về số lượng số nhị phân.

BIT là đơn vị lưu trữ nhỏ nhất. Nó thể hiện 1 số (1 hoặc 0).

Đơn vị đo lớn hơn tiếp theo là **BYTE** - bằng với 8 bit. Ví dụ, 01010101 là 1 byte. (có 8 số).

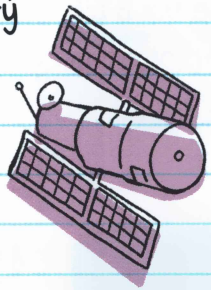
b = bit
B = byte

Số lớn nhất bạn có thể biểu diễn với 1 byte là 11111111, là số 255 trong hệ thập phân.

Đó là lý do tại sao giá trị RGB (màu) nằm trong khoảng từ 0 đến 255, vì đó là những số bạn có thể biểu thị bằng 1 byte.

Những giá trị lớn hơn của byte có thể được thể hiện bằng cách thêm tiền tố:

- **Kilo**byte (KB) - khoảng 1 nghìn byte (một trang tài liệu A5 chiếm khoảng 100KB).
- **Mega**byte (MB) - khoảng 1 triệu byte (các tài liệu âm thanh chiếm khoảng vài MB).
- **Giga**byte (GB) - khoảng 1 tỷ byte (phim độ phân giải cao chiếm khoảng vài GB).
- **Tera**byte (TB) - khoảng 1 nghìn tỷ byte (Một số ổ cứng đời mới có dung lượng khoảng 1-3 TB. Kính viễn vọng không gian Hubble gửi khoảng 10 TB dữ liệu xuống Trái Đất mỗi năm).



Kính viễn vọng không gian Hubble là kính viễn vọng chính thức đầu tiên được phóng vào không gian. Bay vào quỹ đạo Trái Đất vào năm 1990, nó là một trong những kính viễn vọng lớn nhất từng được chế tạo. Các nhà khoa học sử dụng nó để quan sát các ngôi sao và hành tinh ở xa.

TẠO RA THUẬT TOÁN

Trước khi lập trình, lập trình viên phải có được thuật toán hoặc danh sách các bước được sử dụng trong code. Họ thường dùng **CODE GIẢ** và **SƠ ĐỒ** để thiết kế thuật toán trước khi bắt tay vào lập trình.

SƠ ĐỒ
Biểu đồ vẽ ra từng bước của quy trình.

Code giả

Code giả được viết theo định dạng tương đồng với ngôn ngữ lập trình mà lập trình viên định sử dụng. Không có quy tắc viết code giả, nhưng code giả được viết để con người đọc chứ không phải máy tính.

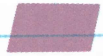
Hầu hết lập trình viên viết code giả sao cho mỗi dòng tượng trưng cho một dòng code thật trong chương trình. Code giả không cần phải hoàn hảo, nó chỉ cần cho thấy chương trình hoàn thiện sẽ trông như thế nào.

Viết code mà không có kế hoạch như xây nhà không có bản thiết kế.

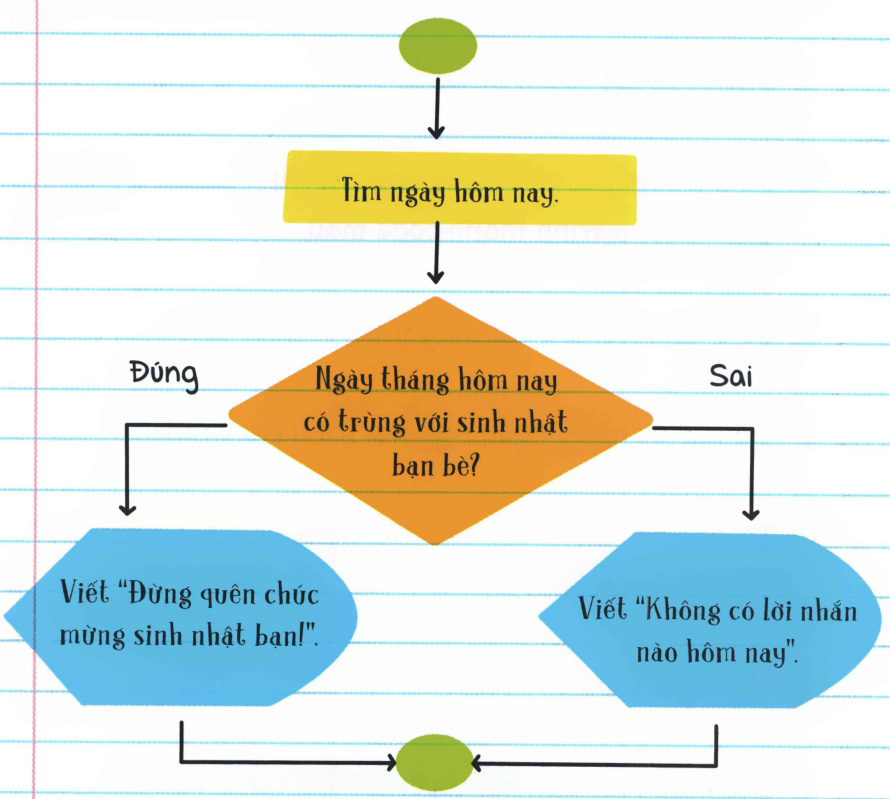
Sơ đồ

Sơ đồ giúp lập trình viên hình dung từng bước trong thuật toán. Giống như viết dàn ý trước khi viết văn, sơ đồ giúp sắp xếp ý tưởng, lập trình viên sử dụng những biểu tượng cụ thể để tượng trưng cho các phần thuật toán khác nhau.

Mỗi biểu tượng trong sơ đồ đều có ý nghĩa.

TÊN	BIỂU TƯỢNG	Ý NGHĨA TRONG SƠ ĐỒ
Hình bầu dục		Bắt đầu hoặc kết thúc chương trình.
Hình bình hành		Đầu vào.
Hình chữ nhật		Quá trình thực hiện (cộng, trừ, chia,...).
Hình thoi		Quyết định (hoặc phân nhánh): chương trình nên đi theo một trong hai đường.
Đa giác		Đầu ra.
Mũi tên		Hướng diễn biến của sơ đồ.

Nếu bạn muốn viết chương trình kiểm tra xem hôm nay có phải sinh nhật bạn bè, sơ đồ có thể sẽ trông như sau:



Có thể phiên bản code giả của chương trình sẽ trông như sau:

```
Tìm ngày hôm nay.
Nếu ngày tháng hôm nay trùng với sinh nhật bạn bè:
    Nếu vậy, hãy viết "Đừng quên chúc mừng
    sinh nhật bạn!"
    Nếu không, viết "Không có lời nhắn nào hôm nay."
```

KIỂM TRA KIẾN THỨC

1. Ngôn ngữ lập trình là gì?

2. Để máy tính đọc và chạy chương trình _____ cần dịch chương trình thành code máy.

3. Số 0 và 1 trong code máy thể hiện điều gì?

4. Tại sao có nhiều ngôn ngữ lập trình khác nhau?

5. Tại sao học một ngôn ngữ lập trình giúp học ngôn ngữ lập trình khác dễ hơn?

6. Một byte chứa bao nhiêu bit?

7. Nếu máy tính dịch hai chương trình cùng viết "Xin chào, Thế giới!", liệu code máy của cụm từ "Xin chào, Thế giới!" sẽ khác nhau?

8. Kilobyte hay gigabyte lớn hơn?

9. Làm thế nào để viết số 2 bằng hệ nhị phân? 101 biểu diễn số nào trong hệ nhị phân?

10. Nêu hai cách tốt để lên kế hoạch cho chương trình trước khi viết code.

ĐÁP ÁN

1. Ngôn ngữ lập trình là ngôn ngữ dùng để đưa chỉ dẫn cụ thể cho máy tính.

2. Bộ phiên dịch

3. Tắt và bật

4. Mỗi ngôn ngữ có thế mạnh riêng, và có thể được sử dụng để tạo ra các loại chương trình khác nhau.

5. Hầu hết ngôn ngữ lập trình sử dụng các cấu trúc tương đồng.

6. 8

7. Không, chúng trông giống nhau.

8. Gigabyte

9. 10, 5

10. Code giả và sơ đồ

Chương 13

TƯ DUY MÁY TÍNH

Khoa học máy tính tập trung vào sử dụng máy tính để tạo ra giải pháp cho vấn đề.

TƯ DUY MÁY TÍNH là quá trình suy nghĩ tạo ra giải pháp có thể thực hiện bởi máy tính.

VÍ DỤ, nếu vấn đề là bạn cần viết luận về sách, giải pháp đơn giản là đọc sách, viết dàn ý rồi viết về những điều bạn đã đọc.

Cách giải quyết vấn đề bằng tư duy máy tính là thu thập dữ liệu về số lần mỗi nhân vật nói chuyện với nhau và đưa thông tin lên biểu đồ. Bạn thậm chí có thể tạo ra chương trình phân tích dữ liệu hộ bạn. Biểu đồ sẽ cho bạn những chứng cứ dựa trên số liệu về mối quan hệ giữa các nhân vật trong truyện. Thông tin đầu ra của chương trình và dữ liệu có thể cho bạn biết nên viết gì trong bài luận.

Một ví dụ phóng đại là lập trình một chương trình trí tuệ nhân tạo (AI) có khả năng hiểu **ngôn ngữ tự nhiên**, phân tích cuốn sách, và viết luận cho bạn. Dù cách này không giúp bạn hiểu cuốn sách, nhưng nó cho thấy tư duy máy tính có thể tiến xa tới đâu.

Cách con người nói chuyện với nhau.

Trong khoa học máy tính, trí tuệ nhân tạo (AI) là cách lập trình siêu việt để máy móc và máy tính học một lượng lớn dữ liệu giúp con người nâng cao hiệu suất công việc. AI cho phép chương trình máy tính đưa ra dự đoán và quyết định vượt ra khỏi những gì chúng được lập trình.

Có 4 ý chính đối với tư duy máy tính:

1. **Phân nhỏ:** Chia vấn đề thành các phần đơn giản.

2. **Nhận diện khuôn mẫu:** Xác định điểm chung giữa những vấn đề khác nhau.

3. **Hình dung:** Tách những chi tiết quan trọng khỏi những chi tiết không quan trọng.

4. **Thiết kế thuật toán:** Tạo ra giải pháp với những bước đơn giản mà ai cũng hiểu.

PHÂN NHỎ

Trước khi làm việc với giải pháp, bạn cần hiểu mọi phần của vấn đề. Một khi bạn đã hiểu mỗi phần, bạn có thể chia vấn đề thành những công việc nhỏ. Phân nhỏ giúp vấn đề dễ giải quyết hơn.

VÍ DỤ, khi viết luận về sách, bạn nên chia bài tập thành những công việc nhỏ, đơn giản hơn:

1. Đọc sách.
2. Xác định nhân vật chính.
3. Xác định bối cảnh.
4. Phân tích ngữ khí, tình tiết và nhân vật trong sách.
5. Viết về ngữ khí, tình tiết và nhân vật trong sách.

Phân nhỏ giúp bạn biết nên bắt đầu từ đâu, và những công việc nào cần hoàn thành.

NHẬN DIỆN KHUÔN MẪU

Khuôn mẫu là những sự kiện lặp lại. Nếu bạn nhận ra mình từng đưa ra giải pháp cho vấn đề tương tự, bạn sẽ dễ dàng tạo ra giải pháp cho vấn đề khác.

VÍ DỤ, nếu bạn từng dùng chung một dàn ý để viết luận về sách, bạn có thể đưa dàn ý đó vào chương trình. Cuốn sách có thể sẽ khác, nhưng quá trình viết thì giống nhau.

HÌNH DUNG

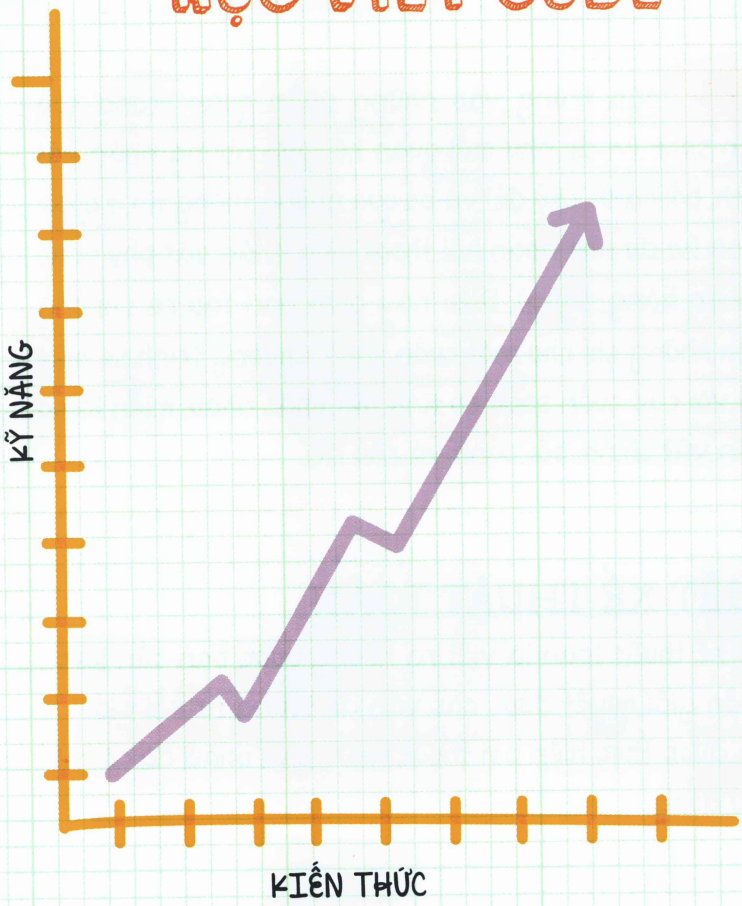
Hình dung là tập trung vào những ý quan trọng của vấn đề và bỏ qua những chi tiết không giúp bạn tìm ra giải pháp.

VÍ DỤ: tập trung vào những phần quan trọng của cuốn sách (ví dụ như mối quan hệ giữa những nhân vật chính, bối cảnh, chủ đề) và bỏ qua những thông tin ít quan trọng (ví dụ như đoạn hội thoại giữa nhân vật phụ hoặc sự kiện hàng ngày). Điều này giúp bạn tạo ra chương trình không chỉ cho một cuốn sách. Chương trình sẽ hiệu quả với các cuốn sách có tình tiết khác nhau, miễn là có nhân vật chính, bối cảnh và chủ đề.

THIẾT KẾ THUẬT TOÁN

Thiết kế thuật toán là viết ra những bước bạn cần làm để luôn giải quyết được các vấn đề tương tự. Khi giải pháp được thực hiện bởi thuật toán, giải pháp đó có thể tái sử dụng.

SƠ ĐỒ TIẾN TRÌNH HỌC VIẾT CODE



KIỂM TRA KIẾN THỨC

1. Tư duy máy tính là gì?

2. Tại sao tạo ra giải pháp có thể tái sự dụng là một phần quan trọng của tư duy máy tính?

3. Lập trình viên có dùng tư duy máy tính khi tạo ra chương trình?

4. Phần nào về tư duy máy tính nói về tập trung vào ý chính?

5. Giải thích phân nhỏ là gì?

6. Phần nào trong tư duy máy tính được cho là ý chính?

ĐÁP ÁN

ĐÁP ÁN

1. Một phương án giải quyết vấn đề bằng cách đưa ra thuật toán cho giải pháp có thể lập trình thành ngôn ngữ máy tính.

2. Nó giúp máy tính hiệu quả hơn. Máy tính có thể tái sử dụng thông tin/giải pháp đã có.

3. Đúng. Từ lên kế hoạch đến thử nghiệm, tư duy máy tính đều có thể áp dụng.

4. Hình dung

5. Phân nhỏ là chia công việc lớn thành những công việc nhỏ hơn.

6. Thiết kế thuật toán

BÀI 5

Nguyên tắc chung của lập trình

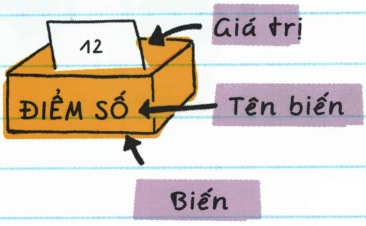
Chương 14

BIẾN

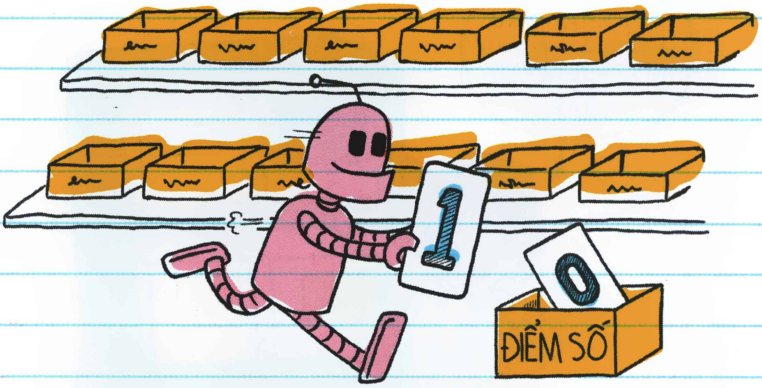
Trong toán học, **BIẾN** là chữ cái hay biểu tượng thể hiện một lượng chưa xác định. Biến đại diện cho một số trong biểu thức hoặc phương trình toán học. Trong lập trình máy tính, biến là nơi chứa giá trị. Biến được lưu trong bộ nhớ máy tính và có thể được dùng hoặc thay thế trong suốt chương trình. Biến đóng vai trò bổ sung hoặc đại diện cho các giá trị được lưu trữ trong máy tính.

Biến được biểu thị bằng **TÊN BIẾN**[1]. Thông tin mà biến chứa đựng gọi là **GIÁ TRỊ**. Giá trị của biến có thể là chữ, số, hoặc dạng dữ liệu khác.

Ví dụ, điểm số trong trò chơi điện tử là biến. Nếu trò chơi bắt đầu từ con số không và người chơi dành được điểm, đây là cách biến hoạt động:



[1] Identifier.

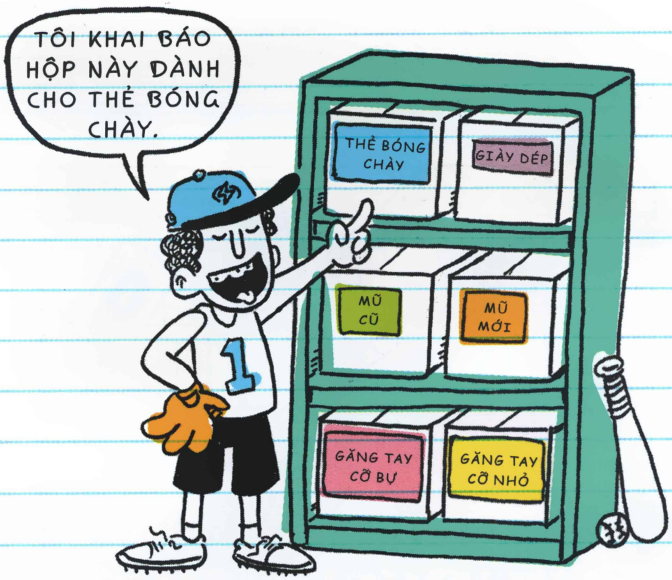


Chương trình tìm nơi lưu trữ giá trị của biến cho "điểm số", sau đó thay thế giá trị cũ là 0 với giá trị mới là 1.

GÁN GIÁ TRỊ VÀ ĐẶT TÊN CHO BIẾN

Để tạo ra biến trong chương trình, bạn phải **KHAI BÁO** nó. Khi bạn khai báo một biến, bạn ra lệnh cho máy tính tạo ra chỗ trống cho thông tin và đặt tên cho chỗ trống đó. Giống như sử dụng những chiếc hộp trên giá sách để sắp xếp đồ chơi vậy. Đối với mỗi loại đồ chơi, bạn chọn một hộp, dán nhãn nó, và đặt đồ chơi vào trong.

KHAI BÁO
Tạo ra biến trong chương trình bằng cách đặt tên cho nó.



Để gán giá trị cho biến, sử dụng dấu bằng (=). Tên biến nằm ở bên trái dấu bằng, và giá trị nằm ở bên phải. Bạn có thể gán giá trị mới cho biến bất kỳ lúc nào bạn muốn.

Dấu = được gọi là **TOÁN TỬ GÁN**[1] trong chương trình vì nó dùng để gán giá trị cho biến.

TOÁN TỬ GÁN
Dấu = dùng để gán giá trị cho biến.

[1] Assignment Operator.

Trong lập trình, dấu = không có có nghĩa là "bằng"; nó có nghĩa là "được gán cho".

name = "Jason" nghĩa là giá trị *Jason* được gán cho biến *name*.

Có vài quy tắc cơ bản cho việc đặt tên biến. Những ngôn ngữ lập trình khác nhau có thể có quy tắc khác nhau, nhưng các quy tắc cơ bản là:

- Tên biến nên ngắn gọn và diễn tả chính xác điều biến đại diện.

  Được: score

  Không được: this_variable_represents_player_1_score

  Quá dài!

- Biến phải bắt đầu với chữ cái.

  Được: firstPlayer

  Không được: 1stPlayer

- Biến có thể bao gồm chữ cái, con số, và một số ký tự như dấu gạch dưới. Những ký tự đặc biệt (#, @, &, %) hoặc dấu cách không được sử dụng. Khi bạn muốn dùng nhiều hơn một từ để đặt tên cho biến, bạn có thể sử dụng dấu gạch dưới (_) để thể hiện khoảng cách, hoặc viết hoa chữ cái đầu của mỗi từ.

  Được: first_player hoặc firstPlayer hoặc FirstPlayer

  Không được: first player hoặc first#player hoặc first@player

Một số từ vốn có nghĩa đặc biệt trong chương trình và không thể dùng để đặt tên. Những từ này tùy thuộc vào ngôn ngữ lập trình bạn sử dụng.

VÍ DỤ: Python dùng từ "return" để tạo ra giá trị của biểu thức hoặc hàm. Dưới đây là những từ bạn không thể dùng để đặt tên trong Python:

NHỮNG TỪ ĐẶC BIỆT TRONG PYTHON

true	for	false	import	and
not	if	or	else	return
none	while	elif		

NHỮNG DẠNG THÔNG TIN BIẾN CÓ THỂ CHỨA

Biến có thể chứa nhiều dạng thông tin (gọi là dạng dữ liệu).

Giá trị chuỗi[1]

Biến **CHUỖI** có thể chứa bất kỳ ký tự nào (bao gồm chữ cái, con số, biểu tượng và các ký tự đặc biệt). Ví dụ, trong chương trình đố vui, chúng ta có thể dùng biến chuỗi để chứa câu hỏi và câu trả lời trong chương trình.

Chuỗi luôn được đặt trong dấu ngoặc kép.

VÍ DỤ

Chuỗi

```
name = "Alan Turing"
fruit_salad = "Yummy yummy"
```

Tên biến

[1] String Values.

Giá trị số[1]

Biến có thể chứa **giá trị số**, bao gồm các số nguyên (số nguyên dương và số nguyên âm) hoặc số thực dấu phẩy động (số có dấu thập phân). Ví dụ, chúng ta có thể dùng biến để chứa số thời gian cần thiết để người chơi hoàn thành một màn trong trò chơi.

Không dùng dấu ngoặc kép với giá trị số.

VÍ DỤ:

Giá trị số

Tên biến

```
age = 13
xp = 245
score = 23.5
x_coordinate = -300.76
```

ĐIỂM CAO

XP = ĐIỂM KINH NGHIỆM

Dùng để thể hiện sự tiến bộ của nhân vật trong trò chơi điện tử.

[1] Numeric Values.

Bạn có thể sử dụng biểu thức đại số khi gán giá trị cho biến. Ví dụ, chúng ta có thể dùng biến để chứa điểm kinh nghiệm của người chơi và cộng 5 điểm vào giá trị mỗi lần họ đạt mục tiêu (score = x + 5).

Chúng ta cũng có thể sử dụng toán học để cộng điểm vào biến của điểm.

VÍ DỤ, để cộng 2 điểm vào giá trị chúng ta có thể sử dụng:

```
score = 5 + 2
```

Vì "score" là biến số nguyên, nó có giá trị là 7, tổng của biến số nguyên.

Nhưng nếu bạn tạo "score" là biến chuỗi bằng cách đặt 5 + 2 vào dấu ngoặc kép như sau:

```
score = "5 + 2"
```

Vậy "score" chứa biến chuỗi là "5 + 2", nội dung trong ngoặc kép.

Giá trị Boolean

Đôi khi lập trình viên cần biết điều gì đó là sai hay đúng. Ví dụ, trong trò chơi, họ muốn biết liệu trò chơi đã kết thúc hay vẫn tiếp tục. Hoặc, người chơi trả lời câu hỏi đúng không? Khi chúng ta muốn lưu thông tin đúng hoặc sai (có thể là "có" hoặc "không"), chúng ta dùng dạng dữ liệu khác gọi là **BOOLEAN.** Biến Boolean chỉ có hai giá trị: **đúng** hoặc **sai.**

VÍ DỤ:

```
I_like_licorice = False
powerUp = True
gameOver = True
```

Tên biến

Giá trị Boolean

Danh sách

Loại **BIẾN DANH SÁCH** này chứa cả danh sách thông tin. Nó hữu ích khi bạn có nhiều mảnh thông tin cần lưu ở một nơi. Ví dụ, trong trò chơi Hangman, bạn có thể dùng biến danh sách để lưu tất cả chữ cái mà người chơi đoán. Mỗi lần người chơi đoán một chữ cái mới, nó sẽ được thêm vào danh sách.

KIỂM TRA KIẾN THỨC

1. Trong khoa học máy tính, biến:

A. Là những số như số Pi

B. Thể hiện những phần được đo hoặc kiểm tra trong thí nghiệm

C. Là nơi điền thông tin

D. Không thể thay đổi

2. Sự khác biệt giữa tên biến và giá trị của biến là gì?

3. Khai báo biến là gì?

4. Biến nào sau đây là biến chuỗi? (có thể chọn nhiều hơn 1 đáp án.)

A. character = "Dr."

B. vehicle = "Ambulance"

C. speed = 37

D. lives = "3"

5. Giả sử điểm của người chơi bắt đầu từ 0 và tăng thêm 1 mỗi lần họ đạt mục tiêu, dạng biến nào phù hợp nhất cho tên biến "điểm"?

A. Biến số

B. Danh sách

C. Boolean

D. Chuỗi

6. Giá trị của "var_x" là gì sau khi câu lệnh này chạy:

```
var_x = 6 - 4
```

A. 10

B. 2

C. "6 - 4"

D. Không xác định

7. Giá trị của "var_x" là gì sau khi câu lệnh này chạy:

```
var_x = "6 - 4"
```

A. 10

B. 2

C. "6 - 4"

D. Không xác định

8. Toán tử gán là gì? Và được dùng để làm gì?

9. Phân loại các dạng biến sau:

A. adaLovelace = "First programmer"

B. Hotel_Floor = "13"

C. Jersey = 18

D. QueenOf = "Sheba"

E. I_love_Mom = True

10. Chọn tên biến không tuân theo quy tắc. Giải thích tại sao những tên biến đó sai:

A. "teacherNames"

B. RangersScore

C. School Assignment

D. My_Absolute_Best_Ever_Summer_Vacation_Do_You_Want_To_Hear_About_It

E. 6ofSpades

ĐÁP ÁN

1. C

2. Tên biến là tên được đặt cho biến và không đổi, nhưng giá trị (dữ liệu) chứa trong biến có thể thay đổi thường xuyên.

3. Khi bạn khai báo biến, bạn ra lệnh cho máy tính tạo ra chỗ trống cho thông tin và đặt tên cho chỗ trống.

4. A, B, D

5. A

6. B

7. C

8. Toán tử gán là dấu =. Nó được dùng để gán giá trị cho biến.

9. **A.** Chuỗi
B. Chuỗi
C. Số
D. Chuỗi
E. Boolean

10. **A.** Bắt đầu và kết thúc với dấu ngoặc kép
C. Có dấu cách trong tên
D. Quá dài
E. Bắt đầu bằng số

Chương 15

CÂU LỆNH CÓ ĐIỀU KIỆN

Câu lệnh điều kiện chạy một đoạn code chỉ khi một điều kiện nhất định được đáp ứng. Câu điều kiện tuân theo **ĐỊNH DẠNG "IF...THEN"** *(Nếu ... thì)*. Nó khiến chương trình linh động và hiệu quả hơn vì nó có thể thay đổi tùy vào điều kiện.

Ví dụ:

- **Nếu** bạn chạm vào kẻ thù, **thì** nhân vật của bạn sẽ chết.
- **Nếu** bạn chạm vạch đích trước, **thì** bạn thắng.
- **Nếu** bạn ghi điểm, **thì** điểm số của bạn sẽ tăng 1.

Câu điều kiện cơ bản trông như sau:

If điều cụ thể này xảy ra
Then điều này sẽ xảy ra tiếp theo

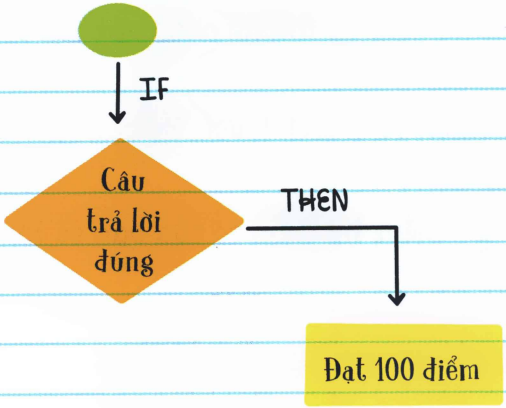
Câu lệnh điều kiện trong trò chơi đố vui có thể trông như sau:

điều kiện

If người chơi đưa ra câu trả lời đúng
Then người chơi đạt được 100 điểm

hệ quả (hành động)

Bạn có thể biểu diễn câu lệnh bằng sơ đồ. Ví dụ, trong trò chơi đố vui mà người chơi được 100 điểm khi trả lời đúng, câu lệnh điều kiện cơ bản có thể trông như sau:



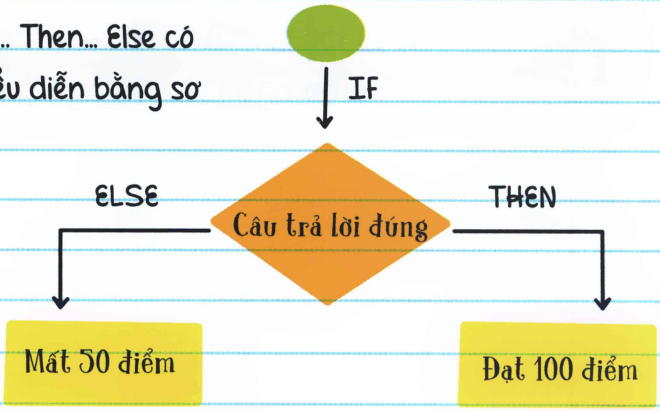
Bạn có thể đưa ra hành động khác trong trường hợp câu điều kiện không được đáp ứng. Để làm vậy, sử dụng **ELSE** (hoặc) để miêu tả hành động khác.

```
If điều cụ thể này xảy ra
    Then điều này sẽ xảy ra tiếp theo
Else
    Điều khác sẽ xảy ra
```

Câu lệnh điều kiện cơ bản có "else" trong trò chơi đố vui sẽ trông như sau:

```
If người chơi đưa ra câu trả lời đúng
    Then người chơi đạt được 100 điểm
Else
    Người chơi mất 50 điểm
```

Câu lệnh If... Then... Else có thể được biểu diễn bằng sơ đồ. Ví dụ:



Bạn có thể kết hợp nhiều câu lệnh điều kiện để tạo ra nhiều trường hợp hơn. Để làm vậy, sử dụng **ELSE IF** (nếu như) trong câu lệnh "if".

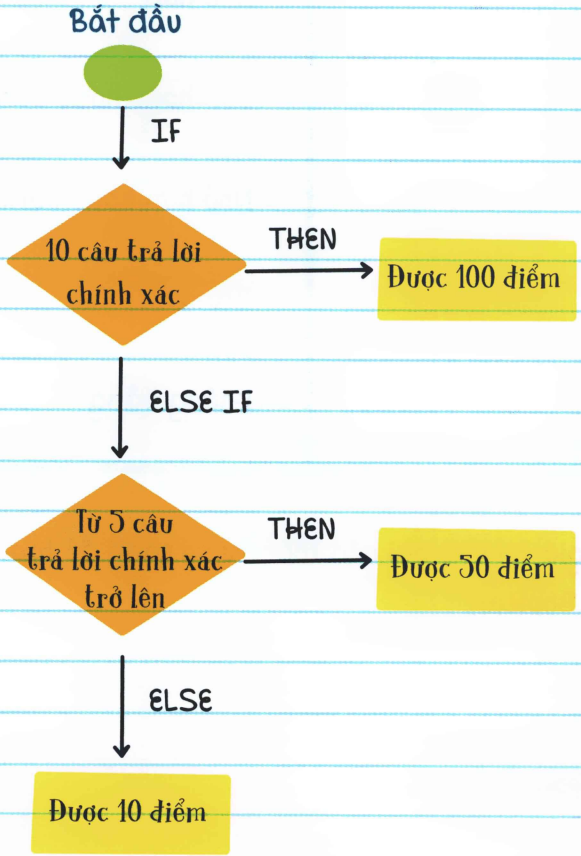
Mỗi "else if" thêm vào câu lệnh sẽ thêm một hành động khác, hoặc hệ quả, có thể xảy ra. Ví dụ, bạn có thể dùng câu lệnh "else if" để tạo ra ba trường hợp (ba hành động khác nhau) có thể xảy ra trong câu lệnh điều kiện:

```
If điều cụ thể này xảy ra
    điều này sẽ xảy ra
Else if điều khác xảy ra
    Điều khác sẽ xảy ra
Else
    Điều khác sẽ xảy ra
```

Dưới đây là ví dụ cụ thể của câu lệnh có "if" "else if" và "else" dùng để tính điểm người chơi trong trò đố vui:

If người chơi trả lời 10 câu chính xác
    Người chơi được thưởng 100 điểm
Else if người chơi trả lời chính xác từ 5 câu trở lên
    Người chơi được thưởng 50 điểm
Else
    Người chơi được thưởng 10 điểm

Sơ đồ sẽ trông như sau:



TOÁN TỬ SO SÁNH[1]

TOÁN TỬ SO SÁNH (ký tự dùng để so sánh giá trị) được dùng để tạo biểu thức Boolean.

TOÁN TỬ	MIÊU TẢ
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hoặc hoặc bằng
<=	Nhỏ hơn hoặc bằng
==	Bằng
!=	Không bằng

Đặt giá trị vào hai bên toán tử so sánh để tạo ra biểu thức Boolean.

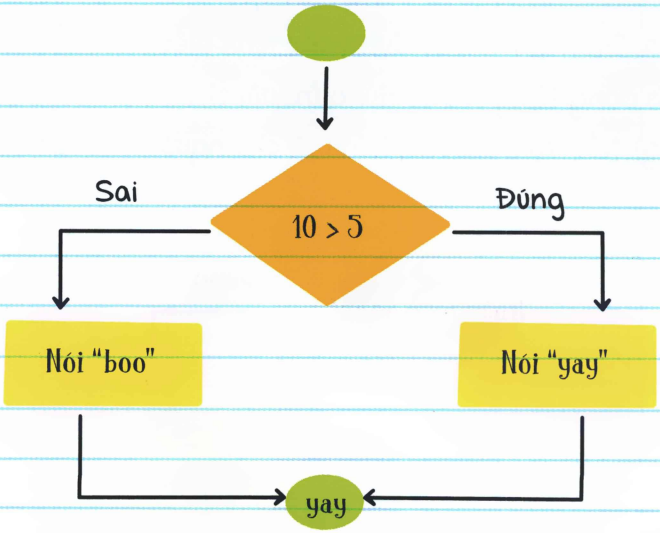
[1] Comparison Operator.

VÍ DỤ:

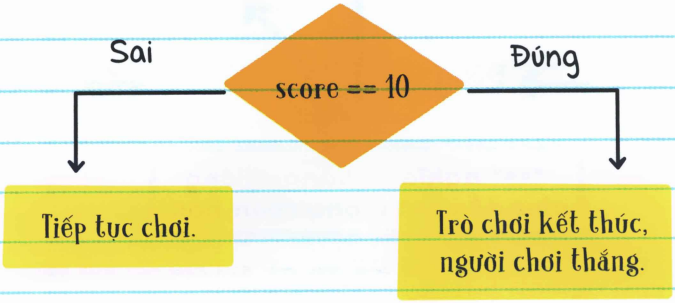
10 > 5 → Đúng	10 <= 5 → Sai
10 < 5 → Sai	5 == 5 → Đúng
5 >= 5 → Đúng	10 != 5 → Đúng

"==" nghĩa là bằng, giống như dấu "=" trong toán học.

Toán tử so sánh có thể dùng trong sơ đồ. Sơ đồ sau cho biết rằng nếu 10 lớn hơn 5, bạn nói "yay", nếu không bạn nói "boo". Bạn nói "yay" vì 10 > 5 là Đúng.

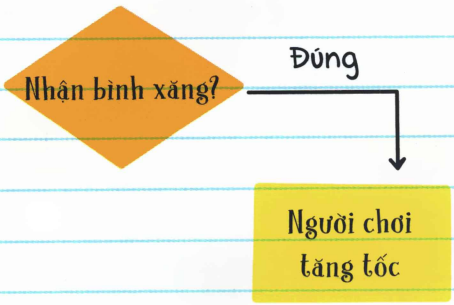


Bạn có thể dùng toán tử == để so sánh hai giá trị. Ví dụ, bạn có thể biết trò chơi đã kết thúc chưa dựa trên việc người chơi đã đạt 10 điểm chưa:



Bạn có thể sử dụng chính biểu thức Boolean.

Ví dụ, trong trò đua xe, bạn có biến "tăng tốc". Giá trị của biến được xác định là Đúng khi người chơi nhận được bình xăng. Chương trình sử dụng câu lệnh điều kiện để xem hành động tăng tốc có đúng (người chơi nhận được bình xăng). Nếu vậy, chương trình sẽ cho người chơi tăng tốc:



ĐIỀU KIỆN PHỨC HỢP

Để kiểm tra nhiều điều kiện có đúng trong một câu lệnh điều kiện, bạn có thể kết hợp biểu thức Boolean trong một câu lệnh điều kiện duy nhất.

ĐIỀU KIỆN PHỨC HỢP là những điều kiện phức hợp từ 2 biểu thức Boolean trở lên.

TOÁN TỬ LOGIC[1] được dùng để kết hợp biểu thức. Toán tử logic phổ biến nhất là AND, OR, và NOT (Và, Hoặc, Không).

Câu lệnh AND đúng khi các điều kiện đều đúng.

Đúng AND Đúng → Đúng

Ví dụ, nếu người chơi hoàn thành đường đua và (and) nhanh hơn tất cả đối thủ, họ là người thắng.

Nhưng nếu một hoặc các điều kiện đều sai, câu lệnh AND sẽ sai.

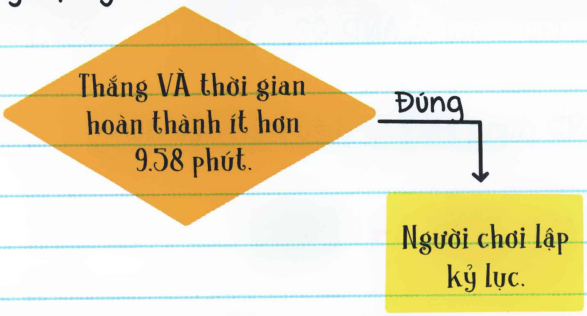
Đúng AND Sai → Sai

Sai AND Sai → Sai

[1] Comparison Operator.

VÍ DỤ: Người chơi lập kỷ lục nếu họ thắng **VÀ (AND)** hoàn thành đường đua nhanh hơn kỷ lục hiện tại.

Nếu người chơi hoàn thành đường đua nhưng không nhanh hơn kỷ lục hiện tại, hoặc không hoàn thành đường đua (cả hai điều kiện đều không đáp ứng), thì người chơi không lập kỷ lục mới.



Câu lệnh OR (hoặc) đúng khi ít nhất một điều kiện đúng.

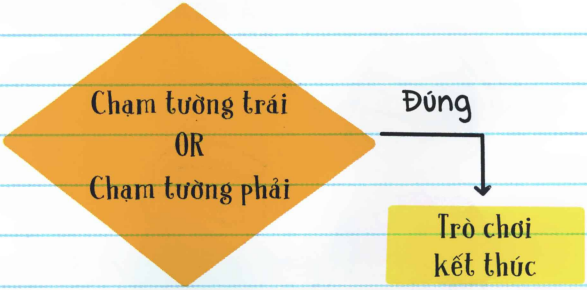
Sai OR Đúng → Đúng

Đúng OR Đúng → Đúng

Sai OR Sai → Sai

VÍ DỤ: Trong trò chơi bóng, nếu người chơi chạm tường phải **Hoặc** (OR) tường trái, họ sẽ thua và trò chơi kết thúc.

Nếu một trong hai điều kiện được đáp ứng, hệ quả là trò chơi kết thúc sẽ đúng.



Câu lệnh NOT (không) không so sánh hai điều kiện. Thay vào đó, nó đảo ngược giá trị Đúng thành Sai và Sai thành Đúng.

NOT Sai → Đúng

NOT Đúng → Sai

NOT đảo ngược giá trị.

VÍ DỤ: Trong trò đua xe, chúng ta có thể dùng điều kiện: Nếu trò chơi KHÔNG (NOT) kết thúc, hãy chơi nhạc nền. Nói cách khác, chơi nhạc nền khi trò chơi vẫn tiếp tục.

ĐIỀU KIỆN LỒNG GHÉP[1]

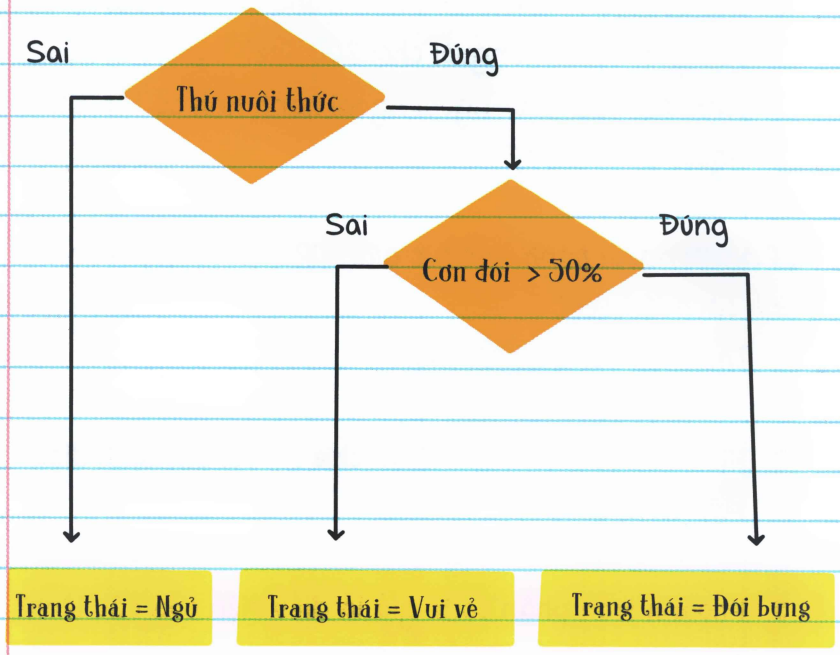
ĐIỀU KIỆN LỒNG GHÉP là điều kiện bên trong điều kiện khác. Ví dụ, trong trò nuôi thú ảo, bạn có thể lồng ghép điều kiện quyết định tâm trạng của vật nuôi dựa trên điều kiện khác như trạng thái đói hoặc mệt. Nếu vật nuôi không thức, hiển thị trạng thái đang ngủ.

[1] Nested Conditionals.

Nếu vật nuôi thức, và nếu trạng thái đói không vượt quá 50%, hiển thị tâm trạng vui vẻ, nếu không hiển thị trạng thái đói bụng.

Vậy, nếu vật nuôi thức và trạng thái đói ở mức 75%, tâm trạng sẽ là đói bụng.

Sơ đồ trông như sau:



KIỂM TRA KIẾN THỨC

1. Sự khác biệt giữa câu lệnh "if" và "else if"?

2. Giả sử người chơi hoàn thành trò chơi với 15 điểm, mệnh đề sau đây đúng hay sai?
   Người chơi > 20 AND Người chơi hoàn thành trò chơi

3. Giả sử người ngoài hành tinh trong trò chơi đang tức giận và cách người chơi 200 feet. Dựa trên câu lệnh điều kiện sau, người ngoài hành tinh có tấn công người chơi không?

If người ngoài hành tinh tức giận OR người ngoài hành tinh cách 100 feet
Then tấn công người chơi

4. Khi nào bạn dùng câu lệnh "else if" với câu lệnh điều kiện lồng ghép?

5. Khoanh tròn phần code giả là điều kiện của câu lệnh?

If đang là mùa hè
Then hiển thị "Mở cửa bể bơi!"

6. Giả sử người trả lời đúng 7 câu hỏi, họ được bao nhiêu vàng?

If người chơi trả lời đúng 10 câu
    Được 10 đồng vàng
Else if người chơi trả lời đúng từ 8 câu trở lên
    Được 4 đồng vàng
Else if người chơi trả lời đúng từ 5 câu trở lên
    Được 1 đồng vàng
Else
    Được 0 đồng vàng

7. Giả sử đang là mùa đông và màn hình chạy dòng code dưới. Màn hình sẽ hiển thị gì?

If đang là mùa hè
    Then hiển thị "Mở cửa bể bơi!"
Else
    hiển thị "Đóng cửa bể bơi"

ĐÁP ÁN

1. Câu lệnh "else if" luôn lồng ghép trong câu lệnh "if" và không tồn tại độc lập. Câu lệnh "if" có thể tồn tại mà không cần câu lệnh "else if".

2. Sai

3. Đúng, người ngoài hành tinh sẽ tấn công người chơi.

4. Nếu điều kiện phụ thuộc vào nhau, sử dụng câu lệnh "Else If" hoặc điều kiện lồng ghép.

5. Khoanh tròn phần code giả là điều kiện của câu lệnh:

If đang là mùa hè
Then hiển thị "Mở cửa bể bơi!"

6. Người chơi được 1 đồng vàng.

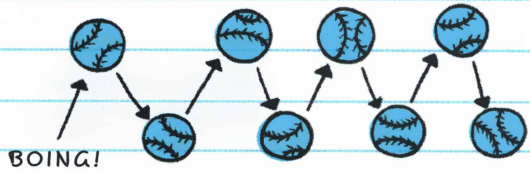
7. Màn hình sẽ hiển thị "Đóng cửa bể bơi".

Chương 16

VÒNG LẶP

CÂU LỆNH LẶP cho phép bạn lặp lại một đoạn code nhiều lần. Ví dụ, trong trò chơi nảy bóng, bạn sẽ tốn rất nhiều thời gian viết code để bóng nảy lên xuống cả nghìn lần (nảy lên, rơi xuống, nảy lên, rơi xuống...). Thay vào đó, bạn có thể lặp lại đoạn code:

```
Lặp lại 1000 lần
    nảy lên, rơi xuống
```



Vòng lặp không chỉ tiết kiệm thời gian, chúng còn khiến chương trình ngắn hơn, từ đó máy tính cũng dễ chạy chương trình hơn.

Khi tạo vòng lặp, bạn cần trả lời những câu hỏi sau:

- Tôi muốn lặp lại gì?
- Khi nào tôi muốn vòng lặp ngừng lại?

Có các dạng vòng lặp khác nhau:

VÒNG LẶP FOR[1] là dạng vòng lặp theo số lần. Vòng lặp *For* được dùng khi bạn biết chính xác số lần vòng lặp cần lặp lại. Vòng lặp *For* và danh sách thường được dùng cùng nhau.

VÍ DỤ: Ta có thể dùng vòng lặp *For* để liệt kê 10 người chơi đứng đầu với danh sách sẵn có.

danh sách chứa tên của 10 người chơi đứng đầu.

```
Top10 = ["player1", "player2", "player3", "player4",
"player5", "player6", "player7", "player8", "player9",
"player10"]
```

bắt đầu vòng lặp For, lặp lại code cho từng phần tử trong danh sách 10 người.

```
for item in Top10:
```

code lặp lại - đưa ra từng tên trong danh sách 10 người.

```
    print(item)
```

[1] For Loop.

VÒNG LẶP WHILE[1] sẽ chạy code đến khi điều kiện được đáp ứng. Nó được dùng khi bạn không biết chính xác số lần code lặp lại vì nó còn phụ thuộc vào yếu tố khác.

VÍ DỤ: Trong trò đố vui, bạn có thể liên tục hỏi lại một câu hỏi đến khi người chơi trả lời đúng:

```
while câu trả lời không đúng:
    Hỏi người chơi, "Tên của vị Phó Tổng thống đương
    nhiệm của Hoa Kỳ?"
in "bạn trả lời đúng rồi!"
```

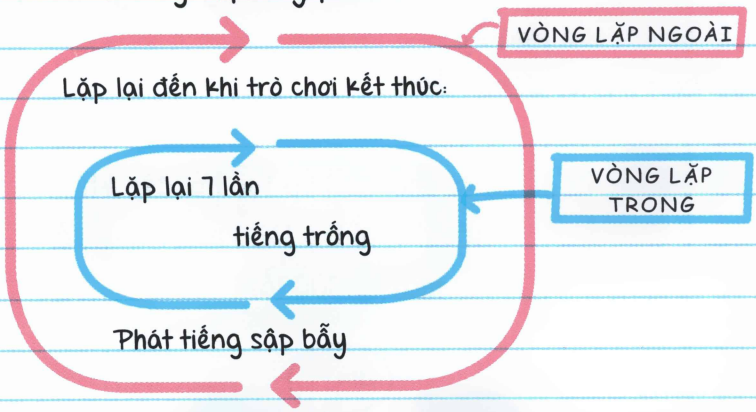
Trò chơi sẽ tiếp tục hỏi đến khi người chơi trả lời đúng, ngay cả khi mất tới 100 lần!

[1] While loop.

VÒNG LẶP LỒNG GHÉP[1] là một vòng lặp được đặt trong một vòng lặp khác. Vòng lặp lồng ghép tạo ra code lặp phức tạp hơn.

VÍ DỤ: Bạn có thể lập trình nhạc nền cho trò chơi bằng vòng lặp lồng ghép.

Vòng lặp lồng ghép sẽ lặp lại giai điệu đến khi trò chơi kết thúc. Vòng lặp ở trong sẽ lặp lại tiếng trống 7 lần trước khi tiếng sập bẫy phát ra:



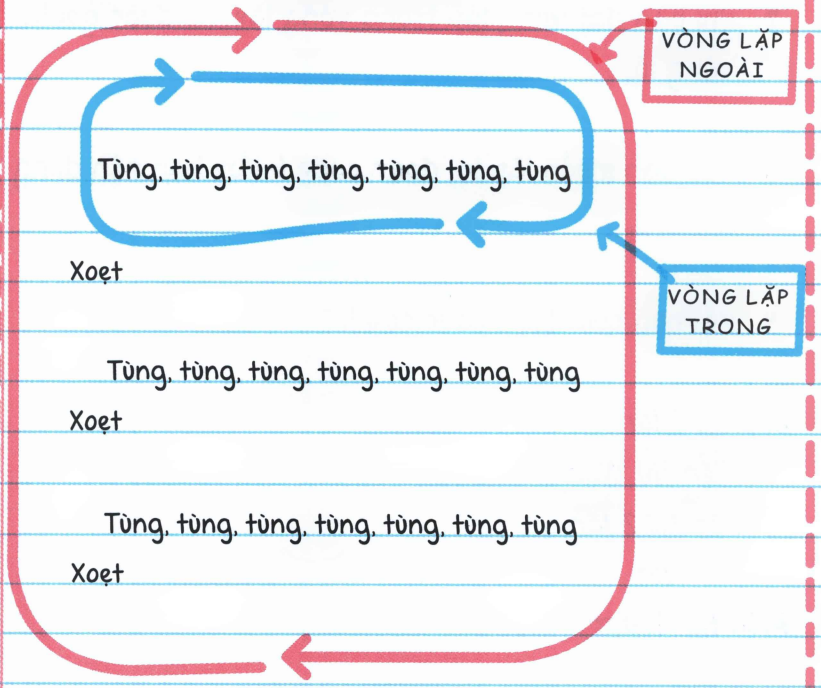
> Đối với code giả, sử dụng thụt lề để thể hiện code ở trong hoặc phụ thuộc vào code khác.

Hãy cẩn thận khi sử dụng vòng lặp lồng ghép, vì bạn sẽ dễ bị nhầm lẫn về đoạn mã cần lặp.

[1] Nested Loop.

Vòng lặp trong luôn phải chạy hết trước khi vòng lặp ngoài chạy tiếp.

VÍ DỤ, nhạc nền sẽ trông như thế này:



Tiếng trống phát ra 7 lần (hoàn tất vòng lặp) trước khi dòng code tiếp theo chạy và phát ra tiếng sập bẫy. Sau đó toàn bộ vòng lặp ngoài sẽ lặp lại đến khi trò chơi kết thúc.

KIỂM TRA KIẾN THỨC

1. Vòng lặp *While* khác gì với vòng lặp *For*?

2. Khi bạn đặt vòng lặp trong vòng lặp, nó được gọi là vòng lặp _____.

3. Nếu bạn muốn lặp lại đoạn code 6 lần, bạn nên dùng _____.

4. Kết quả của đoạn code sau là?

```
Lặp lại 3 lần:
    Hiển thị "Nhảy"
    Lặp lại 2 lần:
        "Né"
Hiển thị "Trượt"
```

5. Khi nào vòng lặp ngoài trong đoạn code giả sau dừng lại?

```
Khi số vỏ ốc quế >0:
    Lấy 1 vỏ ốc quế
    Lặp lại 3 lần:
        Thêm 1 muỗi kem
```

6. Tại sao thụt lề vòng lặp lồng ghép lại hữu ích khi viết code giả?

7. Nếu bạn muốn lặp lại nội dung danh sách, bạn nên dùng loại vòng lặp nào?

ĐÁP ÁN

1. Vòng lặp *While* sẽ lặp lại đến khi điều kiện được đáp ứng, nhưng vòng lặp *For* sẽ lặp lại số lần xác định.

2. Vòng lặp lồng ghép

3. Vòng lặp *For*

4. Nhảy, né, né, nhảy, né, né, nhảy, né, né, trượt

5. Vòng lặp sẽ ngừng khi có 0 vỏ ốc quế nào.

6. Thụt lề là cách tuyệt vời để xác định nơi vòng lặp bắt đầu và kết thúc. Giúp code dễ đọc và dễ hiểu hơn.

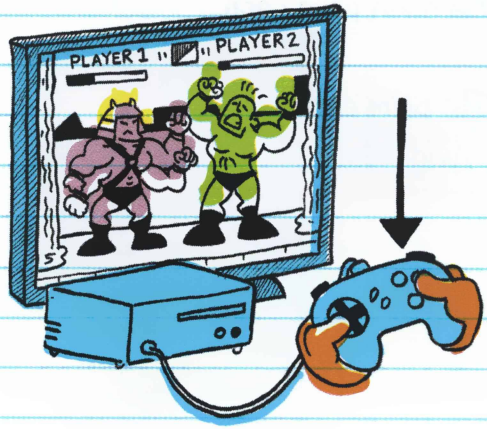
7. Vòng lặp *For*

Chương 17

SỰ KIỆN

SỰ KIỆN là hành động khiến điều gì đó xảy ra trong chương trình. Chúng khiến chương trình có tính tương tác vì người dùng được điều khiển. Ví dụ:

- Nhấn chuột.
- Ấn nút.
- Chạm vào màn hình.



Sự kiện cũng có thể là những điều xảy ra trong chương trình - ví dụ như khi nhân vật chết và dòng "Trò chơi kết thúc" xuất hiện.

Sự kiện có thể là hành động từ bên ngoài của người dùng, ví dụ như nhấn chuột. Chúng cũng có thể là hành động xảy ra bên trong chương trình, ví dụ như khi trình duyệt tải xong trang web.

BỘ XỬ LÝ SỰ KIỆN là code chạy khi sự kiện xảy ra. Nhấn phím cách trong trò chơi là một sự kiện, nhưng bộ xử lý sự kiện là code trong trò chơi khiến nhân vật nhảy khi người chơi nhấn phím cách.

VÍ DỤ, trong trò chém hoa quả, bạn có thể thêm code tạo hành động vung kiếm mỗi lần người chơi nhấn phím cách.

```
Nhấn phím cách:
    Vung kiếm
```

KIỂM TRA KIẾN THỨC

1. Cho ví dụ về sự kiện từ bàn phím.

2. Bộ xử lý sự kiện khác sự kiện như thế nào?

3. Ứng dụng yêu thích của bạn sẽ như thế nào nếu thiếu sự kiện?

4. Sự kiện trong đoạn code sau là gì?

```
Chạm vào màn hình:
    Nếu người chơi chạm vào bóng bay:
        Quả bóng vỡ
```

5. Bộ xử lý sự kiện trong đoạn code giả sau là gì?

```
Chạm vào màn hình:
    Nếu người chơi chạm vào bóng bay:
        Quả bóng vỡ
```

ĐÁP ÁN

ĐÁP ÁN

1. Hành động nhấn bất kỳ phím nào trên bàn phím máy tính đều có thể được lập trình thành sự kiện kích hoạt điều gì đó trong chương trình.

2. Bộ xử lý sự kiện là code chạy khi sự kiện xảy ra. Sự kiện là hành động.

3. Ứng dụng và tất cả chương trình tương tác sẽ vô dụng nếu không có sự kiện. Bạn sẽ không thể tương tác với nó. Chương trình sẽ trở thành những hình ảnh bạn có thể thấy nhưng không thể tương tác.

4. Sự kiện là hành động người chơi chạm vào màn hình khi chương trình đang chạy.

5. Bộ xử lý sự kiện là code kiểm tra xem bóng bay có bị chạm vào và nổ tung.

Chương 18

THỦ TỤC

THỦ TỤC

THỦ TỤC là đoạn code bạn có thể dễ dàng dùng lại nhiều lần. Vòng lặp rất hữu ích khi bạn muốn liên tục lặp lại một hành động. Nhưng khi bạn muốn dùng cùng một đoạn code ở các phần khác nhau của chương trình, bạn sẽ cần thủ tục.

Ví dụ, trong trò chơi 4 người, chúng ta lập trình mỗi nhân vật nhảy đơn khi mỗi người chơi nhấn nút nhảy. Nhưng cách tốt hơn là tạo ra thủ tục nhảy đơn cho mỗi nhân vật để dùng khi mỗi người chơi nhấn nút nhảy.

THỦ TỤC
Đoạn code có tên và dùng cho nhiệm vụ cụ thể.

Khai báo thủ tục

Thay vì sao chép và dán cùng một đoạn code trong suốt chương trình, lập trình viên có thể lưu đoạn code và đặt tên cho nó. Khi tên của đoạn code (thủ tục) được viết trong chương trình, máy tính sẽ hiểu cần dùng đoạn code đấy.

Để tạo ra một thủ tục, bạn phải khai báo nó bằng cách đặt tên cho nó và thêm đoạn code bạn muốn dùng vào thủ tục. (Tương tự như khai báo biến, bạn đặt tên và gán giá trị cho biến.)

Bạn có thể khai báo một thủ tục nhảy cho trò chơi 4 người bằng cách đặt tên cho thủ tục là "nhảy" và thêm vào nó đoạn code khiến nhân vật nhảy.

> Hàm cũng là một đoạn code. Mỗi hàm thực hiện một nhiệm vụ cụ thể, ví dụ như "hiển thị".

tên thủ tục

```
Chức năng nhảy
    Di chuyển lên
    Ngừng lại
    Di chuyển về phía mặt đất
```

code của thủ tục

BƯỚC 1: Xác định đoạn code cần lưu:

di chuyển lên, ngừng lại, di chuyển về phía mặt đất.

BƯỚC 2: Đặt tên cho đoạn code và lưu:

Nhảy = di chuyển lên, ngừng lại, di chuyển về phía mặt đất.

Gọi thủ tục

Khai báo thủ tục chỉ là đặt tên chứ không chạy code. Thủ tục sẽ không hoạt động nếu ra lệnh. Khi bạn muốn sử dụng thủ tục trong chương trình, bạn phải **GỌI** nó.

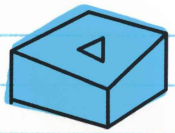
GỌI THỦ TỤC
Khi bạn gọi tên của thủ tục trong chương trình để chạy code.

Ví dụ, thủ tục có thể được gọi bao nhiêu lần tùy bạn.

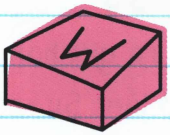
Khi người chơi 1 nhấn phím cách
Nhảy

tên thủ tục

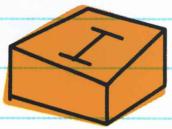
Khi người chơi 2 nhấn phím mũi tên
Nhảy



Khi người chơi 3 nhấn phím W
Nhảy



Khi người chơi 4 nhấn phím I
Nhảy



Thủ tục nhảy đều được gọi cho 4 sự kiện khác nhau. Khi máy tính đọc "nhảy", nó sẽ tìm hàm nhảy và chạy code máy tính tìm được.

Tham số và trả về

THAM SỐ là biến có giá trị được chuyển vào thủ tục. Nó đưa ra giới hạn hoặc khoảng cho thủ tục, ví dụ như "nhảy 3 lần". Sự khác biệt duy nhất giữa tham số và biến là giá trị tham số chỉ có thể được sử dụng trong thủ tục, nhưng biến có thể dùng trong suốt chương trình.

Đầu vào của tham số có thể thay đổi cách hàm hoạt động.

VÍ DỤ: Trong trò chiến tranh vũ trụ, chúng ta có thể dùng hàm để vẽ tia lase bắn ra từ súng tàu chiến. Vì chúng ta muốn tia lase bắn vào nơi người chơi nhắm chứ không phải luôn cùng một vị trí, chúng ta cần nhập vị trí người chơi nhắm đến (cho hàm lase). Do vậy, hàm lase có thể thay đổi vị trí bắn mỗi lần người chơi sử dụng súng tàu chiến.

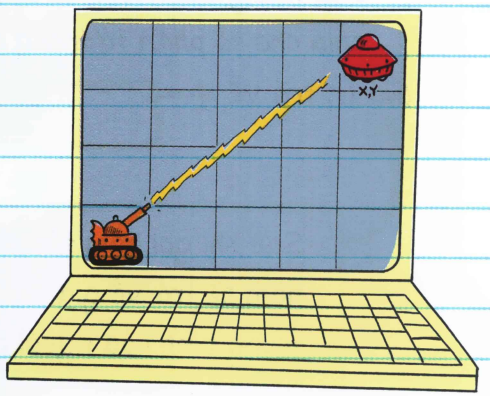
Tham số ***x*** và ***y*** cho hàm biết tọa độ ***x, y***, từ đó tia lase sẽ bắn về phía người chơi nhắm đến.

Hàm Bắn_Lase (x,y):

Hiển thị tia lase cạnh tàu chiến

Trong lúc đó trên màn hình:

Di chuyển về phía (x,y)



Một số thủ tục hoặc hàm có thể **TRẢ VỀ GIÁ TRỊ**. Giá trị trả về là đầu ra của thủ tục. Tùy thuộc vào thủ tục, bạn sẽ nhận được giá trị trả về khác nhau mỗi lần chạy thủ tục.

Tham số và giá trị trả về thường được dùng trong cùng một hàm.

VÍ DỤ: Nếu bạn tạo thủ tục tính tốc độ phản ứng trung bình của người chơi, tham số là số lần phản ứng của người chơi, và giá trị trả về là trung bình cộng tốc độ phản ứng của người chơi.

Đây là cách nhiệm vụ được hoàn thành

```
Hàm tốc_độ_trung_bình (danh_sách_số_lần_chơi):
    Tốc độ trung bình = tổng tốc độ các lần phản
    ứng trong danh_sách_số_lần_chơi
    chia cho số phần tử trong danh sách
Giá trị trả về
```

Đây là kết quả

Sau khi gọi hàm tốc_độ_trung_bình, hàm sẽ trả về giá trị là tốc độ trung bình của người chơi.

KIỂM TRA KIẾN THỨC

1. Đoạn code bạn có thể dễ dàng dùng lại nhiều lần gọi là _____.

2. Lợi ích của việc dùng tham số trong hàm?

3. Đầu ra của hàm gọi là giá trị _____.

4. Nếu bạn muốn dùng một đoạn code nhiều lần trong chương trình, bạn nên biến đoạn code thành _____.

5. Giá trị trả về của hàm sau đây là?

```
Điểm = 5
    Hàm điểm_thưởng (score):
        Điểm trả về + 10
```

ĐÁP ÁN

ĐÁP ÁN

1. Tham số

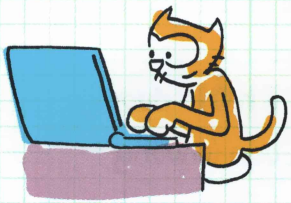
2. Nó cho phép bạn dùng một đoạn code nhiều lần, dù là với dữ liệu khác nhau.

3. Trả về

4. Tham số hoặc hàm

5. 15

BÀI 6



Lập trình bằng Scratch[1]

[1] (chú thích của người dịch) Hầu hết các từ trong chương 6 được dịch theo bản dịch tiếng việt của chương trình Scratch online.

Chương 19

KHỞI ĐỘNG

LÀM QUEN VỚI SCRATCH

Scratch là ngôn ngữ lập trình dạng đồ họa miễn phí.

- Scratch là ngôn ngữ lập trình dễ sử dụng vì bạn không phải gõ lệnh.
- Nó hoạt động bằng cách ghép **CÁC KHỐI**[1] được code sẵn lại với nhau để tạo ra kịch bản (chương trình).
- Scratch rất hữu ích khi làm trò chơi điện tử và câu chuyện có tính tương tác.

KHỐI

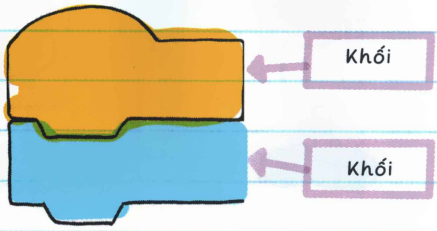
Khối trong Scratch là đồ họa đại diện cho đoạn code. Khối khớp với nhau tùy vào hình dạng của chúng (như trò xếp hình).

[1] Block.

KHỐI

Thay vì lập trình bằng cách gõ lệnh, chương trình tạo ra bởi Scratch bằng cách kết nối các khối theo chiều dọc. Người dùng chỉ phải ghép khối lại với nhau, và code sẽ chạy từ khối đầu (ở trên cùng) tới khối cuối (ở dưới cùng).

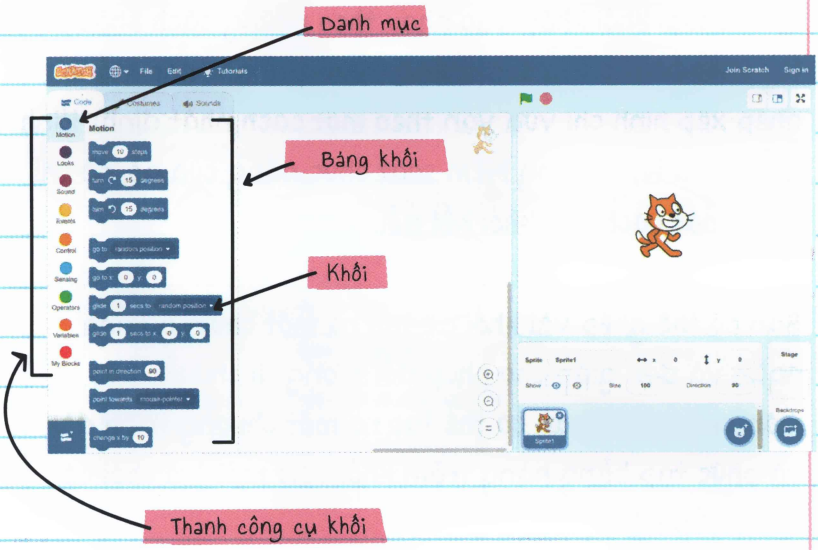
Khối trông như thế này



Khối khớp với nhau như trò xếp hình, các khối thực hiện hành động tương tự được xếp vào cùng một danh mục. Có sáu loại khối khớp vào nhau theo những cách khác biệt tùy theo hình dạng của chúng (tương tự như các mảnh ghép xếp hình chỉ vừa vặn theo một cách nhất định). Hình dạng của khối không kiểm soát chức năng của nó, nó chỉ kiểm soát cách các khối kết nối.

Bạn có thể ghép vài khối để tạo ra một chương trình ngắn và đơn giản cho phép đối tượng di chuyển qua màn hình. Hoặc bạn có thể tạo ra một chương trình dài và phức tạp bằng hàng trăm khối, như một trò chơi có nhiều cấp độ.

Thanh công cụ khối được chia thành 9 danh mục, mỗi danh mục có chấm màu và tên riêng.



Mỗi danh mục là các nhóm khối có chức năng tương tự nhau:

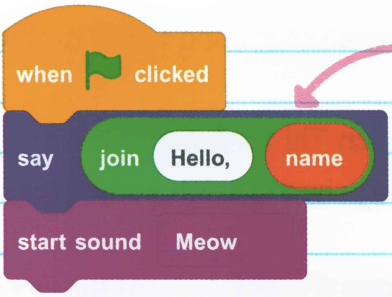
- **Các khối Chuyển Động, Hiển Thị** và **Âm thanh** điều khiển các đối tượng trong chương trình và hành động của chúng.
- **Các khối Biến** và **Toán Tử** lưu trữ và kiểm soát thông tin.
- **Các khối Sự Kiện** và **Cảm Biến** thực hiện các hành động cụ thể.
- **Các Khối Điều Khiển** chỉ đạo chương trình, kịch bản.
- **Khối Của Tôi** cho phép bạn tạo và lưu trữ khối tùy chỉnh riêng mình.

Kịch bản[1]

Một nhóm các khối được kết nối với nhau gọi là **KỊCH BẢN**. Một kịch bản có thể là chương trình đơn giản gồm 2 khối, hoặc một triệu khối (hoặc nhiều hơn). Bất kỳ số lượng kịch bản nào cũng có thể được sử dụng cùng nhau để thực hiện một dự án duy nhất. Dự án lớn như trò chơi nhiều cấp có thể sử dụng đến hàng tá kịch bản dài.

Khi tạo kịch bản, bạn có thể ghép các khối lại hoặc đặt một khối vào trong khối khác.

[1] Script.

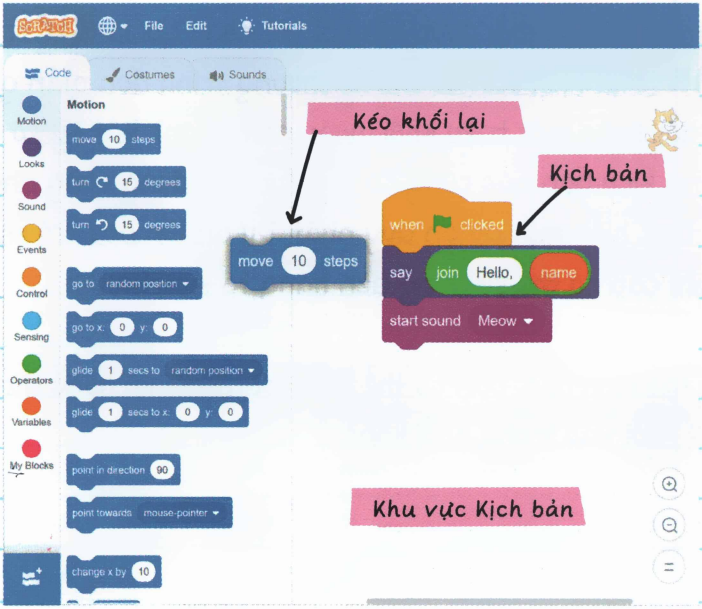


Khối xanh được đặt trong khối tím

Kịch bản được tạo trong **KHU VỰC KỊCH BẢN**.

KHU VỰC KỊCH BẢN
Vùng trên màn hình nơi khối được kết hợp để tạo ra kịch bản.

Sử dụng các khối bằng cách kéo chúng lại từ bảng Khối vào Khu vực Kịch bản.

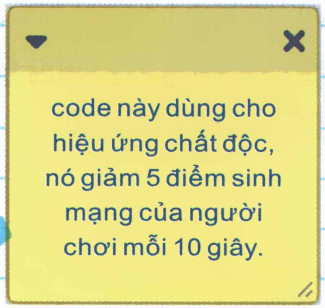


Bạn có thể thêm **BÌNH LUẬN** bằng cách nhấn chuột phải (hoặc giữ Ctrl và nhấn đối với máy Mac) vào khối hoặc vào khu vực kịch bản và chọn "Add Comment" (thêm bình luận). Bình luận giúp lập trình viên theo dõi các đoạn mã cụ thể làm gì và chia sẻ với người khác dễ hơn.

Bình luận không được chạy cùng code.

VÍ DỤ, nếu bạn viết đoạn code từ từ trừ đi điểm sinh mạng của đối tượng trong trò chơi, bạn có thể bình luận về nó như một lời nhắc cho bản thân rằng "code này dùng cho hiệu ứng chất độc, nó giảm 5 điểm sinh mạng của người chơi mỗi 10 giây".

Bình luận trong Scratch trông như sau.



Nếu bạn dùng phiên bản Scratch trên mạng, bạn có thể lưu trữ kịch bản trong khu vực Ba lô.[1]

Lưu trữ kịch bản rất hữu ích vì bạn có thể tiết kiệm thời gian và công sức bằng cách sử dụng cùng kịch bản cho nhiều hơn một dự án.

[1] Backpack.

VÍ DỤ, bạn muốn tạo ra trò chơi có chú kangaroo nhảy. Nếu bạn đã làm trò chơi có chú thỏ nhảy, bạn có thể mở trò chơi con thỏ, kéo kịch bản nhảy từ trò chơi con thỏ vào ba lô, mở trò chơi mới, và thả kịch bản nhảy từ ba lô vào trò kangaroo.

MẸO CODE TUYỆT VỜI!

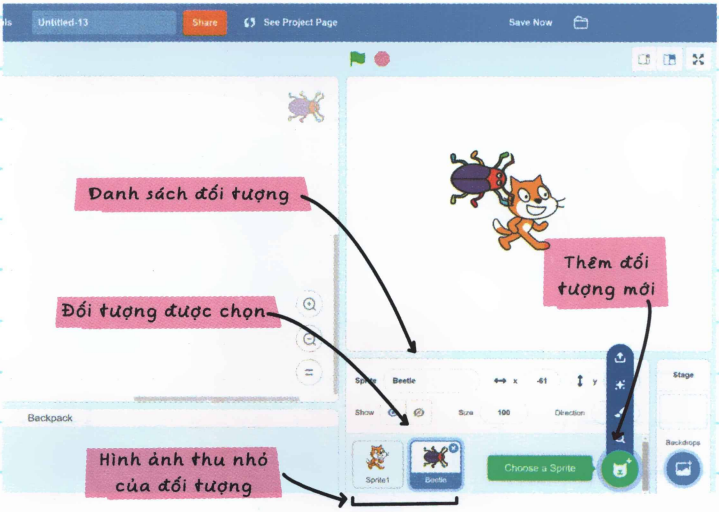
ĐỐI TƯỢNG[1]

Nhân vật hay đối tượng trong chương trình được gọi là **SPRITE**. Bạn có thể điều khiển đối tượng bằng khối.

Chương trình có thể sử dụng một hoặc nhiều đối tượng. Scratch có thư viện đối tượng. Bạn cũng có thể vẽ và đăng tải đối tượng riêng của mình.

DANH SÁCH ĐỐI TƯỢNG bao gồm những đối tượng được dùng trong chương trình. Mỗi đối tượng được thể hiện bằng **HÌNH ẢNH THU NHỎ.**[2]

HÌNH ẢNH THU NHỎ
Hình ảnh nhỏ hơn của hình ảnh gốc.



[1] Sprite (cuốn sách này sẽ dịch là đối tượng - chú thích của Dịch giả/BTV).
[2] Thumbnail image.

Khi bạn sử dụng nhiều đối tượng trong chương trình, nhấn vào hình ảnh thu nhỏ để chọn đối tượng bạn muốn viết kịch bản, thay đổi hình ảnh, hoặc thêm âm thanh.

Bạn có thể bổ sung đối tượng mới vào danh sách bằng cách nhấn nút " New Sprite" (thêm đối tượng mới). Tại đây, bạn có thể Choose a Sprite (chọn), Paint (vẽ), Surprise (tạo ngẫu nhiên) (một đối tượng ngẫu nhiên từ thư viện được chọn cho bạn), hoặc Upload Sprite (tải đối tượng) lên.

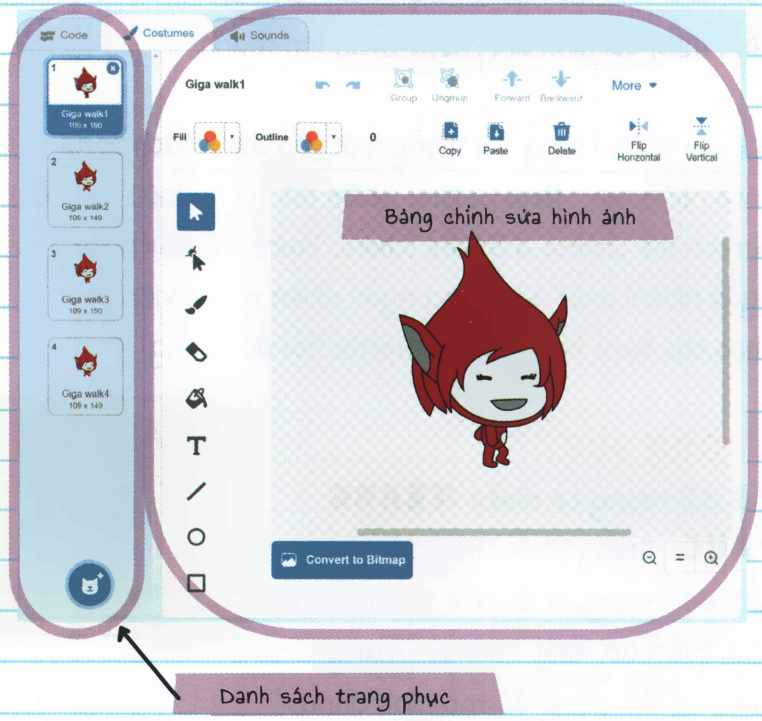
Trang phục

Mỗi đối tượng có nhiều **TRANG PHỤC**. Trang phục thay đổi hình ảnh đối tượng, ví dụ như vị trí chân tay, quần áo, hoặc kích cỡ. Trang phục có thể dùng để tạo hình động, như cách tạo hoạt hình bằng cách lật nhanh trang sách.

TRANG PHỤC
Những hình ảnh khác nhau dùng để thay đổi ngoại hình của một đối tượng.

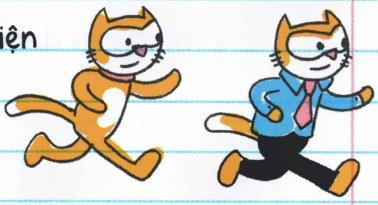
VÍ DỤ, dưới đây là 4 trang phục có thể dùng để khiến đối tượng trông như đang chuyển động:

Trang phục được chỉnh sửa trong tab trang phục.



DANH SÁCH TRANG PHỤC hiển thị hình thu nhỏ của trang phục đối tượng. Ấn nút tạo đối tượng, bạn có thể:

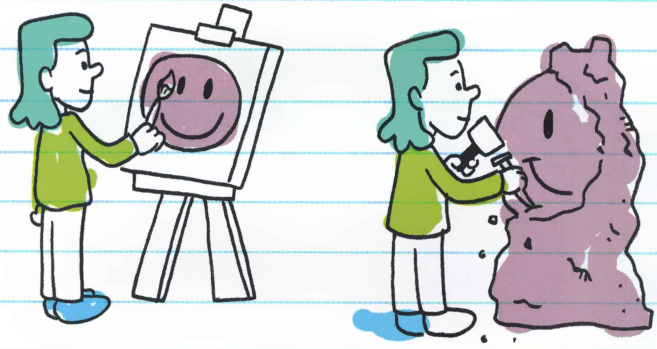
- Chọn trang phục trong thư viện
- Vẽ hoặc tải hình ảnh
- Chụp ảnh



Bạn có thể xem trang phục của đối tượng khác bằng cách chọn đối tượng trong danh sách đối tượng.

BẢNG CHỈNH SỬA HÌNH ẢNH có công cụ để vẽ trang phục mới hoặc chỉnh sửa trang phục của đối tượng có sẵn. Trang phục có thể tạo bằng chế độ **BITMAP** hoặc **VECTOR**. Cả hai chế độ đều có thể tạo ra cùng một hình ảnh, nhưng cách thực hiện khác nhau.

Chế độ bitmap: Công cụ dễ nhận biết hơn (Cọ vẽ, đường kẻ, hình tròn, chữ, tô màu, tẩy và những công cụ khác). Nhưng hình ảnh có thể bị nhòe khi phóng to lên.



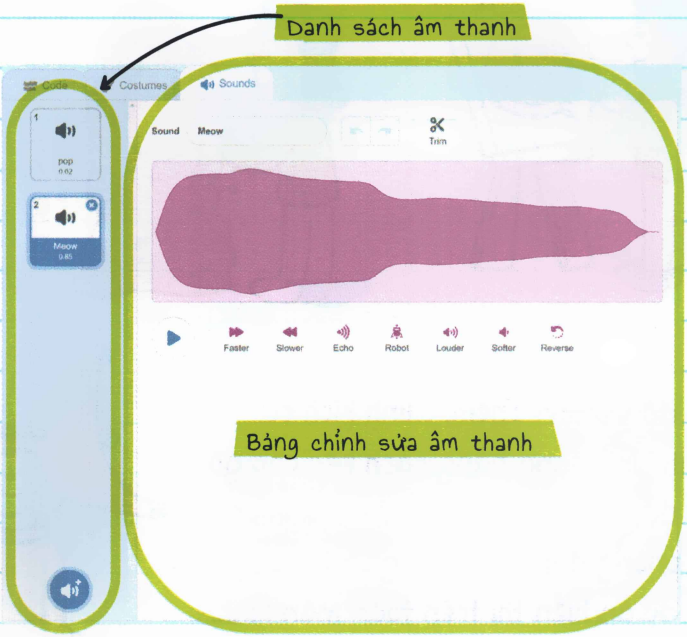
Chế độ vector: Thêm, chỉnh kích cỡ, chỉnh hình dạng bằng cách kéo các góc. Chế độ vector tạo ra các hình ảnh có thể phóng đại mượt mà, giúp ích cho các dự án hiển thị trên toàn màn hình.

Âm thanh

Bạn có thể lập trình đối tượng phát ra âm thanh bằng khối trong danh mục âm thanh.

Danh sách âm thanh hiển thị các âm thanh có sẵn cho đối tượng. Bạn có thể tải nhạc lên hoặc thêm âm thành từ thư viện của Scratch. Hoặc bạn có thể tự ghi âm.

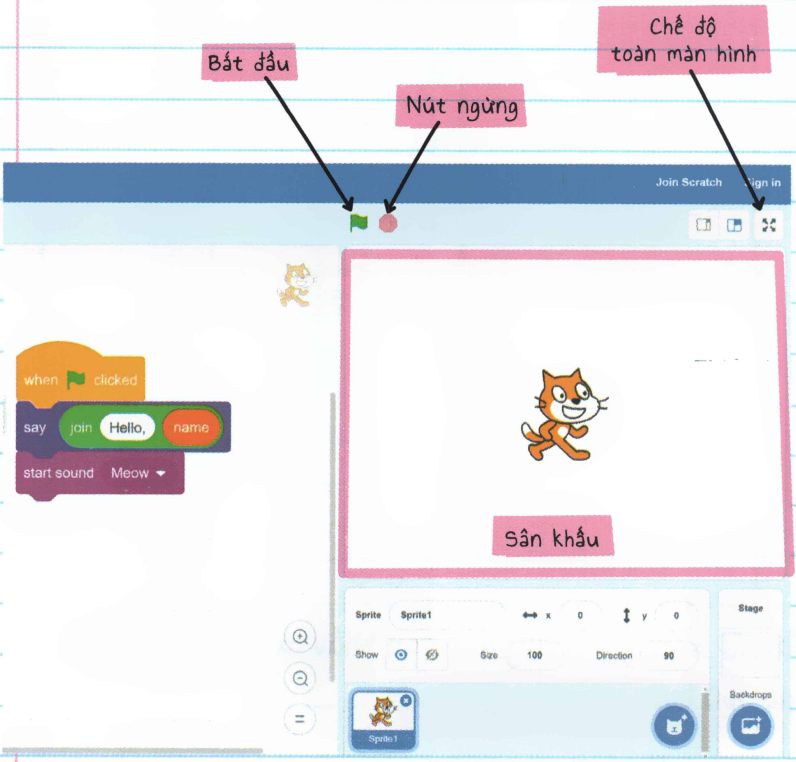
Bảng chỉnh sửa âm thanh cho phép bạn tùy chỉnh âm thanh theo ý mình.



Hình dáng trên màn hình thể hiện âm thanh. Nó thay đổi khi âm thanh thay đổi.

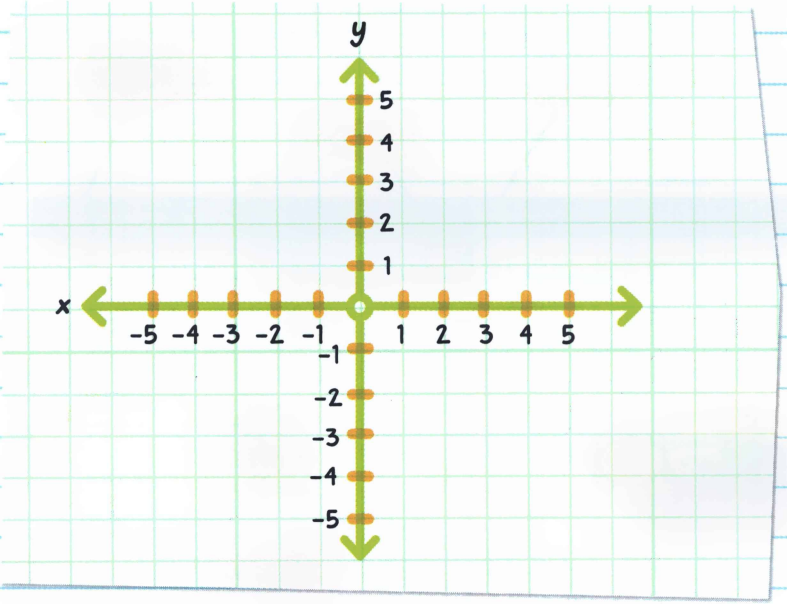
Cài đặt sân khấu

SÂN KHẤU[1] là nơi bạn thấy thành quả khi chạy kịch bản. Biểu tượng cờ xanh và nút tròn hồng lần lượt biểu thị bắt đầu và tạm dừng chương trình.



[1] Stage.

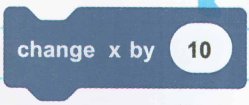
Đối tượng di chuyển quanh sân khấu bằng **hệ thống lưới** - tương tự như hệ thống tọa độ trong toán học. Tọa độ *x* và *y* được dùng để xác định vị trí.



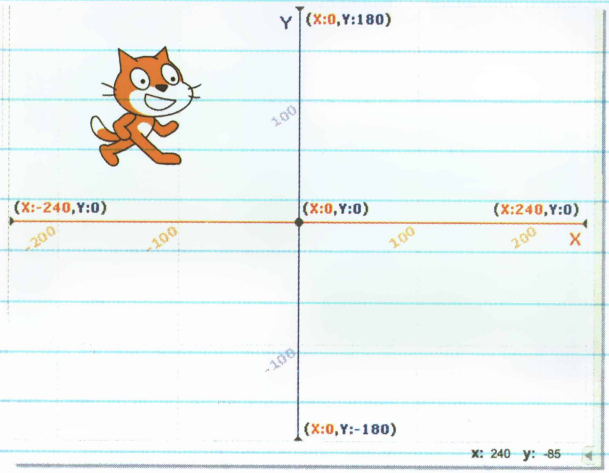
Tọa độ *x* và *y* ở trung tâm sân khấu là (0.0). Sân khấu cao 360 bước và rộng 480 bước.

(x, y)

KHỐI CHUYỂN ĐỘNG di chuyển đối tượng. Chúng cho phép bạn thay đổi vị trí tọa độ *x* và *y*.

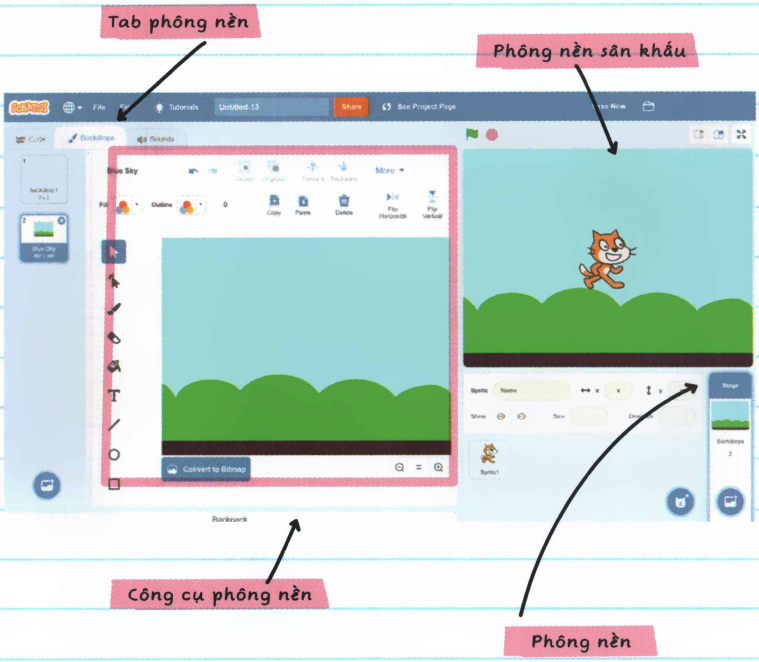
VÍ DỤ, khối 

Khối chuyển động sẽ di chuyển đối tượng 10 bước theo trục *x* (10 bước về phía bên phải). Bạn có thể sử dụng khối tương tự nhưng đổi 10 thành -10 để di chuyển đối tượng -10 bước theo trục *x* (10 bước về phía tay trái). Khối "change y by ( )" di chuyển đối tượng lên và xuống theo trục *y*.



Nền sân khấu được gọi là **PHÔNG NỀN**[1]. Chọn hình thu nhỏ của sân khấu để tạo kịch bản cho phông nền và bật tab phông nền. Bạn có thể chỉnh phông nền bằng công cụ phông nền trong tab phông nền. Bạn có thể thêm kịch bản, phông nền, và âm thanh vào phông nền.

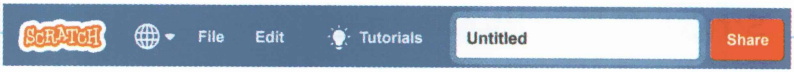
> Mẹo: Thêm nhạc nền chủ đề vào phông nền để phát nhạc trong khi người dùng chơi trò chơi của bạn!



[1] Backdrop.

Chia sẻ dự án

Phiên bản Scratch trên mạng cho phép người dùng xuất và chia sẻ *(share)* dự án với cộng đồng. Khi dự án được chia sẻ, bất kỳ ai trên thế giới lên trang Scratch đều có thể thấy nó.



Khi dự án được chia sẻ, thành viên cộng đồng có thể để lại bình luận (câu hỏi, phản hồi tích cực) trên trang dự án. Mỗi dự án có trang dự án riêng, nơi chủ sở hữu thêm chỉ dẫn, ghi chú, và nguồn tham khảo[1].

Phần mở rộng

PHẦN MỞ RỘNG

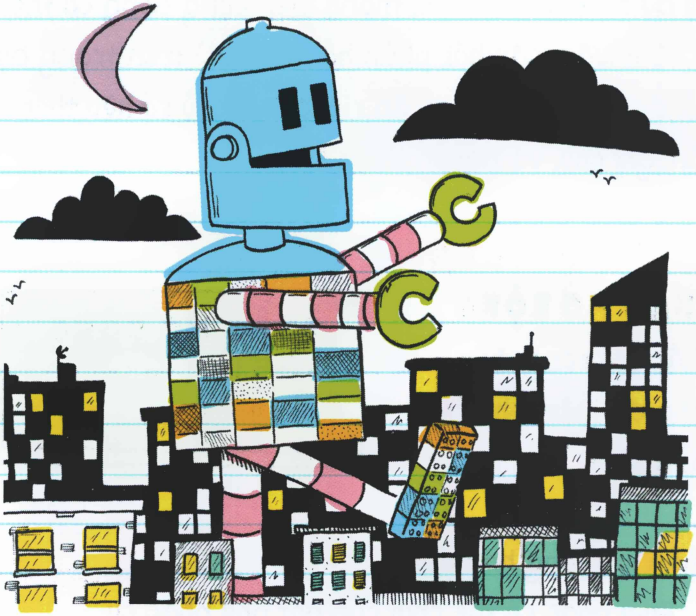
Phần mở rộng được dùng để thêm khối vào bảng khối.

Phần mở rộng bao gồm:

- Khối nhạc có thể chơi các nốt như nhạc cụ.
- Khối bút vẽ có thể dùng để vẽ trên sân khấu.
- Khối thêm khả năng tương tác, ví dụ như cảm biến video, đọc chữ và dịch.
- Khối cho phép Scratch lập trình những thiết bị khác như micro:bit, LEGO Mindstorms EV3, và LEGO WeDo 2.0.

[1] Tiếng Anh: *credit*.

VÍ DỤ, bạn có thể xây rô-bốt lego bằng bộ Mindstorms EV3, kết nối nó với máy tính và sử dụng Scratch để lập trình nó di chuyển.



KIỂM TRA KIẾN THỨC

1. Kịch bản trong Scratch là gì?

2. Khối trong Scratch được dùng làm gì?

3. Khu vực kịch bản dùng làm gì và dùng như thế nào?

4. Nếu bạn muốn lưu đoạn code để sử dụng cho dự án khác, bạn nên sử dụng chức năng nào của Scratch?

5. Bình luận nào sau đây không phù hợp để đăng tải lên dự án?

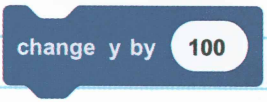
   A. Dự án tuyệt vời!

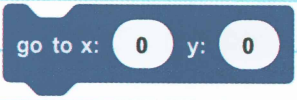
   B. Bạn khiến con mèo ngồi lên cây xương rồng như thế nào?

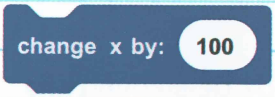
   C. Dự án hay. Nhưng tôi tìm thấy lỗi ở màn 2 của trò chơi.

   D. Đúng là phí thời gian, bạn nên bỏ code đi.

6. Giải thích cách đối tượng di chuyển quanh sân khấu đối với từng khối sau:

A. 

B. 

C. 

7. Tổ hợp khối chạy như một phần chương trình được gọi là _____.

8. Giải thích trang phục là gì?

9. Sân khấu dùng để làm gì?

10. Bạn gọi hình nền trên sân khấu là gì?

11. Bạn có thể tham gia cộng đồng Scratch bằng cách nào?

ĐÁP ÁN

1. Kịch bản là tổ hợp khối được gắn kết với nhau. Kịch bản có thể bao gồm vài khối hoặc rất nhiều khối. Các kịch bản có thể kết hợp để tạo ra chương trình.

2. Mỗi khối là một đoạn code. Thay vì gõ ra ký tự, Scratch sử dụng khối để chỉ dẫn chương trình.

3. Khu vực kịch bản là nơi chương trình được tạo ra. Khối được kéo từ bảng khối tới khu vực kịch bản và ghép với nhau để tạo ra kịch bản.

4. Ba lô

5. D

6. **A.** Đối tượng di chuyển lên trên 100 bước.
   B. Đối tượng di chuyển tới trung tâm màn hình.
   C. Đối tượng di chuyển sang phải 100 bước.

7. Kịch bản

8. Trang phục là các phiên bản hình ảnh khác nhau của đối tượng.

9. Là nơi bạn xem code chạy.

10. Phông nền

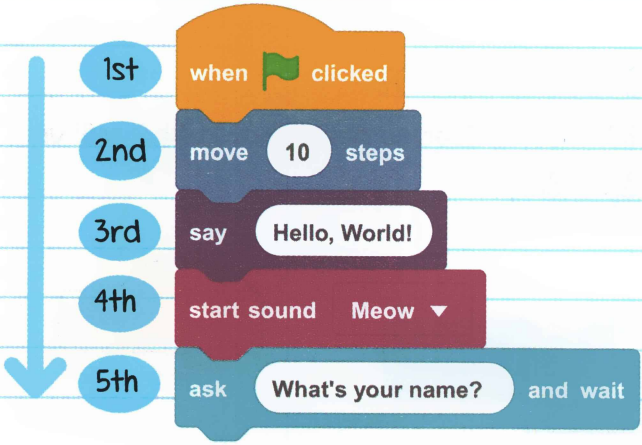
11. Bạn có thể chia sẻ kịch bản và bình luận vào kịch bản của những thành viên khác.

Chương 20

THUẬT TOÁN CƠ BẢN

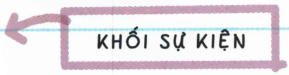
Khi bạn chạy một chương trình trong Scratch, mỗi khối hoàn thành nhiệm vụ của nó trước khi chuyển sang khối tiếp theo trong kịch bản. Kịch bản chạy từ khối trên cùng đến khối cuối cùng.

> Thuật toán là một danh sách các bước để hoàn thành một nhiệm vụ.



Có hai cách để chạy kịch bản:

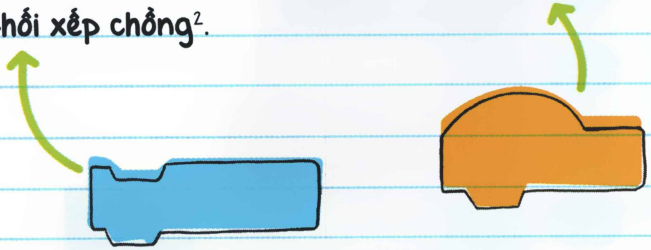
- Nhấn vào bất kỳ khối nào trong kịch bản (việc này sẽ chạy cả kịch bản từ đầu).
- Sử dụng khối sự kiện.



Khi kịch bản chạy, tất cả khối trong kịch bản đều phát sáng.

CÁC LOẠI KHỐI

Loại khối thường được sử dụng nhất là **khối mũ**[1] và **khối xếp chồng**[2].



[1] Hat block.
[2] Stack Block.

KHỐI MŨ bắt đầu kịch bản mà nó được gắn vào. Nó không có khớp ở trên và không thể nằm dưới khối khác. Khối mũ là **bộ xử lý sự kiện** trong Scratch - chúng được gắn kết với sự kiện và chạy khi sự kiện xảy ra. Ví dụ, **"nhấn cờ xanh"** là một trong chín khối sự kiện, khối mũ sẽ chạy code gắn liền với nó khi người dùng nhấn vào cờ xanh.

Ví dụ về sự kiện là khi người dùng nhấn phím trên bàn phím hoặc khi nhấn vào đối tượng.

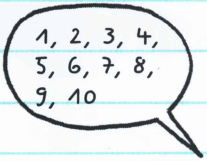
KHỐI XẾP CHỒNG là khối hình chữ nhật, kết nối các khối với nhau.

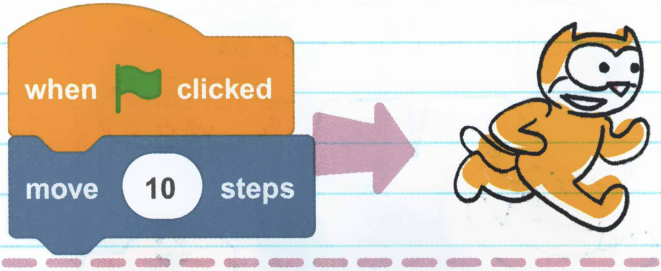
TẠO RA CHÚ MÈO HOẠT HÌNH

Đối tượng mặc định cho mỗi dự án mới là **MÈO SCRATCH**. Bạn có thể dùng khối trong danh mục Chuyển Động để lập trình đối tượng di chuyển quanh sân khấu.

Bước đi

Kịch bản này khiến Mèo Scratch di chuyển 10 bước khi nhấn vào cờ xanh.



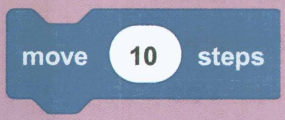


Một số khối có khoảng trống để bạn điền số hoặc chữ vào.

Khoảng trống này được gọi là **THAM SỐ**. Nhấn vào chúng để thay đổi hoặc thêm số hoặc chữ.

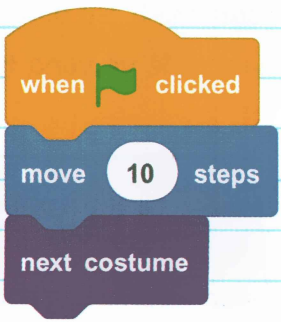
Tham số cho khối thêm thông tin và cho phép bạn tùy chỉnh nhiệm vụ của khối. Ví dụ, bạn có thể sử dụng khối say ( ) để khiến đối tượng nói điều bạn muốn.

Dấu ngoặc ( ) dùng để thể hiện tham số hoặc khoảng trống cần điền khi code bằng Scratch. Ví dụ, khối:



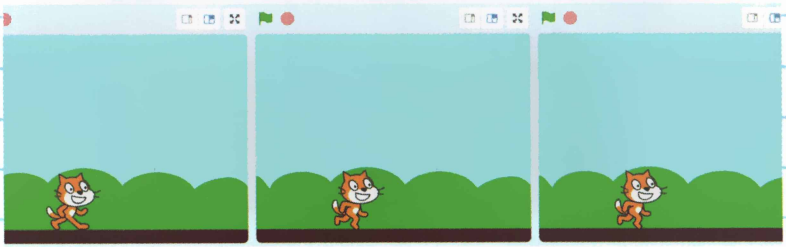
sẽ được viết thành: **move ( ) steps**.

Không phải mọi khối đều có tham số - một số được dán nhãn bằng lệnh chúng thực hiện.

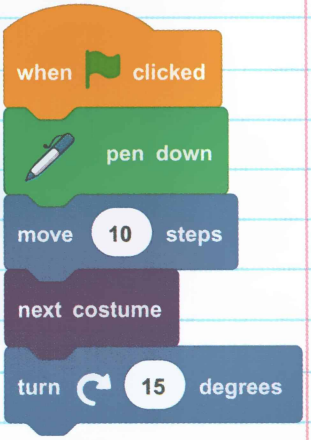


Kịch bản này khiến Mèo Scratch di chuyển 10 bước tới trang phục tiếp theo khi nhấn vào cờ xanh.

Nếu một kịch bản tạo bởi khối sự kiện và khối "move (1) step", khi bạn nhấn vào cờ xanh liên tục, Mèo Scratch sẽ đi bộ qua sân khấu. Nhấn vào cờ xanh 3 lần sẽ khiến chú mèo đi 3 bước như hình minh họa:



Đối tượng không nhất thiết phải đi theo đường thẳng. Để khiến Mèo Scratch đi thành vòng tròn, thêm khối "turn clockwise (15) degrees" vào cuối kịch bản. Sau đó nhấn cờ xanh 24 lần để khiến Mèo Scratch đi thành vòng tròn hoàn chỉnh. Sử dụng khối "pen down" để vẽ vòng tròn.

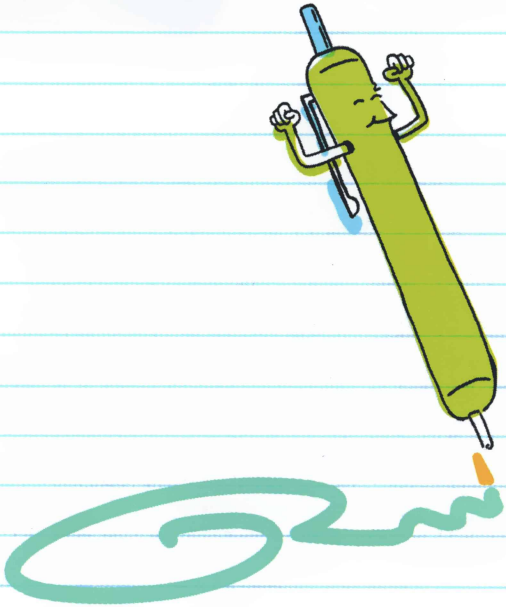


Bạn nhấn cờ xanh 24 lần vì 360 (số độ của 1 vòng tròn) chia cho 15 (số độ của mỗi lần quay) bằng 24.

Để khiến Mèo Scratch di chuyển theo vòng tròn lớn hơn hoặc nhỏ hơn, bạn có thể thay đổi tham số trong khối **move ( ) steps** và **turn clockwise ( ) degrees.** Ví dụ, để tạo vòng tròn lớn hơn, bạn có thể đổi thông tin trong khối **turn clockwise ( ) degrees** từ 15 thành 5. Nhấn cờ xanh 72 lần và chú mèo sẽ đi thành vòng tròn lớn hơn.

Các khối trong danh mục **Pen** dùng để vẽ hình khối trên sân khấu. Khối **pen down** dùng để vẽ đường theo sau đối tượng.

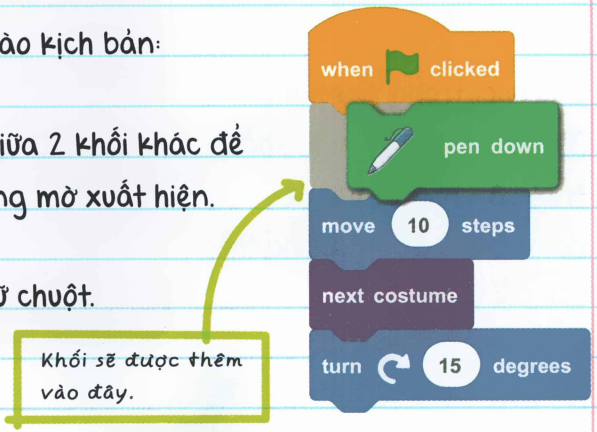
Để thêm khối Bút Vẽ, mở thêm phần mở rộng (góc dưới bên trái) và chọn danh mục Bút Vẽ. Sau khi thêm vào, bạn sẽ thấy danh mục khối Bút Vẽ ở phía dưới thanh công cụ khối.



Có thể chèn khối vào giữa kịch bản, không phải chỉ có thêm ở cuối. Kịch bản lần lượt chạy từ khối đầu tiên tới khối cuối cùng. Nếu muốn để lại đường khi đối tượng di chuyển, khối "pen down" phải được chạy trước khi Mèo Scratch bắt đầu di chuyển. Chèn khối vào đầu kịch bản thay vì cuối cùng.

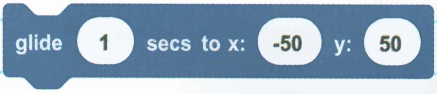
Để chèn khối vào kịch bản:

1. Giữ khối ở giữa 2 khối khác để khoảng trống mờ xuất hiện.
2. Thả tay giữ chuột.



Lướt đi

Một cách khác để điều khiển đối tượng di chuyển quanh sân khấu là khối **glide ( ) secs to x: ( ) y: ( )** trong danh mục Chuyển Động. Thay vì khiến đối tượng đi hoặc di chuyển liên tục như cách khối **move ( ) steps** làm, khối **glide ( ) secs to x: ( ) y: ( )** đưa ra một khoảng thời gian cố định để đối tượng di chuyển tới vị trí mới. Tham số đầu tiên cho biết nhân vật di chuyển trong bao lâu (tính theo giây), và 2 tham số còn lại cho biết vị trí đối tượng di chuyển tới.

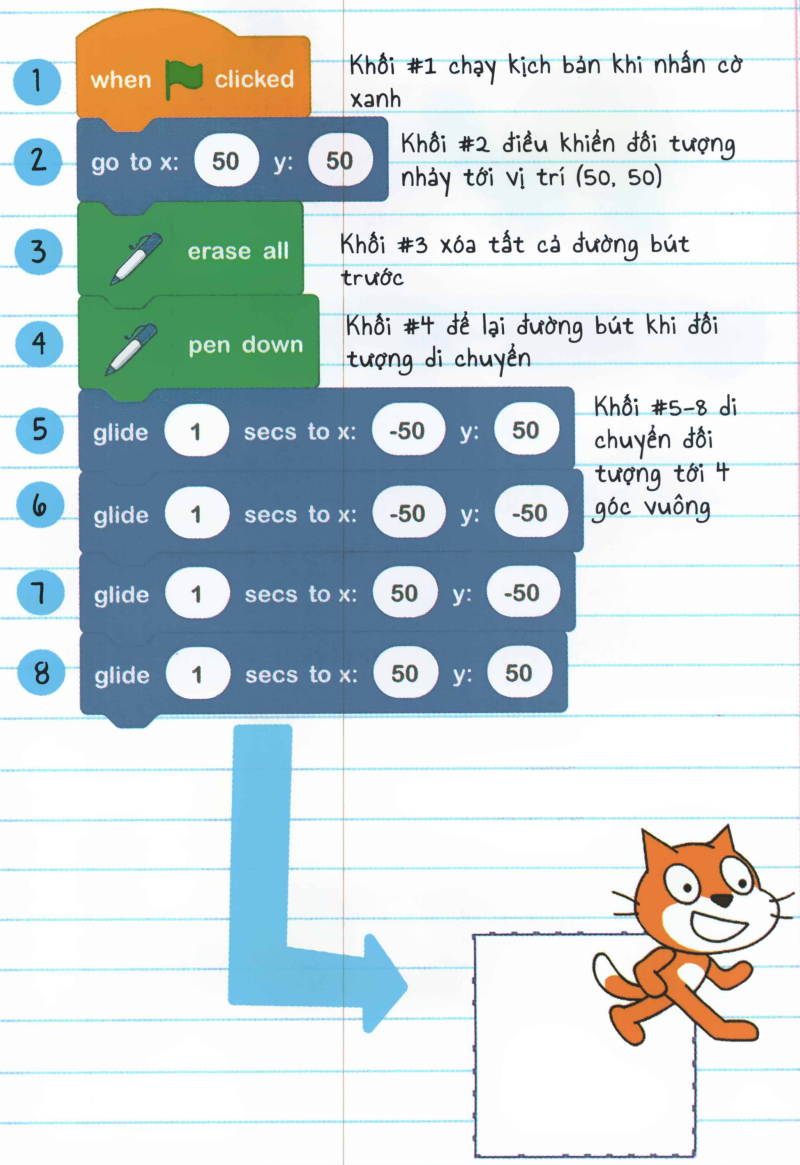


= lướt tới tọa độ (-50, 50) trong 1s

Một khối di chuyển khác - khối **go to x: ( ) y: ( )** - ngay lập tức chuyển đối tượng tới vị trí cụ thể trên màn hình mỗi lần kịch bản chạy. Bằng cách nhập 50 vào tham số x và 50 vào tham số y, Mèo Scratch sẽ tự động nhảy tới vị trí (50, 50).

TRƯỢT TỚI TRANG TIẾP THEO

VÍ DỤ: Kịch bản sau khiến Mèo Scratch lướt đi theo hình vuông.



CHIẾN LƯỢC SỬA LỖI

Chương trình không phải lúc nào cũng chạy theo dự tính của lập trình viên. Có những chiến lược để sửa lỗi (debug):

- Xem lại từng khối trong kịch bản. Đọc to từng khối và nghĩ xem nhiệm vụ của khối là gì.

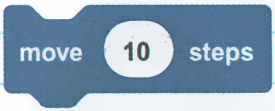
- Làm việc với từng đoạn code ngắn. Chia kịch bản là chạy từng đoạn ngắn. Sau khi chạy một đoạn không gặp vấn đề, thêm một nhóm khối nữa và tiếp tục kiểm tra lỗi sai.

- Bình luận trong kịch bản để giải thích nhiệm vụ của chúng, và xem lại khối để đảm bảo chúng làm đúng dự tính.

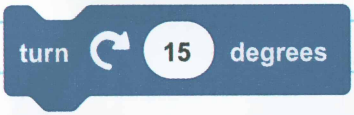
- Nhờ bạn cùng lớp giúp bằng cách cùng thực hiện một trong những chiến lược trên cùng nhau.

Chuyển Động

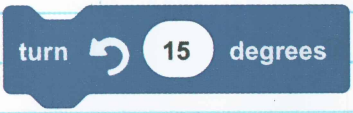
Khối Chuyển Động dùng để di chuyển đối tượng xung quanh.



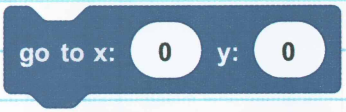
Move ( ) steps: Khối xếp chồng di chuyển đối tượng về phía trước.



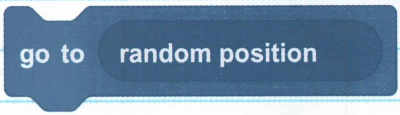
Turn clockwise ( ) degrees: Khối xếp chồng quay đối tượng theo chiều kim đồng hồ (về phía bên phải).



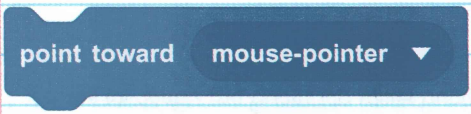
Turn counterclockwise ( ) degrees: Khối xếp chồng quay đối tượng theo ngược chiều kim đồng hồ (về phía bên trái).



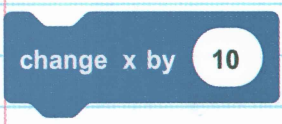
Go to x: ( ) y: ( ): Khối xếp chồng đưa đối tượng nhảy tới vị trí tọa độ x, y.



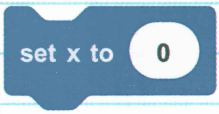
Go to ( ): Khối xếp chồng đưa đối tượng tới nhảy vị trí con trỏ chuột hoặc vị trí ngẫu nhiên.



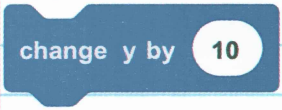
Point toward (): Khối xếp chồng hướng đối tượng về phía con trỏ chuột hoặc đối tượng khác.



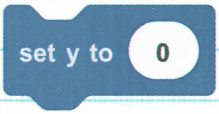
Change x by (): Khối xếp chồng di chuyển đối tượng theo trục x.



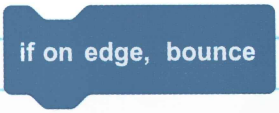
Set x to (): Khối xếp chồng đưa đối tượng nhảy tới vị trí dọc trục x.



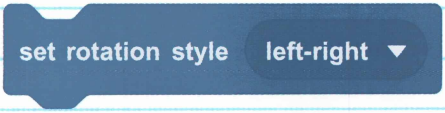
Change y by (): Khối xếp chồng di chuyển đối tượng theo trục y.



Set y to (): Khối xếp chồng đưa đối tượng nhảy tới vị trí dọc trục y.



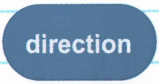
If on edge, bounce: Khối xếp chồng khiến đối tượng bật ngược lại khi chạm rìa sân khấu.



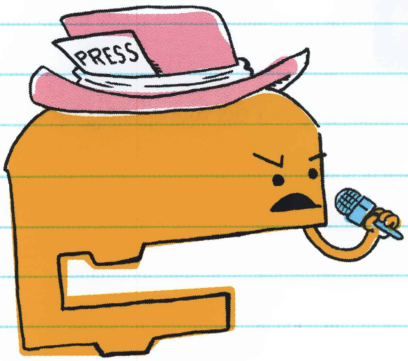
Set rotation style (): Khối xếp chồng kiểm soát cách xoay.

X position: Khối báo cáo giá trị tọa độ x của đối tượng.

Y position: Khối báo cáo giá trị tọa độ y của đối tượng.

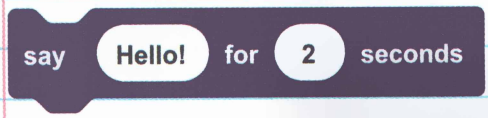


Direction: Khối báo cáo hướng hiện tại của đối tượng.

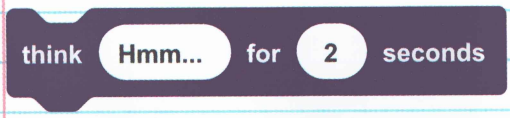


Hình ảnh

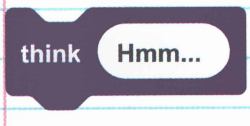
Khối hình ảnh dùng để thay đổi hình ảnh đối tượng.



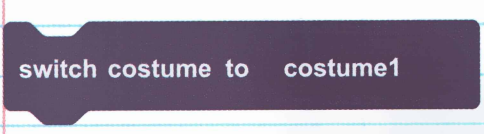
Say ( ) for ( ) seconds: Khối xếp chồng khiến bong bóng hội thoại xuất hiện phía trên đối tượng trong một khoảng thời gian cố định.



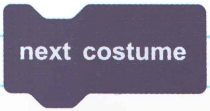
Think ( ) for ( ) seconds: Khối xếp chồng khiến bong bóng suy nghĩ xuất hiện trên đối tượng trong một khoảng thời gian cố định.



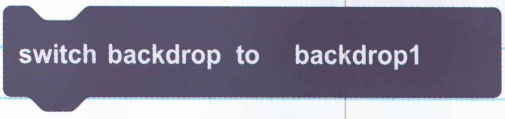
Think ( ): Khối xếp chồng khiến bong bóng suy nghĩ xuất hiện cố định trên đối tượng.



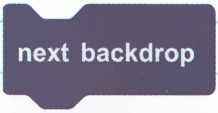
Switch costume to ( ): Khối xếp chồng đổi sang trang phục được đặt tên.



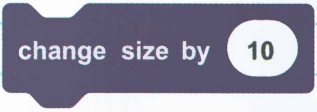
Next costume: Khối xếp chồng thay đổi trang phục đối tượng sang trang phục tiếp theo trong danh sách.



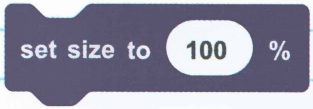
Switch backdrop to ( ): Khối xếp chồng đổi sang phông nền được đặt tên.



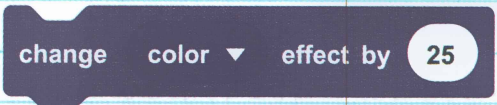
Next backdrop: Khối xếp chồng thay đổi phông nền sang phông nền tiếp theo trong danh sách.



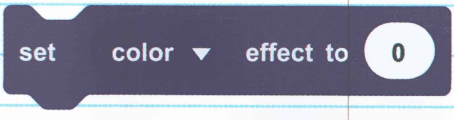
Change size by ( ): Khối xếp chồng thay đổi kích cỡ đối tượng theo chỉ số cụ thể.



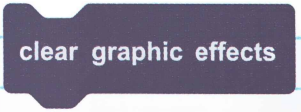
Set size to ( )%: Khối xếp chồng đặt kích cỡ đối tượng về mức phần trăm đã đặt.



Change ( ) effect by ( ): Khối xếp chồng thay đổi hình ảnh của đối tượng theo số cụ thể.



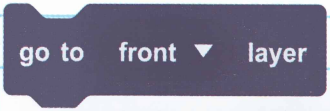
Set ( ) effect to ( ): Khối xếp chồng đặt hình ảnh đối tượng về mức đã đặt.



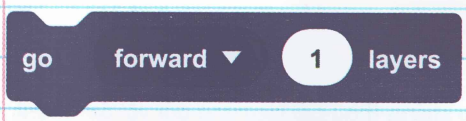
Clear graphic effects: Khối xếp chồng cài đặt lại hiệu ứng đồ họa về 0.

Show: Khối xếp chồng hiển thị đối tượng.

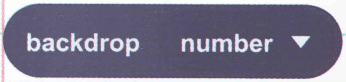
Hide: Khối xếp chồng ẩn đối tượng.



Go to ( ) layer: Khối xếp chồng đưa đối tượng ra phía trước hoặc phía sau tất cả đối tượng khác.



Go ( ) ( ) layers: Khối xếp chồng đưa đối tượng ra phía trước hoặc phía sau đối tượng khác.

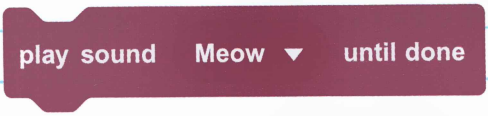


Backdrop ( ): Khối báo cáo giữ số phông nền hiển thị.

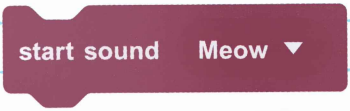
Size: Khối báo cáo giữ giá trị kích cỡ của đối tượng.

Âm Thanh

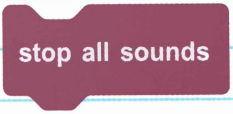
Khối Âm Thanh dùng để tạo và điều chỉnh âm thanh.



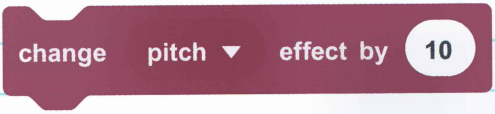
Play sound ( ) until done: Khối xếp chồng phát âm thanh được chọn và kịch bản không dừng chạy đến khi âm thanh phát xong.



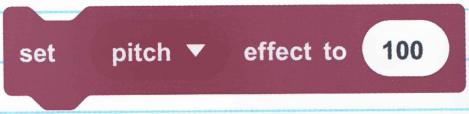
Start sound ( ): Khối xếp chồng phát âm thanh được chọn.



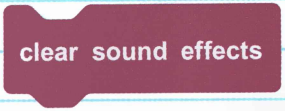
Stop all sound: Khối xếp chồng dừng mọi âm thanh trong chương trình.



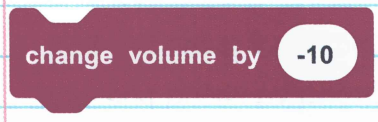
Change ( ) effect by ( ): Khối xếp chồng thay đổi hiệu ứng đối với âm thanh đang phát theo mức đã đặt.



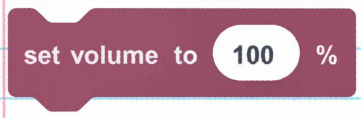
Set ( ) effect to ( ): Khối xếp chồng đặt hiệu ứng đối với âm thanh đang phát về mức đã đặt.



Clear sound effects: Khối xếp chồng xóa toàn bộ hiệu ứng âm thanh.

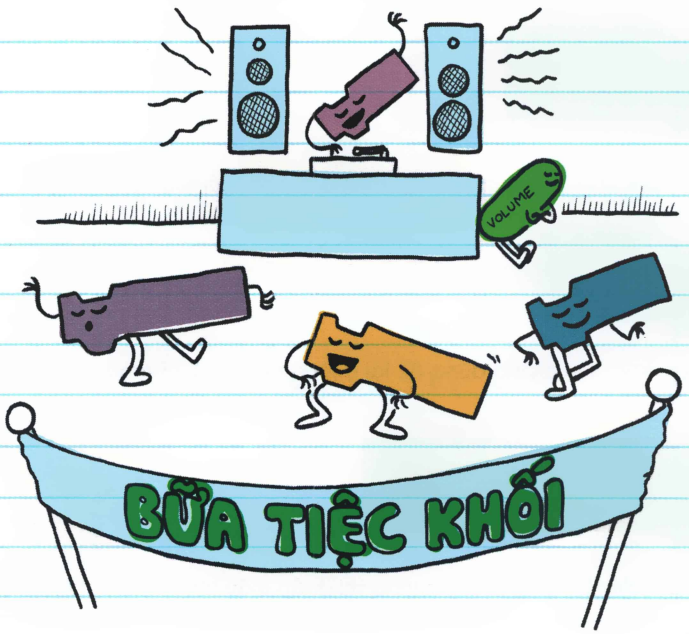


Change volume by ( ): Khối xếp chồng thay đổi âm lượng về mức đã đặt.



Set volume to ( ): Khối xếp chồng thay đổi âm lượng về mức phần trăm đã đặt.

Volume: Khối báo cáo giá trị của mức âm lượng hiện tại.



Bút

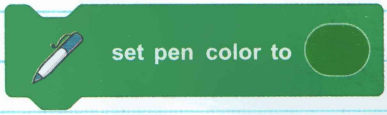
Khối trong danh mục Bút Vẽ dùng để vẽ trên sân khấu. Danh mục sân khấu là phần mở rộng và không phải là một trong những danh mục mặc định.

Erase all: Khối xếp chồng xóa tất cả đường bút và dấu trên sân khấu.

Stamp: Khối xếp chồng đóng dấu hình ảnh đối tượng trên sân khấu. Không thể điều khiển dấu bằng khối. Dấu là bản sao của hình ảnh đối tượng in trên sân khấu.

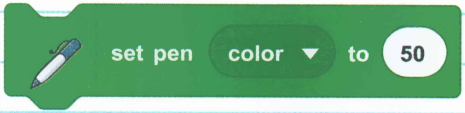
Pen down: Khối xếp chồng để lại đường bút khi đối tượng di chuyển.

Pen up: Khối xếp chồng dừng để lại đường bút khi đối tượng di chuyển.



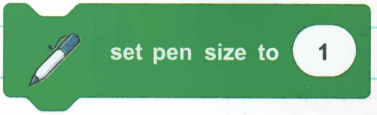
Set pen color to ( ): Khối xếp chồng thiết lập màu bút về màu được chọn.

Change pen color by ( ): Khối xếp chồng thay đổi màu bút với các tham số bên trong.

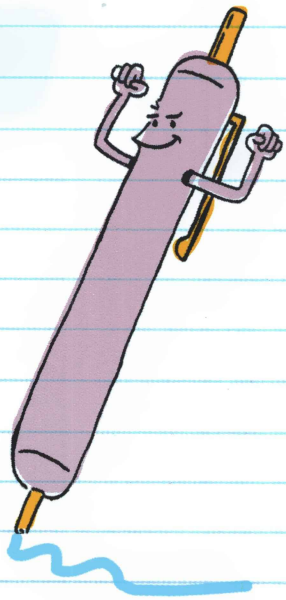


Set pen color to ( ): Khối xếp chồng thiết lập màu cho bút bằng cách chọn màu trên giao diện Scratch.

Change pen size by ( ): Khối xếp chồng thay đổi kích cỡ ngòi bút về số tùy chỉnh.



Set pen size to ( ): Khối xếp chồng thiết lập kích cỡ ngòi bút về số tùy chỉnh.



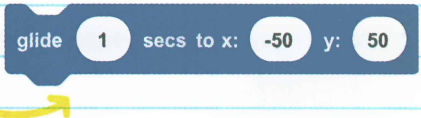
TÔI THẤY VUI
HƠN KHI XEM
BẠN LÀM MẪU.

KIỂM TRA KIẾN THỨC

1. Tất cả khối mũ có điểm chung là gì?

2. Khối **pen down** có chức năng gì?

3. Khối này điều khiển đối tượng làm gì?



4. Làm sao bạn biết kịch bản đang chạy?

5. Chọn các đáp án đúng. Khối mũ đặt ở:
   - **A.** Phía trên khối khác.
   - **B.** Phía dưới khối khác.
   - **C.** Bên trong khối khác.

6. Chọn các đáp án đúng. Khối xếp chồng đặt ở:
   - **A.** Phía trên khối khác.
   - **B.** Phía dưới khối khác.
   - **C.** Bên trong khối khác.

7. Làm thế nào để chèn khối vào giữa hai khối khác?

8. Bạn dùng khối nào để xóa toàn bộ đường bút trên sân khấu?

ĐÁP ÁN

ĐÁP ÁN

1. Khối mũ luôn ở trên cùng kịch bản, và chúng khiến kịch bản chạy.

2. Khối "pen down" khiến đối tượng vẽ theo đường di chuyển.

3. Mất 1 giây để chuyển tới vị trí (-50, 50).

4. Các khối phát sáng ở khu vực kịch bản.

5. **A**

6. **A, B**

7. Kéo khối vào giữa các khối khác, nơi bạn muốn đặt nó. Chờ bóng mờ xuất hiện, sau đó thả chuột để đặt khối vào.

8. **Erase all**

Chương 21

DỮ LIỆU VÀ TOÁN TỬ

BIẾN

Lưu dữ liệu rất hữu ích vì bạn có thể tái sử dụng nó khi chạy chương trình. Có thể sử dụng biến để lưu trữ thông tin:

- **Biến**: giữ một loại giá trị – số, chuỗi (chữ), Boolean, hoặc danh sách.

Nhắc nhở:
Giá trị Boolean chỉ có thể là đúng hoặc sai.

KHỐI BÁO CÁO[1] cho biết giá trị hiện tại của biến mà chúng đại diện. Chúng có thể lưu số và chữ như giá trị biến.

[1] Reporter block.

Khối báo cáo được đặt bên trong khoảng trống tham số của bất kỳ khối nào khác.

> Có khối báo cáo ở mọi danh mục khối trừ danh mục Sự Kiện và Điều Khiển.

Khối báo cáo có hình dạng như này:

Biến

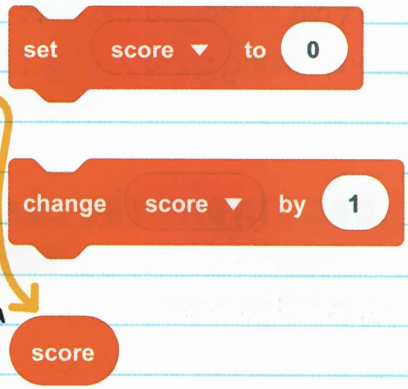
Bạn có thể tạo biến mới bằng cách nhấn nào "Tạo Một Biến" trong danh mục Biến trên thanh công cụ khối. Sau khi bạn tạo biến mới, một khối báo cáo với tên của biến sẽ xuất hiện trong bảng khối.

Có hai cách để thay đổi giá trị của biến:

1. Khối set ( ) to ( ): sử dụng để thay thế giá trị hiện tại với giá trị mới.

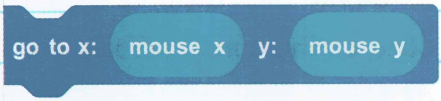
2. Khối change ( ) by ( ): sử dụng để thêm hoặc bớt giá trị hiện tại của biến.

VÍ DỤ, chúng ta có thể tạo biến tên là "score" (điểm) trong trò chơi và sử dụng khối xếp chồng set (score) to (0) để đặt biến điểm về 0 khi trò chơi bắt đầu. Chúng ta cũng có thể dùng khối xếp chồng change (score) by (1) để tăng 1 điểm mỗi lần người chơi ghi điểm trong trò chơi. Chúng ta có thể truy cập giá trị của biến điểm bằng khối báo cáo điểm. Khối báo cáo có thể dùng để hiển thị điểm trên sân khấu hoặc thông báo khi nào người chơi thắng dựa trên số điểm đã ghi.



Biến cài sẵn (Built-In Variables)

Một số khối báo cáo có biến có sẵn lưu trữ dữ liệu. Giá trị của biến có thể dùng trong chương trình.

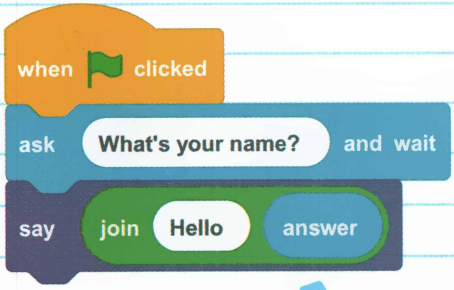


Ví dụ, biến có sẵn gọi "mouse x" và "mouse y" là vị trí tọa độ x y của con trỏ chuột trên sân khấu. Những giá trị này có thể được điền vào ô tham số của khối xếp chồng **go to x: ( ) y: ( )** để điều khiển đối tượng đi tới vị trí của con trỏ chuột trên sân khấu.

thứ người dùng gõ vào.

THÔNG TIN ĐẦU VÀO CỦA NGƯỜI DÙNG

Một số biến có sẵn lưu trữ **THÔNG TIN ĐẦU VÀO CỦA NGƯỜI DÙNG**[1]. Chúng ta yêu cầu người dùng nhập từ ngữ hoặc chữ số bằng cách dùng khối xếp chồng **ask ( ) and wait** trong danh mục Cảm Biến. Câu trả lời của họ được lưu trong khối báo cáo **answer**. Chúng ta có thể yêu cầu người dùng điền tên, sau đó dùng khối **say( )** để cho đối tượng chào người dùng bằng tên của họ.



Khối báo cáo Answer: Thông tin đầu vào của người dùng được lưu ở đây.

[1] User input.

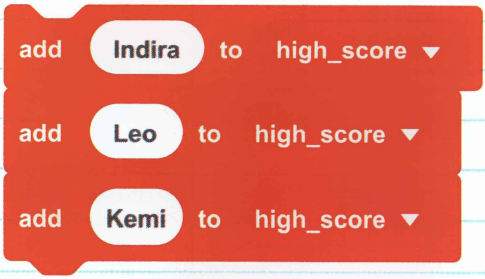
Khối **ask () and wait** tạo bong bóng hội thoại xuất hiện phía trên đối tượng và hộp văn bản xuất hiện ở dưới dùng sân khấu. Kịch bản tạm ngừng đến khi người dùng gõ câu trả lời vào hộp văn bản và nhấn phím enter trên bàn phím.

Danh sách[1]

Trong Scratch bạn có thể lưu nhiều biến tại một chỗ bằng cách sử dụng DANH SÁCH.

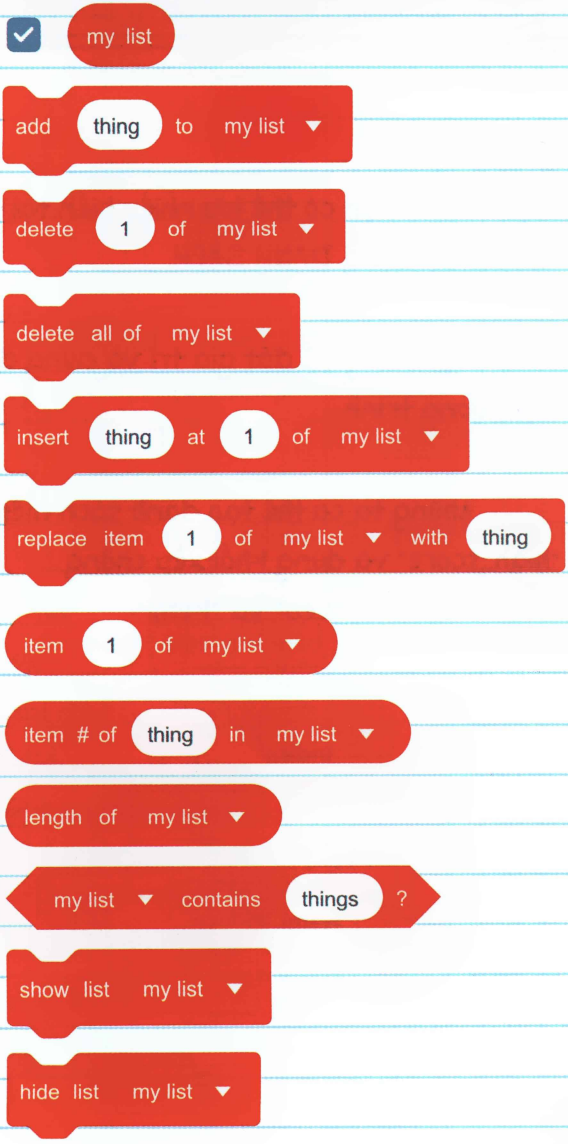
Khối list được dùng để đặt giá trị và dùng dữ liệu đó sau trong chương trình.

VÍ DỤ, chúng ta có thể tạo danh sách điểm cao gọi là "high_score" và dùng khối xếp chồng **add () to ()** để thêm tên "Indira", "Leo" và "Kemi":

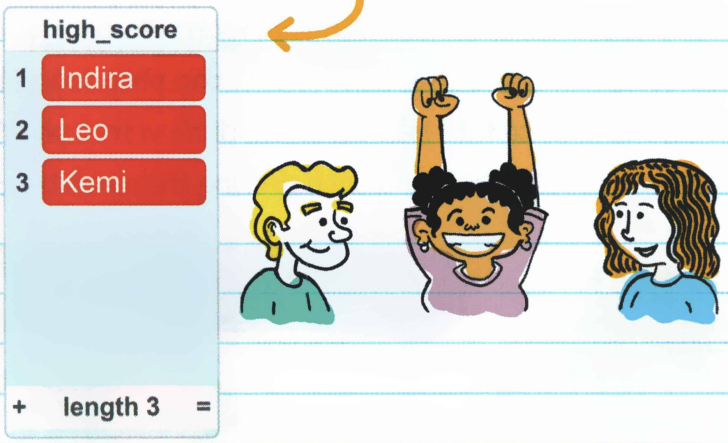


[1] List.

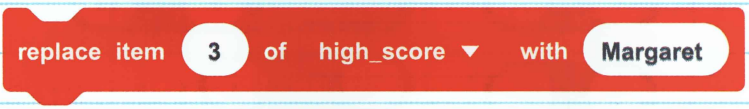
Tạo một danh sách mới bằng cách nhấn nút "Tạo một danh sách" trong danh mục Biến ở thanh công cụ khối. Sau khi tạo danh sách mới, những khối sau sẽ xuất hiện trên bảng khối:



Ban đầu danh sách trống. Sử dụng khối add ( ) to ( ) để thêm các cái tên vào danh sách high_score. Danh sách hiển thị (trên sân khấu) ba cái tên đã được thêm vào danh sách high_score.



Khối xếp chồng replace item ( ) of ( ) with ( ) dùng để thay thế phần tử trong danh sách. Ở tham số đầu, gõ số thứ tự của phần tử cần thay thế. Để thay thế Kemi với Margaret, gõ "3" vì Kemi xếp thứ 3 trong danh sách. Ở tham số cuối, gõ "Margaret" vì đó là giá trị mới.



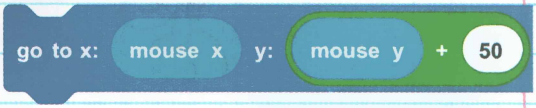
Những khối list khác có thể dùng để xóa hoặc thêm danh sách phần tử mới.

TOÁN TỬ

TOÁN TỬ dùng để thực hiện phép tính trên giá trị. Có hai loại khối toán tử:

BÁO CÁO và **BOOLEAN**

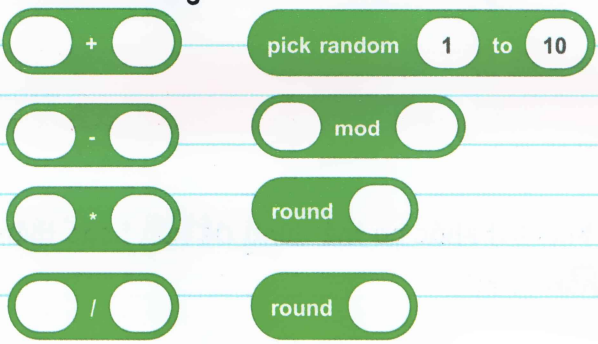
Khối báo cáo dùng để thực hiện hàm toán trên giá trị (ví dụ như cộng, trừ, nhân, chia, làm tròn và các phép toán khác). Ví dụ, chúng ta có thể khiến đối tượng đi tới vị trí ở phía trên con trỏ chuột 50 bước, bằng cách sử dụng thêm khối báo cáo:



Khối toán tử Boolean dùng để đánh giá mệnh đề đúng hay sai. Ví dụ, chúng ta có thể dùng khối ( ) > ( ) để xem liệu điểm người chơi có lớn hơn 50:

Khối Toán

Ở danh mục Toán Tử, có những khối báo cáo có sẵn toán tử toán học bên trong:

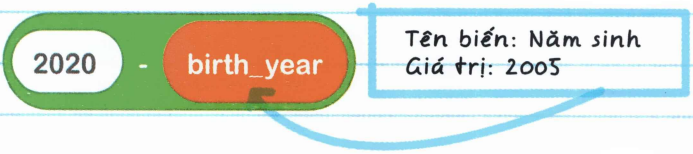


Mỗi khối có một hoặc hai ô trống để bạn điền số hoặc giá trị. Giá trị của mỗi khối là kết quả của phép toán nằm trong khối.

VÍ DỤ, nếu bạn muốn thực hiện phép tính 6-2, bạn sẽ dùng khối ( ) - ( ):

Bạn có thể thực hiện phép toán với các biến bằng cách chèn khối báo cáo biến vào trong ô trống của khối toán.

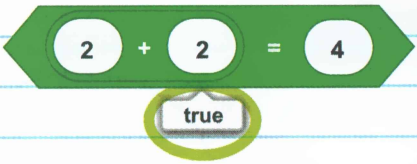
VÍ DỤ, để biết tuổi của một người, bạn có thể đặt biến "birth_year" là năm sinh của người ấy (ví dụ, 2005) và lấy năm hiện tại trừ đi biến đó.



Khối Boolean

KHỐI BOOLEAN[1] đánh giá thông tin đúng hay sai. Bên trong chúng chứa mệnh đề hoặc phép toán có thể đúng hoặc sai.

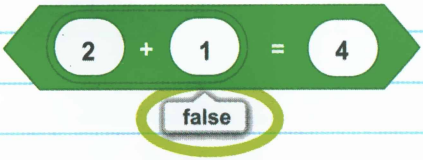
VÍ DỤ:



Khối boolean "=" xem xét hai vế và đánh giá xem chúng có bằng nhau.

Giá trị của khối boolean đúng vì 2 + 2 = 4.

Nếu khối boolean chứa phép toán 2 + 1 = 4, giá trị của khối là sai.



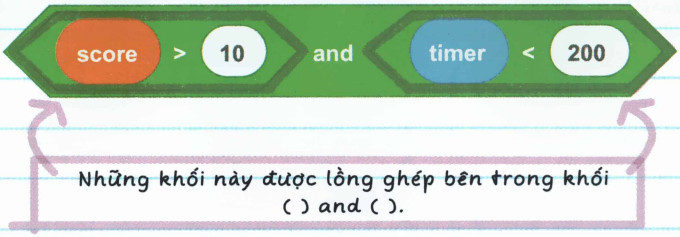
[1] Khối logic.

Khối boolean có thể dùng để so sánh từ ngữ. Ví dụ, Marcos = Marcos là đúng, Marcos = Greg là sai.

Cẩn thận với phím cách! Khoảng trống trước hoặc sau từ ngữ được so sánh phải giống nhau.

Trong danh mục Toán Tử, có các khối Boolean để so sánh (<,=,>) và phép toán logic (and, or, not). Những khối này có thể kết hợp bằng cách **lồng ghép** vào nhau.

VÍ DỤ, bạn có thể dùng khối boolean ( ) > ( ), ( ) < ( ), và ( ) and ( ) để xem liệu biến score có lớn hơn 10 và biến timer có nhỏ hơn 200. Để làm việc này, chèn khối ( ) > ( ) và ( ) < ( ) vào ô trống bên trong khối ( ) and ( ):



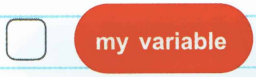
Lồng ghép khối nghĩa là đặt một hoặc nhiều khối vào bên trong khối khác.

Trong danh mục Cảm Biến, có các khối Boolean đánh giá các điều kiện cụ thể. Nếu điều kiện thỏa mãn, giá trị khối là đúng, ngược lại giá trị là sai.

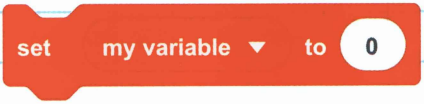
VÍ DỤ, bạn có thể dùng khối Boolean key () pressed? để xem phím cách trên bàn phím đã được nhấn. Hoặc, bạn có thể dùng khối touching color () để xem liệu đối tượng đã chạm đến màu nhất định. Khi những điều kiện trên thỏa mãn, khối sẽ có giá trị đúng.

Biến

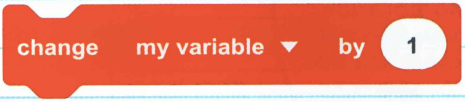
Những khối trong danh mục Biến dùng để lưu và thay đổi dữ liệu với biến giá trị đơn hoặc danh sách đa giá trị.



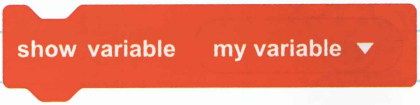
variable name: Khối báo cáo giữ giá trị của biến được đặt tên.



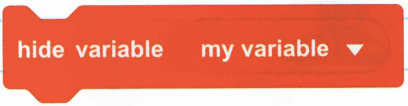
Set ( ) to ( ): Khối xếp chồng đặt giá trị của biến được chọn. Giá trị của biến có thể là số, chuỗi hoặc Boolean đã đặt.



Change ( ) by ( ): Khối xếp chồng thay đổi giá trị của biến được chọn theo mức đã đặt.

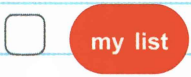


Show variable ( ): Khối xếp chồng hiển thị giá trị biến ở màn hình nhỏ trên sân khấu.

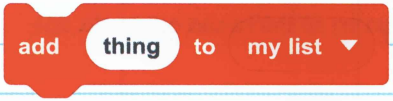


Hide variable ( ): Khối xếp chồng ẩn hiển thị giá trị biến trên sân khấu.

Những khối sau cho phép người dùng thay đổi giá trị danh sách.

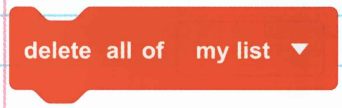


list name: Khối báo cáo giữ (các) giá trị của danh sách được đặt tên.

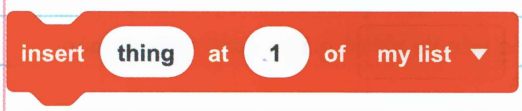


Add ( ) to ( ): Khối xếp chồng thêm giá trị đã đặt vào cuối danh sách được chọn.

Delete ( ) of ( ): Khối xếp chồng xóa phần tử trong danh sách được chọn.



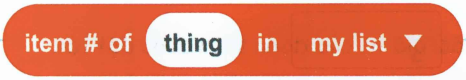
Delete all of ( ): Khối xếp chồng xóa tất cả phần tử trong danh sách được chọn.



Insert ( ) at ( ) of ( ): Khối xếp chồng chèn giá trị tại một vị trí trong danh sách được chọn.

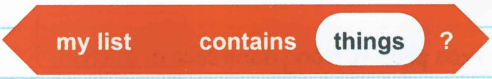
Replace item ( ) of ( ) with ( ): Khối xếp chồng thay thế giá trị tại một vị trí trong danh sách được chọn.

Item ( ) of ( ): Khối báo cáo giữ giá trị của một phần tử trong danh sách được chọn.

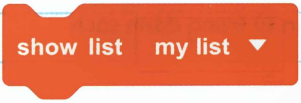


Item # of ( ) in ( ): Khối báo cáo giữ giá trị số thứ tự của giá trị đã đặt trong danh sách.

Length of ( ): Khối báo cáo giữ số phần tử trong danh sách được chọn.



( ) contains ( )?: Khối Boolean hỏi danh sách được chọn có giá trị đã đặt (Đây có phải là một phần tử trong danh sách? Nếu có thì đúng, không thì sai).



Show list ( ): Khối xếp chồng hiển thị tất cả giá trị danh sách tại màn hình hiển thị trên sân khấu.

Hide list ( ): Khối xếp chồng ẩn màn hình hiển thị giá trị danh sách trên sân khấu.

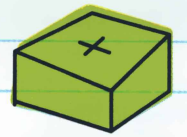
Những khối trên cho phép người dùng thay đổi nhiều hơn một giá trị trong danh sách được chọn.

Toán Tử

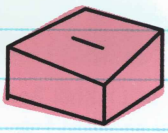
Khối trong danh mục Toán Tử dùng để kiểm soát dữ liệu. Nhiều khối báo cáo là hàm tính (ví dụ như cộng, trừ, nhân, chia và hơn nữa). Các khối khác dùng để thay đổi **GIÁ TRỊ CHUỖI**.

GIÁ TRỊ CHUỖI
Vài biểu tượng hoặc giá trị được đặt trong một hàng.

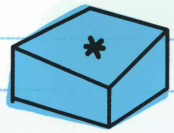
() + (): Khối báo cáo giữ giá trị của phép cộng.



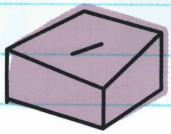
() - (): Khối báo cáo giữ giá trị của phép trừ.



() * (): Khối báo cáo giữ giá trị của phép nhân.



() / (): Khối báo cáo giữ giá trị của phép chia.



Pick random ( ) to ( ): Khối báo cáo đặt giá trị là số ngẫu nhiên giữa hai giá trị đã đặt.

( ) > ( ): Khối Boolean đánh giá đúng hay sai dựa vào giá trị bên trái có lớn hơn giá trị bên phải.

( ) = ( ): Khối Boolean đánh giá đúng hay sai dựa vào giá trị bên trái có bằng giá trị bên phải.

( ) and ( ): Khối Boolean đánh giá đúng hay sai dựa trên việc cả hai khối bên trong là đúng.

( ) or ( ): Khối Boolean đánh giá đúng hay sai dựa trên việc một hoặc cả hai khối bên trong là đúng.

Not ( ): Khối Boolean đánh giá giá trị ngược lại của giá trị tham số, vậy nên nếu tham số đúng thì khối sai và ngược lại.

Join ( ) ( ): Khối báo cáo ghép hai giá trị vào một chuỗi.

Letter ( ) of ( ): Khối báo cáo giữ giá trị của một chữ cái nhất định trong một từ.

Length of ( ): Khối báo cáo giữ giá trị của số chữ cái trong một từ.

( ) contains ( )?: Khối Boolean đánh giá đúng sai dựa vào giá trị bên trái có chứa chữ cái bên phải.

( ) mod ( ): Khối báo cáo giữ giá trị phần còn lại của giá trị đầu tiên chia cho số giây.

Round ( ): Khối báo cáo làm tròn giá trị thành số nguyên gần nhất.

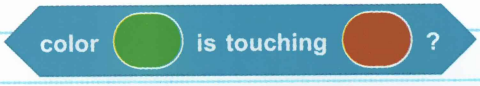
( ) of ( ): Khối báo cáo giữ giá trị kết quả phép tính được chọn bằng cách sử dụng giá trị số đã đặt (Ví dụ, bạn có thể dùng khối này để tìm ra căn bậc hai của một giá trị).

Cảm biến

Khối trong danh mục Cảm Biến dùng để hiểu và lưu thông tin về cách người dùng tương tác với chương trình.

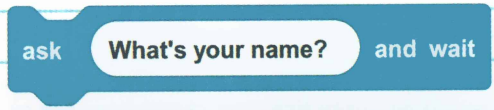
Touching ()?: Khối Boolean đánh giá đúng sai dựa trên đối tượng đã chạm vào con trỏ chuột hoặc đối tượng khác hay chưa.

Touching color ()?: Khối Boolean đánh giá đúng sai dựa trên đối tượng đã chạm vào màu được chọn.



Color ( ) is touching ()?: Khối Boolean đánh giá đúng sai dựa trên màu được chọn đã chạm vào màu được chọn khác chưa.

Distance to (): Khối báo cáo giữ giá trị khoảng cách giữa đối tượng và con trỏ chuột hoặc đối tượng khác.



Ask ( ) and wait: Khối xếp chồng yêu cầu người dùng trả lời câu hỏi, sử dụng hộp văn bản và chờ câu trả lời.

Answer: Khối báo cáo giữ giá trị thông tin người dùng lấy từ khối ask.

Key ( ) pressed?: Khối Boolean đánh giá đúng sai dựa trên phím được chọn đã được ấn hay chưa.

Mouse down?: Khối Boolean đánh giá đúng sai dựa trên con chuột đã được ấn và giữ chưa.

Mouse x: Khối báo cáo giữ giá trị vị trí chuột trên trục x.

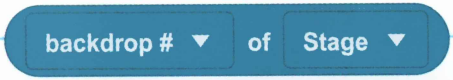
Mouse y: Khối báo cáo giữ giá trị vị trí chuột trên trục y.

Set drag mode ( ): Khối xếp chồng cho phép người dùng kéo đối tượng hoặc không.

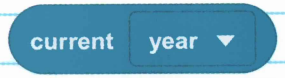
Loudness: Khối báo cáo giữ giá trị độ lớn của micro sử dụng.

Timer: Khối báo cáo giữ giá trị khoảng thời gian tự động tính từ lúc chương trình chạy.

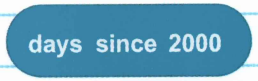
Reset timer: Khối xếp chồng đặt thời gian về 0.



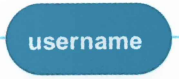
( ) of ( ): Khối báo cáo giữ giá trị biến được chọn cho đối tượng hoặc sân khấu được chọn.



Current ( ): Khối báo cáo giữ giá trị mốc thời gian hiện tại, có thể là năm, tháng, ngày, thứ, giờ, phút, hoặc giây.



Days since 2000: Khối báo cáo giữ giá trị số ngày tính từ năm 2000.



Username: Khối báo cáo giữ giá trị tên người dùng Scratch.

KIỂM TRA KIẾN THỨC

1. Chọn đáp án đúng. Khối báo cáo có thể đặt ở:

   A. Trên các khối khác.

   B. Dưới các khối khác.

   C. Trong các khối khác.

2. Bạn có thể điền gì vào ô nhập dữ liệu?

3. Hai giá trị khả dĩ của khối Boolean là?

4. Giải thích nhiệm vụ của khối báo cáo. Tại sao chúng hữu dụng?

5. Sự khác biệt giữa khối set ( ) to ( ) và khối change ( ) by ( )?

6. Giá trị của khối dưới đây?

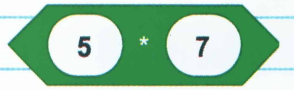
   A. Đúng

   C. Sai

   C. 12

   D. 35

7. Giá trị của khối dưới đây?



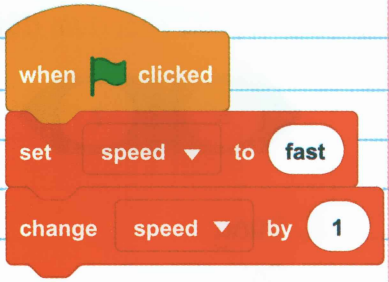
A. Đúng
B. Sai
C. 12
D. 35

8. Bạn dùng khối Boolean nào để biết phím W được ấn?

9. Sự khác biệt giữa khối toán (+, -, *, /) và khối so sánh (<, >, =)?

10. Lợi ích của việc sử dụng một biến nhiều lần thay vì gõ ra giá trị mỗi lần biến được sử dụng làm tham số?

11. Code sau có gì sai?



12. Giải thích cách tạo biến "vận tốc" trong Scratch.

ĐÁP ÁN

1. C

2. Chữ, số, và khối báo cáo

3. Đúng hoặc sai

4. Khối báo cáo là biến. Chúng lưu thông tin. Bạn có thể đặt chúng và ô trống của khối khác.

5. Khối set ( ) to ( ) gán giá trị cho biến, còn khối change ( ) by ( ) thêm số vào giá trị biến.

6. A

7. D

8. Khối key ( ) pressed? và chọn phím W

9. Khối toán lưu giá trị số, nhưng toán tử so sánh là Boolean có giá trị đúng hoặc sai.

10. Nếu bạn cần thay đổi giá trị, bạn có thể thay đổi nó một lần bằng cách gán giá trị mới cho biến thay vì tìm từng chỗ có giá trị và thay nó một cách thủ công.

11. Bạn không thể dùng khối change ( ) by ( ) cho giá trị biến không phải số. Ví dụ, "tốc độ" được đặt là "nhanh" - đây không phải số.

12. Trong bảng khối "Biến", ấn vào "Tạo một biến". Sau đó gõ "vận tốc" vào ô "Tên biến" và nhấn "OK".

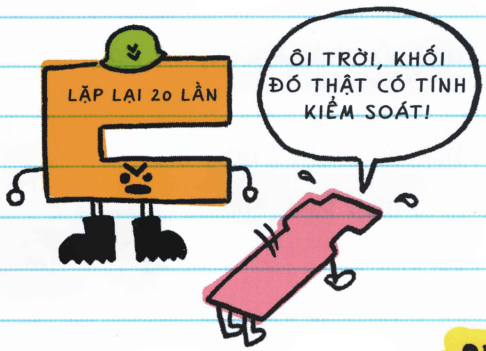
Chương 22

KHỐI ĐIỀU KHIỂN VÀ KHỐI SỰ KIỆN

KHỐI ĐIỀU KHIỂN

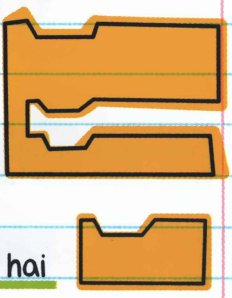
Những khối này ra lệnh cho những khối khác. Chúng điều khiển kịch bản chạy, tạm ngừng, lặp lại và dừng lại. Chúng khác với khối mũ xử lý sự kiện vì chúng chạy dựa trên những điều kiện bên trong thay vì điều kiện bên ngoài (ví dụ như nhấn vào cờ xanh).

Một dạng khối điều khiển là khối lặp lại, nó sẽ lặp lại code bên trong nó theo số lần đã đặt.



KHỐI C[1] VÀ KHỐI NẮP[2]

Danh mục Điều Khiển có hai loại khối khác nữa: **KHỐI C** và **KHỐI NẮP.**



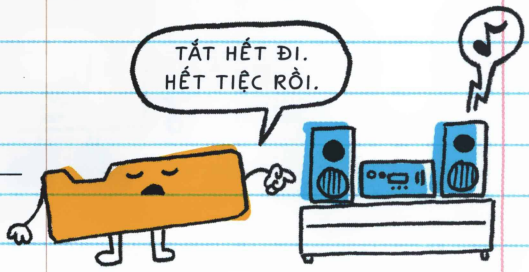
Khối C trông giống như chữ cái C. Các khối khác có thể đặt trong khoảng trống của chữ C để khối C bọc xung quanh chúng. Có hai loại khối C chính: **lặp** và **điều kiện.**

- **Khối lặp** sẽ lặp lại code đặt trong khoảng trống của khối.
- **Khối điều kiện** chỉ chạy code đặt trong khoảng trống khi điều kiện thỏa mãn.
- Khối C nằm trong danh mục Điều Khiển. Có những dạng lặp và điều kiện khác nhau, nhưng chúng đều có hình chữ C.

Khối nắp gắn ở cuối kịch bản và dùng để dừng kịch bản đang chạy.

Các khối khác không thể nằm dưới khối nắp.

Khối nắp nằm trong danh mục Điều Khiển.



[1] C Block.
[2] Cap Block.

KHỐI SỰ KIỆN

Có 6 khối xử lý sự kiện. Khối mũ chờ những thao tác như nhấn nút, nhấn chuột, hoặc phát tin. Khi bộ xử lý sự kiện nhận thấy sự kiện đã diễn ra, chúng sẽ chạy kịch bản gắn liền với chúng.

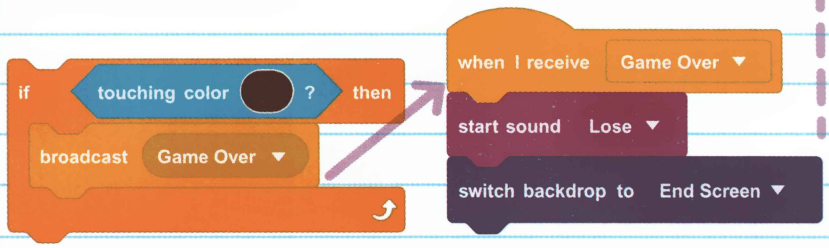
Khối sự kiện có thể dùng để tạo bảng âm thanh phát những tiếng khác nhau khi ấn những phím khác nhau.

VÍ DỤ, số 1, 2, 3 và 4 được gán cho các âm thanh khác nhau.

Khối when backdrop switches to ( ) theo dõi phông nền đổi. Ví dụ, nếu bạn tạo một trò chơi đổi phông nền mỗi lần sang màn mới, bạn có thể cài sự kiện theo dõi phông nền đổi và kích hoạt: Đối thủ mới xuất hiện.

Khối phát tin dùng để phát tin nhắn. Khi khối này chạy, nó phát tin nhắn tới tất cả đối tượng. Người dùng không thấy tin nhắn này, chỉ khối when I receive ( ) phản hồi khi tin nhắn được phát đi.

VÍ DỤ, bạn có thể dùng khối phát tin trong kịch bản phát tin nhắn "Game over". Bạn có thể thêm khối **when I receive (Game Over)** vào trong đối tượng hoặc hình nền để chạy khiến khối bật tiếng thua cuộc và chuyển phông nền kết thúc trò chơi.



LẶP

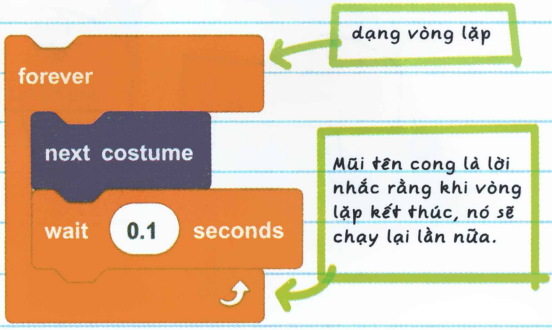
Tất cả khối lặp đều thuộc khối chữ C. Trên cùng chữ C là dạng lặp (forever, repeat, hoặc repeat until)[1]. Code được lặp lại nằm trong "miệng" chữ C. Các khối khác có thể chồng ở trên hoặc dưới vòng lặp.

Vòng lặp cơ bản

Scratch sử dụng 3 dạng vòng lặp:

forever, **repeat ( )**, và **repeat until ( )**.

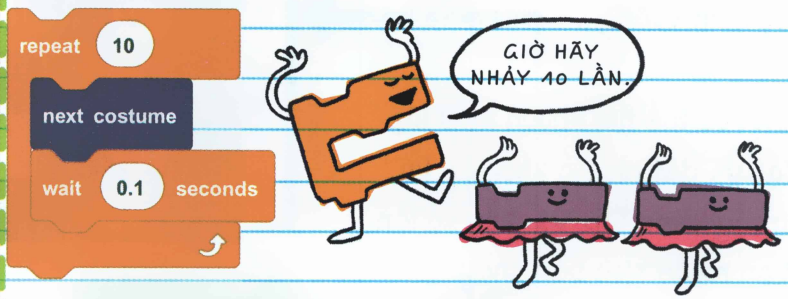
Khối **Forever** sẽ chạy mãi mãi, hoặc đến khi chương trình dừng. Bạn có thể hoạt hóa đối tượng trông như đang đi bộ bằng cách sử dụng khối **forever**. Bạn có thể khiến đối tượng liên tục đổi sang trang phục tiếp theo. Vòng lặp chạy nhanh, nên bạn cần thêm khối **wait ( ) secs** để làm chậm vòng lặp.



[1] Mãi mãi, lặp lại, lặp lại đến khi.

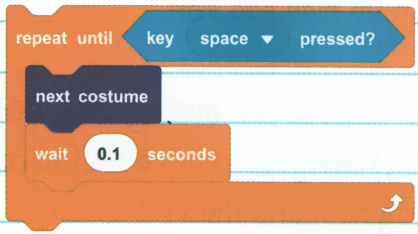
Vòng lặp **Repeat ( )** ra lệnh cho đối tượng chính xác số lần bạn muốn nó chạy bằng cách gõ số vào ô trống.

VÍ DỤ, bạn có thể khiến đối tượng thay đổi trang phục tiếp theo và chờ 0.1 giây chỉ 10 lần thay vì mãi mãi:

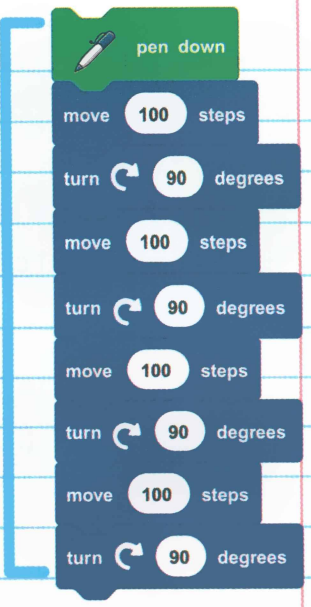


Vòng lặp **Repeat until ( )** sẽ tiếp tục lặp lại đến khi mệnh đề Boolean bên trong vòng lặp là đúng.

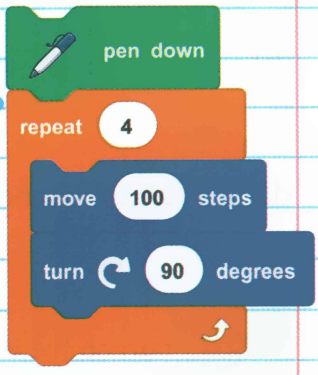
VÍ DỤ, bạn có thể khiến đối tượng liên tục thay đổi trang phục tiếp theo và chờ 0.1 cho đến khi người dùng nhấn phím cách:



Vòng lặp giúp chương trình hiệu quả hơn. Ví dụ, nếu bạn muốn vẽ hình vuông, bạn có thể dùng vòng lặp để cải thiện kịch bản của bạn. Thay vì dùng toàn bộ chỗ khối này để vẽ hình vuông...



bạn có thể tìm khuôn mẫu lặp lại, và đặt những khối lặp lại trong vòng lặp, và điền số lần bạn muốn kịch bản chạy.



Vòng lặp lồng ghép

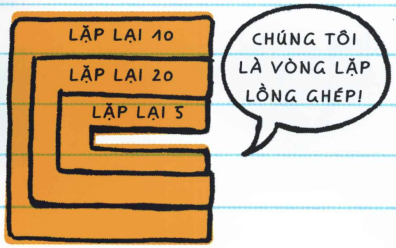
Vòng lặp lồng ghép là vòng lặp chèn trong vòng lặp. Vòng lặp lồng ghép rất hữu ích để lặp lại **CHƯƠNG TRÌNH CON**[1] trong nhiệm vụ. Chúng giúp kịch bản ngắn và hiệu quả hơn.

Chương trình con thường được dùng trong những trò chơi và để tạo ra những hình ảnh thú vị.

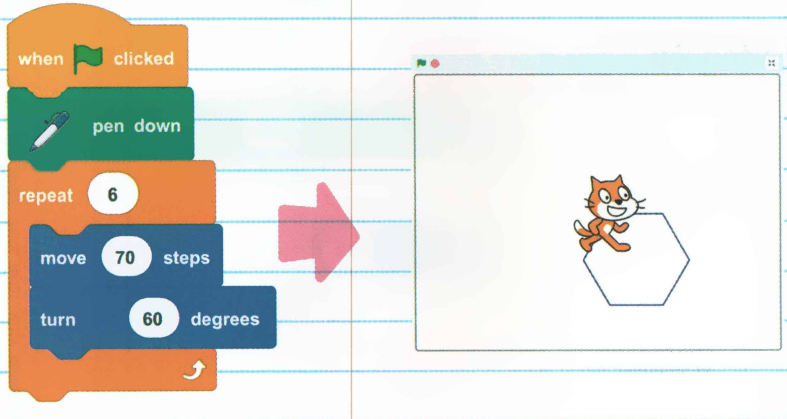
CHƯƠNG TRÌNH CON
Tổ hợp code thực hiện một nhiệm vụ cụ thể, và được dùng như một phần của chương trình lớn hơn.

[1] Subroutine.

VÍ DỤ, trong trò chơi mà người chơi cố bắt quả táo đang rơi, vòng lặp **forever** có thể dùng để khiến trò chơi tiếp diễn mãi mãi, tạo quả táo ở trên cùng màn hình mỗi lần nó chạm đáy. Vòng lặp **repeat ()** dùng để tạo hoạt ảnh quả táo rơi:

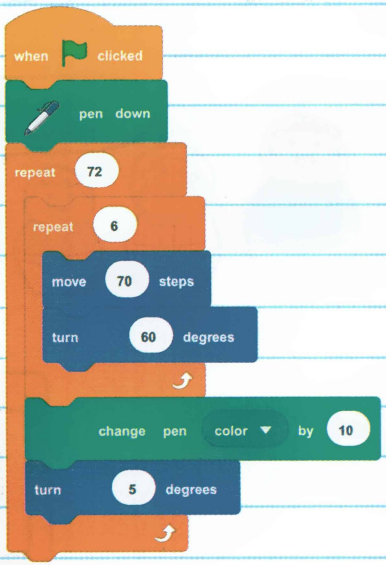


Bạn có thể dùng vòng lặp lồng ghép để tạo hình nghệ thuật. Vòng lặp **repeat (6)** vẽ một hình lục giác. Mèo Scratch dùng bút để vẽ hình lục giác khi nó di chuyển theo đường được thiết lập bởi vòng lặp **repeat (6)**.

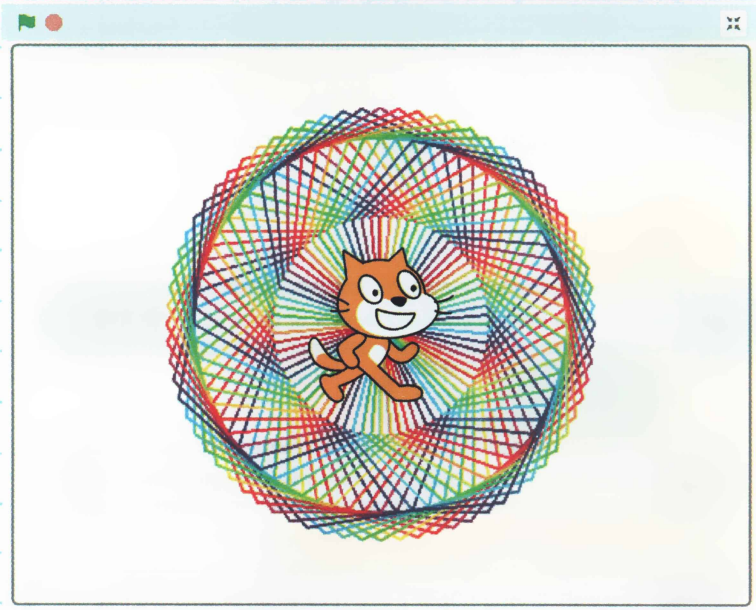


Sau đó hãy lồng ghép vòng lặp đó trong vòng lặp **repeat (72)** để vẽ hình lục giác 72 lần.

Bạn cũng có thể xoay 5 độ mỗi lần thay màu hình lục giác.



Dưới đây là thành quả của kịch bản. Có 72 hình lục khác khác màu, mỗi hình xoay 5 độ so với hình bên tay phải.



CÂU LỆNH ĐIỀU KIỆN

Scratch có hai dạng khối điều kiện:

If ( ) then

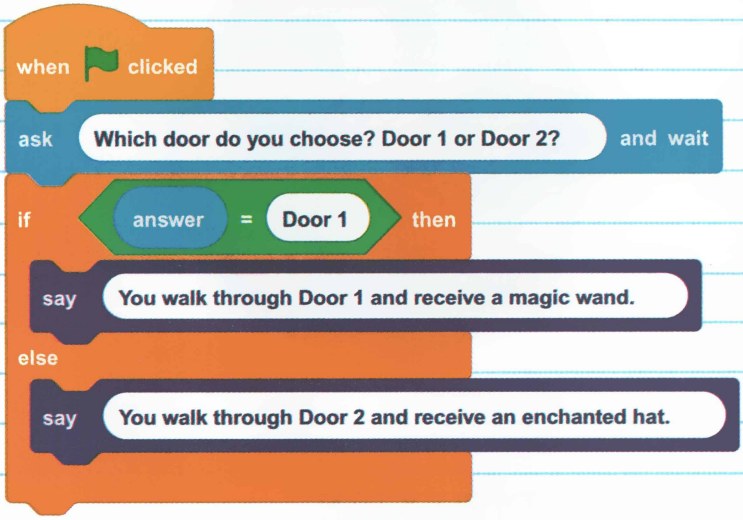
If ( ) then else

Câu lệnh điều kiện sử dụng khối dạng C và nằm trong danh mục Điều Khiển của thanh công cụ khối.

Câu lệnh điều kiện cơ bản

Câu lệnh điều kiện cơ bản chỉ dùng một khối Boolean làm điều kiện và không thể lồng ghép.

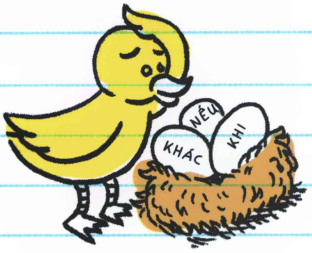
VÍ DỤ, bạn có thể dùng câu lệnh điều kiện cơ bản để cho phép người chơi chọn đường đi trong trò chơi:



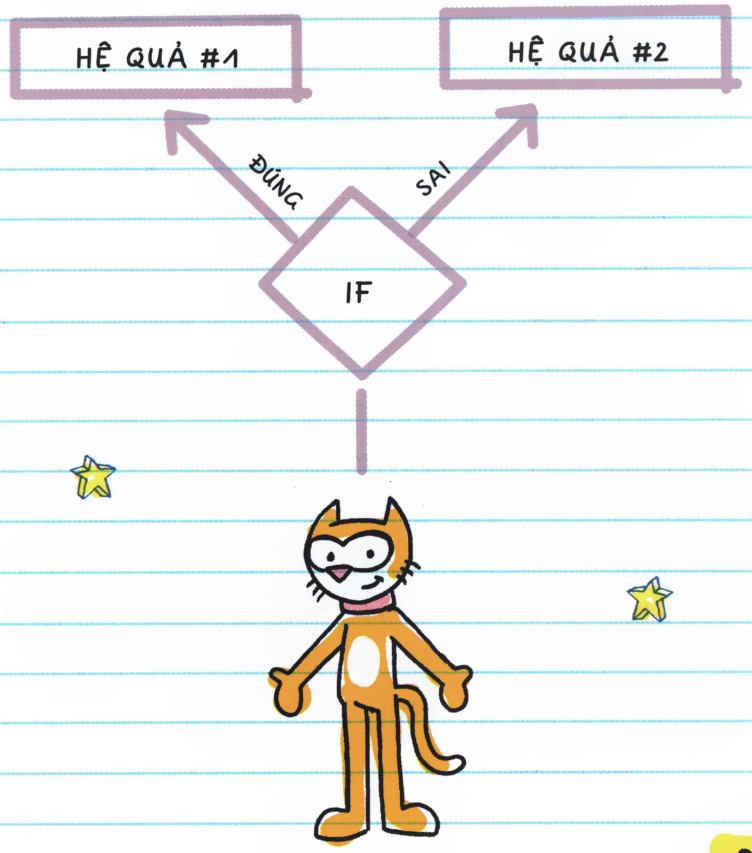
1. Kịch bản hỏi người chơi chọn cửa nào.
2. Khối điều kiện so sánh câu trả lời của người chơi với từ "Door 1" trong khối Boolean ( ) = ( ) .
3. Nếu người chơi trả lời "Door 1", thì khối nằm ở "miệng" đầu tiên sẽ chạy.
4. Nếu người chơi không trả lời "Door 1", thì code dưới từ "else" và nằm trong "miệng" 2 sẽ chạy.

Câu lệnh điều kiện phức và lồng ghép

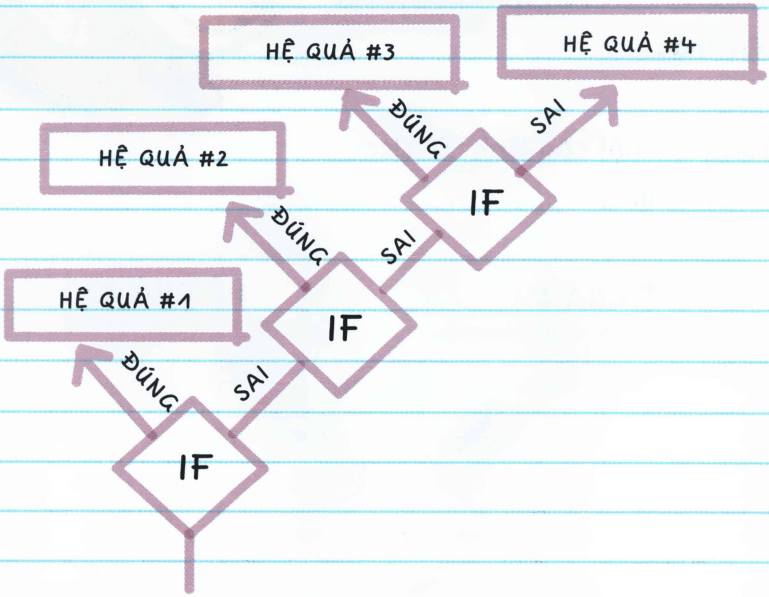
Khối điều kiện lồng bên trong khối điều kiện khác, tương tự như khối lặp. Điều kiện lồng ghép sinh ra khi bạn đặt một điều kiện bên trong một điều kiện khác.



Trong chỉ một câu lệnh "If else", có hai hệ quả.

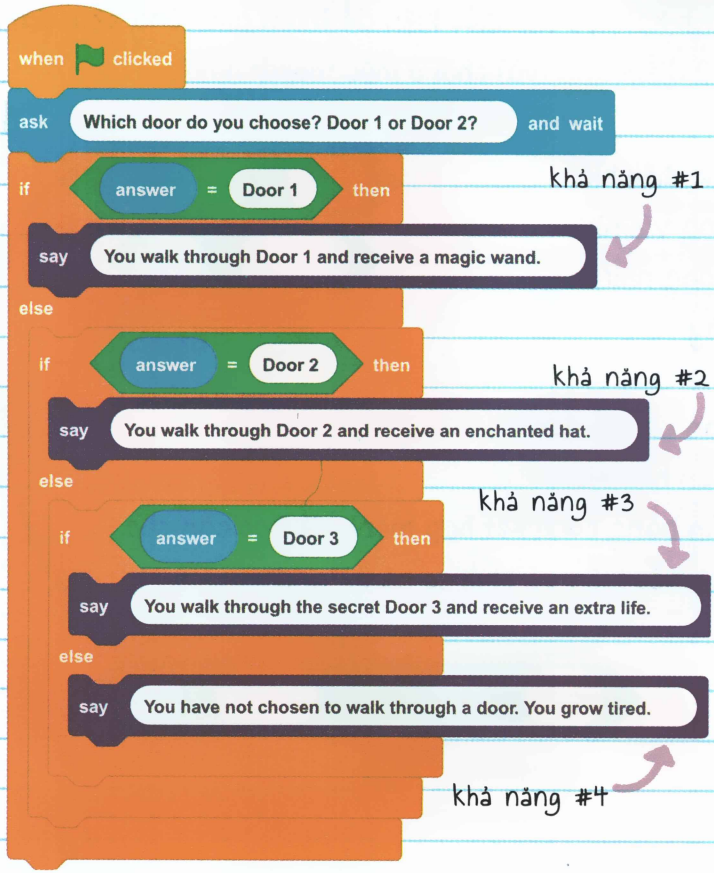


Khối điều kiện có thể được lồng trong khối điều kiện khác, tương tự như khối lặp. Khối điều kiện lồng ghép tạo ra hệ thống nhánh phức tạp, nơi nhiều kết quả khác nhau có thể xảy ra.



Khối điều kiện lồng ghép phức tạp có thể dùng để tạo trò chơi tự-chọn-cuộc-thám-hiểm-của-bạn. Bắt đầu với câu hỏi yêu cầu người chơi chọn cửa, điều kiện đầu tiên với khối Boolean đúng nếu người chơi chọn "Door 1". Trong trường hợp "else" (hoặc SAI) ở "miệng" 2 của khối điều kiện, chúng ta có thể thêm điều kiện "Door 2". Tiếp tục lồng khối điều kiện để thêm các khả năng.

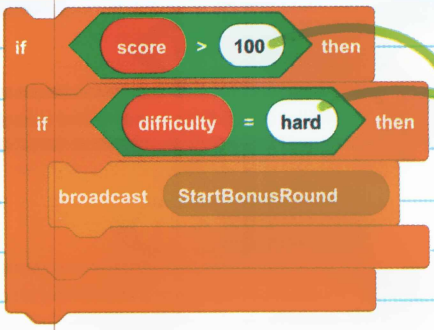
VÍ DỤ, trò chơi này có 4 khả năng:



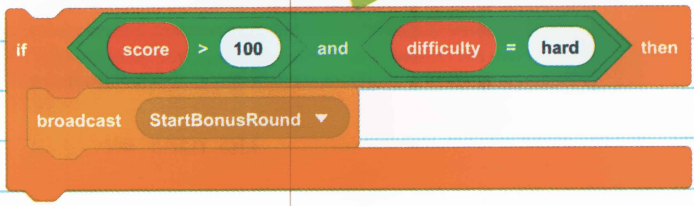
Câu lệnh điều kiện có thể được kết hợp để tiết kiệm diện tích và đơn giản hóa kịch bản. Khối Boolean **and** dùng để kết hợp hai khối Boolean.

VÍ DỤ. bạn có thể thêm vòng phụ cho trò chơi bằng cách kết hợp hai điều kiện để tham gia vòng phụ: đạt 100 điểm and (và) chơi ở mức "hard" (khó).

Các điều kiện lồng với nhau trông như thế này:



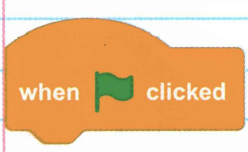
Bạn có thể đơn giản hóa kịch bản bằng cách kết hợp mệnh đề Boolean cho hai điều kiện và khối so sánh "and".



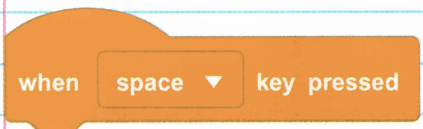
AND là đúng nếu cả hai điều kiện thỏa mãn;
OR là đúng khi một trong hai điều kiện thỏa mãn;
NOT biến mệnh đề đúng thành sai và sai thành đúng.

Sự Kiện

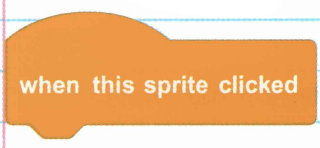
Khối trong danh mục Sự Kiện dùng để chạy kịch bản dựa trên sự kiện xảy ra khi chương trình chạy.



When green flag clicked: Khối mũ chạy kịch bản gắn lên khi nhấn vào cờ xanh.



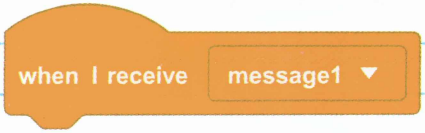
When ( ) key pressed: Khối mũ chạy kịch bản gắn liền khi nhấn phím được chọn.



When this sprite clicked: Khối mũ chạy kịch bản gắn liền khi nhấn vào đối tượng.

When backdrop switches to ( ): Khối mũ chạy kịch bản gắn liền khi phông nền chuyển sang phông nền được chọn.

When ( ) > ( ): Khối mũ chạy kịch bản gắn liền khi video hoặc giá trị thời gian trên mức đã đặt.



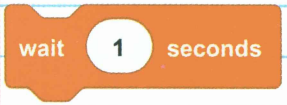
When I receive ( ): Khối mũ chạy kịch bản gắn liền khi nhận được tin nhắn được chọn.

Broadcast ( ): Khối xếp chồng phát tin nhắn được chọn tới mọi đối tượng.

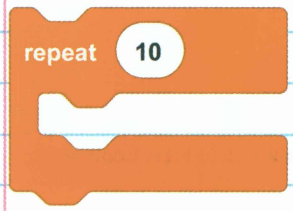
Broadcast ( ) and wait: Khối xếp chồng phát tin nhắn được chọn tới mọi đối tượng và chờ để chuyển sang khối tiếp.

Điều Khiển

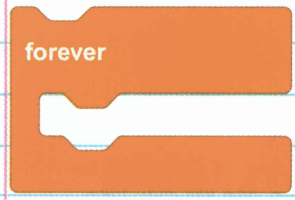
Danh mục Điều Khiển chứa khối dùng để điều khiển cách chương trình chạy.



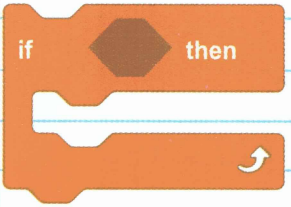
Wait ( ) seconds: Khối xếp chồng dừng kịch bản trong khoảng thời gian đã đặt.



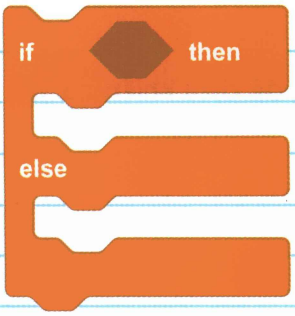
Repeat ( ): Khối C lặp code bên trong theo số lần đã đặt.



Forever: Khối C lặp code bên trong mãi mãi hoặc đến khi chương trình ngừng.

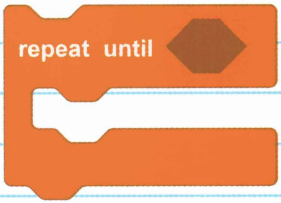


If ( ) then: Khối C chạy khi mệnh đề Boolean đúng.

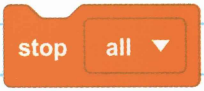


If ( ) then else: Khối C chạy phần 1 khi mệnh đề Boolean đúng, ngược lại thì chạy phần 2.

Wait until ( ): Khối xếp chồng dừng kịch bản đến khi điều kiện Boolean đúng.

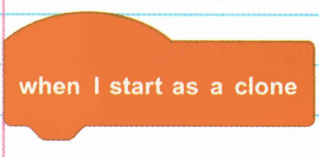


Repeat until ( ): Khối xếp chồng lặp khối bên trong đến khi điều kiện Boolean đúng.



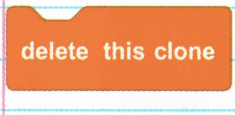
Stop ( ): Khối nắp dừng (những) kịch bản đã chọn.

Khối Bản Sao: Sao chép đối tượng là tạo một bản sao của nó. Sao chép hữu dụng khi bạn cần vài bản sao của cùng một đối tượng trên cùng một sân khấu.



When I start as a clone: Khối mũ chạy kịch bản gắn liền khi bản sao được tạo.

Create clone of ( ): Khối xếp chồng tạo bản sao của đối tượng được chọn.



Delete this clone: Khối nắp xóa bản sao.

KIỂM TRA KIẾN THỨC

1. Vòng lặp là gì? Tại sao chúng hữu dụng?

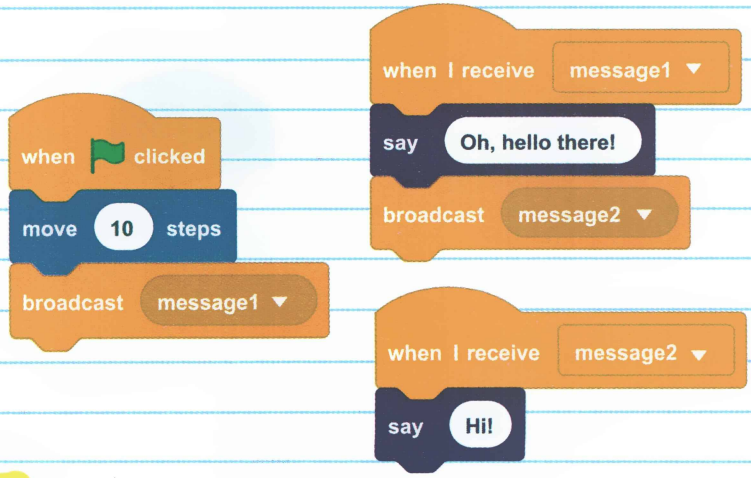
2. _____ lặp là vòng lặp trong vòng lặp khác.

3. Vòng lặp repeat ( ) khác gì vòng lặp forever?

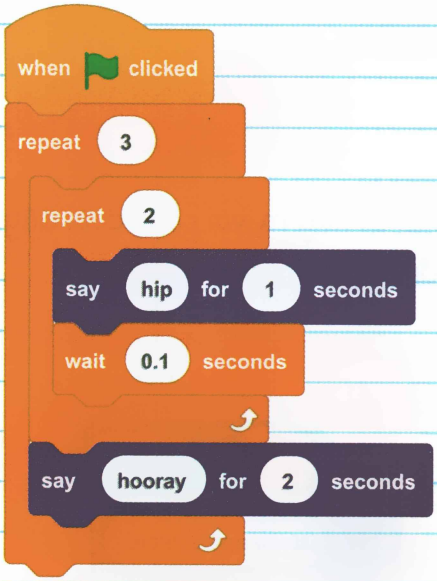
4. Khối phát tin broadcast ( ) làm gì? Sau khi chương trình phát tin nhắn, tin nhắn có hiển thị trên sân khấu?

5. Bạn dùng khối C nào để tạo vòng lặp liên tục đến khi người chơi đạt 10 điểm?

6. Trong code dưới đây, câu nào được hiển thị đầu tiên khi chương trình khởi động?

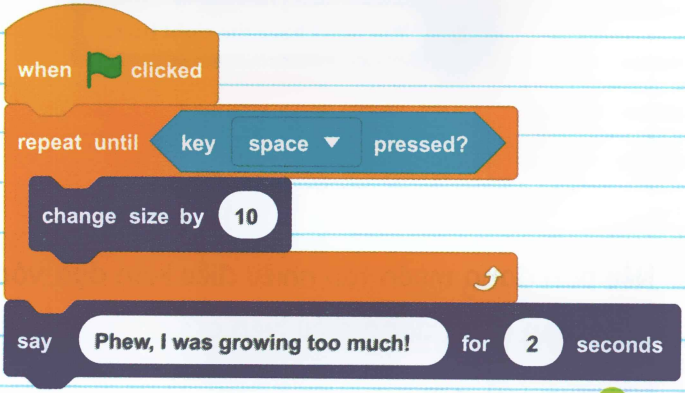


7. Từ "hip" được lặp lại bao lần?



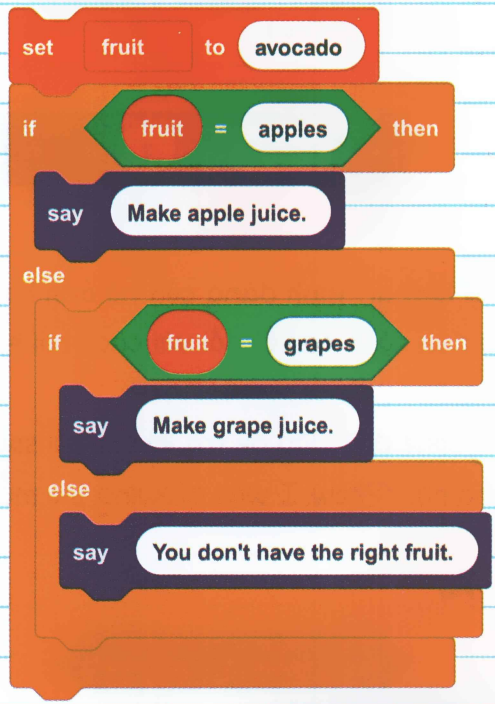
8. Khối nắp làm gì? Hình dáng của nó ảnh hưởng như thế nào đến cách nó kết nối với các khối khác?

9. Sử dụng code dưới đây để trả lời câu hỏi sau. Khi nào đối tượng nói: "Phew, I was growing too much!"?



10. Khối sự kiện dưới đây làm gì? Tại sao khối sự kiện quan trọng?

11. Chuyện gì sẽ xảy ra với code dưới đây nếu biến "fruit" được đặt thành "avocado"?



12. Nếu bạn đang muốn tạo nhiều điều kiện dựa vào lẫn nhau, bạn dùng dạng điều kiện gì?

ĐÁP ÁN

1. Vòng lặp là đoạn code lặp. Chúng hữu dụng vì chúng giúp code hiệu quả và dễ viết hơn, bởi bạn không phải sao chép kịch bản nhiều lần.

2. Lồng ghép

3. Vòng lặp repeat ( ) sẽ lặp lại code bên trong nó theo số lần đã đặt, nhưng vòng lặp forever sẽ chạy mãi mãi hoặc đến khi chương trình ngừng lại.

4. Khối Phát Tin phát tin nhắn để chạy khối sự kiện tương ứng với nó. Tin nhắn không hiện trên sân khấu và người dùng không nhìn thấy nó.

5. repeat until ( )

6. "Oh, hello there!"

7. 6 lần - vòng lặp trong lặp từ "hip" hai lần, và vòng lặp ngoài lặp vòng lặp trong 3 lần. 2 x 3 bằng 6.

8. Khối nắp kết thúc kịch bản, các khối khác không thể nằm dưới khối nắp.

9. Sau khi người dùng nhấn phím cách.

10. Khối Sự Kiện when this sprite clicked chạy khi người dùng nhấn vào đối tượng. Khối Sự Kiện cho phép người dùng tương tác. Chúng cho phép chương trình chạy nhiều kịch bản khác nhau tại các thời điểm khác nhau dựa trên thời điểm xảy ra sự kiện.

11. Đối tượng sẽ nói: "You don't have the right fruit".

12. Điều kiện lồng ghép.

Chương 23

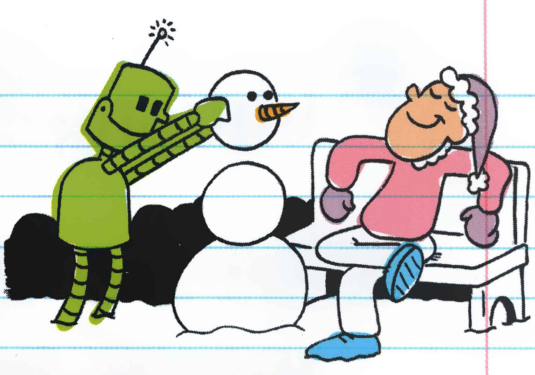
TÁI SỬ DỤNG KỊCH BẢN

TỰ TẠO KHỐI CỦA RIÊNG BẠN

Bạn có thể biến một phần lặp của kịch bản thành khối. Khối mới này gọi là **KHỐI THỦ TỤC**[1]. Bạn có thể tạo khối thủ tục mới với lệnh và hành vi của riêng nó.

> Thủ tục là đoạn code được đặt tên mà bạn có thể sử dụng nhiều lần. Trong Scratch, bạn có thể tạo khối thủ tục bằng cách tạo khối **define ( )** mới.

[1] Procedure Block.



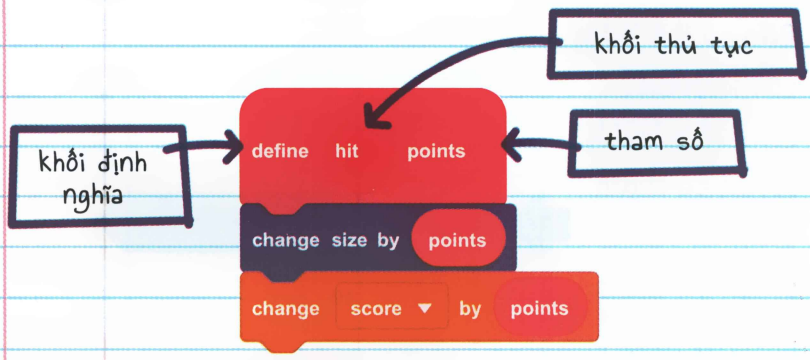
Khi bạn tạo thủ tục mới, bạn nhấn vào "Tạo một khối", và chọn dạng khối bạn muốn tạo. Sau đó bạn định nghĩa thủ tục với khối mũ định nghĩa (gọi là define ()) và thả khối code của bạn vào đó. Sau khi bạn tạo khối thủ tục mới, một KHỐI XẾP CHỒNG GỌI THỦ TỤC có cùng tên với thủ tục sẽ được thêm vào danh mục My Blocks (Khối Của Tôi) trong thanh công cụ khối. Bạn có thể **GỌI THỦ TỤC** (chạy nó) bằng khối xếp chồng *new block name* có cùng tên với thủ tục mới.

Ví dụ, bạn có thể dùng khối thủ tục để tạo một trò chơi mà đối tượng nhặt vật phẩm tốt và tránh vật phẩm tồi. Trò chơi sẽ cộng điểm khi đối tượng nhặt được vật phẩm tốt và trừ điểm khi đối tượng nhặt phải vật phẩm tồi. Trò chơi của bạn có 4 vật phẩm, mỗi vật phẩm tương ứng với số điểm khác nhau:

đá quý = 10 points
lọ thuốc = 5 points
dơi = -10 points
sét = -5 points

Trong trò chơi, dấu (-) nghĩa là bị mất điểm.

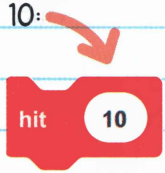
Bạn có thể tạo khối thủ tục gọi là **hit** và dùng khối mũ **define ( )** để định nghĩa thủ tục và thêm tham số gọi là "điểm":



Khối **change size by (points)** sẽ thay đổi kích cỡ đối tượng.

Giá trị tham số được điền vào ô tham số trong khối xếp chồng gọi thủ tục.

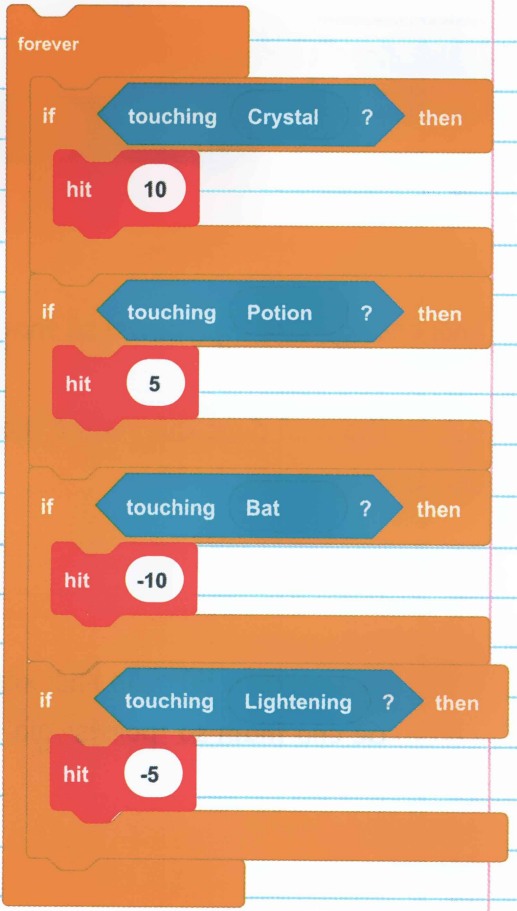
VÍ DỤ, bạn điền "10" vào khối xếp chồng gọi thủ tục **hit ( )** để cho tham số "points" giá trị là 10:



Nếu bạn chạy khối xếp chồng **hit (10)** với tham số "points" bằng 10, đối tượng sẽ to hơn 10% trên sân khấu. Nó cũng sẽ thêm 10 điểm vào biến "score".

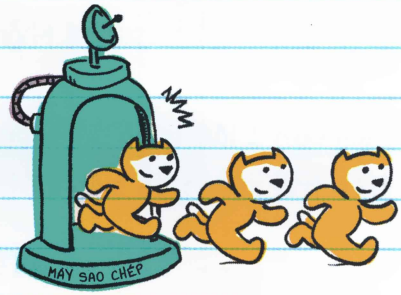
Bạn có thể thêm câu lệnh điều kiện để biết đối tượng có chạm vào mỗi vật phẩm và thêm khối xếp chồng thủ tục "hit" vào mỗi câu lệnh điều kiện.

Bạn có thể điền giá trị tham số khác nhau vào một khối xếp chồng thủ tục "hit" dựa trên giá trị của vật phẩm tương ứng.



SAO CHÉP

SAO CHÉP một đối tượng là tạo một bản sao của nó. Khối bản sao nằm ở danh mục Điều Khiển trong thanh công cụ khối.

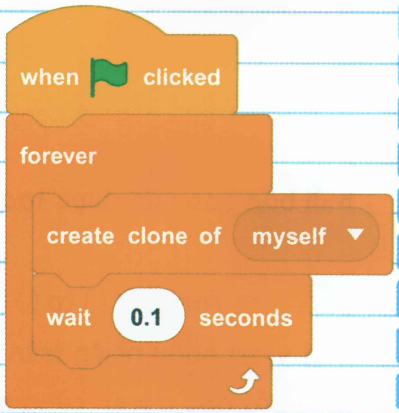


VÍ DỤ, nếu bạn muốn tạo mưa trên sân khấu, thay vì tạo hằng trăm mưa riêng lẻ, bạn có thể sao chép hằng trăm lần 1 hạt mưa duy nhất.

Để làm vậy, bạn cần 2 kịch bản.

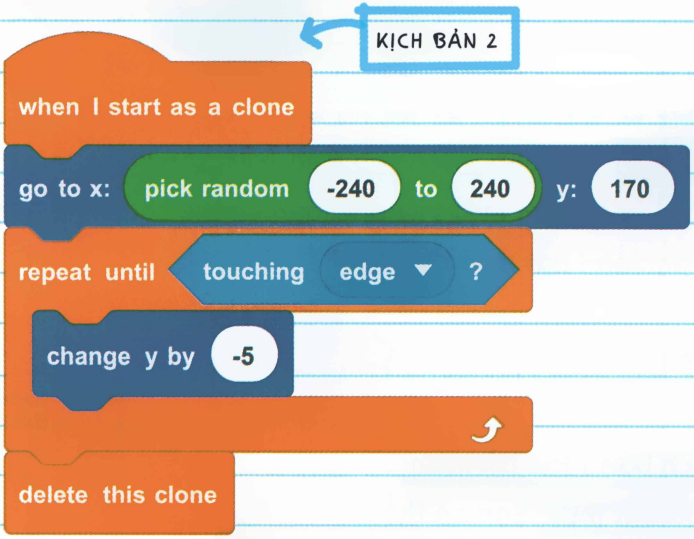
KỊCH BẢN 1

Kịch bản 1 tạo bản sao của hạt mưa. Bạn có thể dùng vòng lặp để mãi mãi liên tục tạo bảo sao mới:



Khi bạn chỉ chạy kịch bản 1, nó sẽ tạo vô số bản sao của đối tượng, nhưng bản sao sẽ không được lập trình làm gì cả.

Kịch bản 2 lập trình đối tượng trông như hạt mưa, rơi từ trên trời xuống và biến mất khi chạm đất.



- Khối mũ sự kiện when I start as a clone chạy cho mỗi bản sao mới được tạo.

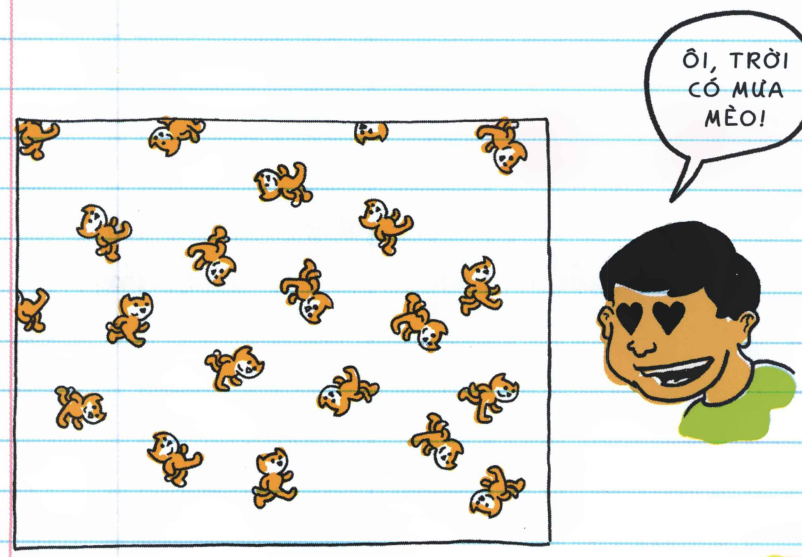
- Khối pick random () to () khiến một hạt mưa bắt đầu ở một vị trí ngẫu nhiên trên trục x. Bạn có thể khiến hạt mưa bắt đầu ở trên cùng màn hình bằng cách đặt vị trí trục y. Để đặt vị trí, thêm khối go to x: () y: () và thêm khối pick random () to () và điền các tham số.

- Thêm vòng lặp **repeat until** để hoạt hóa đối tượng rơi xuống cho đến khi chạm đất.

- Sử dụng khối **change y by ()** và điền giá trị tham số là -5 để khiến hạt mưa di chuyển từ trên xuống dưới.

- Khối **Delete this clone** sẽ xóa bản sao khi nó chạm đất.

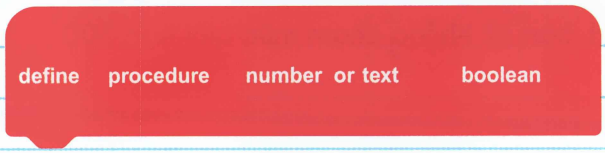
Giới hạn của Scratch là 300 bản sao. Khi số đối tượng trên màn hình chạm mức giới hạn, bản sao sẽ không được tạo ra nữa. Nhưng vì bạn xóa bản sao khi chúng chạm mặt đất, bạn sẽ không chạm mức giới hạn 300 bản sao.



Khối của tôi

Danh mục Khối Của Tôi khác với tất cả danh mục khác vì bạn có thể tạo số lượng khối mới không giới hạn. Với một khối bạn tạo, khối mũ **define ( )** và khối xếp chồng ***new block name*** xuất hiện trong menu danh mục.

Lưu ý: Khối trong danh mục Khối Của Tôi chỉ xuất hiện sau khi khối thủ tục mới đã được tạo.



Define ( ): Khối mũ định nghĩa khối thủ tục mới cùng với tham số cụ thể.

***New block name*:** Khối xếp chồng dùng để gọi, chạy thủ tục tương ứng. Khi khối này chạy, code phía dưới khối **define ( )** tương ứng sẽ chạy.

KIỂM TRA KIẾN THỨC

1. Nếu bạn muốn tạo vô số đối tượng, bạn cần dùng khối nào?

2. Việc tạo khối mới khác với việc dùng khối lặp như thế nào?

3. Tại sao bạn nên xóa bản sao khi bạn dùng xong?

4. Tại sao việc tạo khối mới tốt hơn việc sao chép và dán code?

5. Khi bản sao mới được tạo, nó chạy kịch bản nào?

ĐÁP ÁN

ĐÁP ÁN

1. Khối Bản Sao

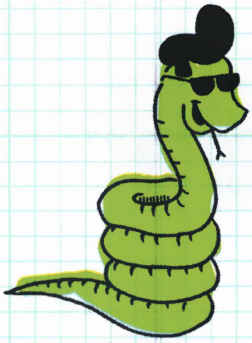
2. Khối lặp sẽ lặp lại đoạn code nhiều lần, trong khi khối mới bạn tạo có thể dùng ở các chỗ khác nhau. Bạn có thể điền thông tin khác nhau để thay đổi nhẹ cách khối hoạt động.

3. Giới hạn là 300 bản sao, và khi bạn chạm giới hạn thì sẽ không có bản sao nào được tạo ra.

4. Lập trình viên giỏi không sao chép và dán code. Code ngắn gọn, hiệu quả, chính xác tốt hơn vì nó dễ viết, sửa lỗi và tái sử dụng hơn.

5. Bản sao chạy bất kỳ kịch bản nào gắn liền với khối when I start as a clone.

BÀI 7



Lập trình bằng Python[1]

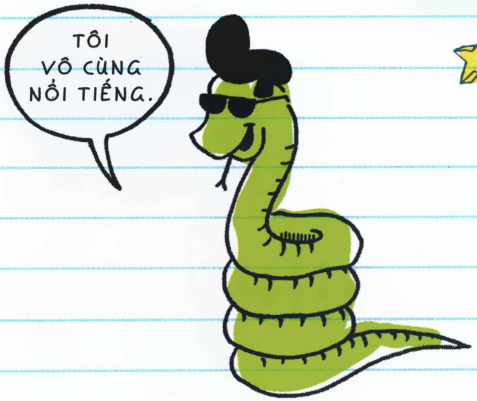
[1] Các code trong chương này được giữ nguyên từ sách tiếng anh – chú thích của BTV.

Chương 24

KHỞI ĐỘNG CÙNG PYTHON

GIỚI THIỆU PYTHON

Python là một trong những ngôn ngữ lập trình dễ học và được sử dụng nhiều nhất, vì vậy nó rất nổi tiếng với người mới. Nó có thể dùng cho nhiều loại dự án, từ thiết kế web đến kiểm tra lượng lớn dữ liệu.



Python có nhiều HÀM CÀI SẴN[1], hoặc công cụ, ví dụ như:

- Hàm toán với các phép tính cơ bản, căn bậc hai và chọn số ngẫu nhiên.
- Các lựa chọn để vẽ đồ họa.
- Thanh công cụ và các nút dễ sử dụng.

IDLE

Code Python được viết trong **CHƯƠNG TRÌNH MÔI TRƯỜNG PHÁT TRIỂN TÍCH HỢP (IDE)** giống như IDLE (Môi trường Học hỏi và Phát triển Tích hợp), nó đi cùng Python khi bạn cài đặt.

Chương trình IDLE được cài cùng Python. Có hai cửa sổ khác nhau: **CỬA SỔ SHELL** và **CỬA SỔ EDITOR**.

MÔI TRƯỜNG PHÁT TRIỂN TÍCH HỢP (IDE)[2]
Chương trình mà lập trình viên dùng để gõ và sửa code và để tạo chương trình Python.

[1] Built-in funtion.
[2] Integrated Development Environment.

Cửa sổ shell là cửa sổ xuất hiện khi bạn mở IDLE. Ở trên màn hình bạn sẽ thấy biểu tượng ">>>", đây là nơi bạn bắt đầu gõ.

Biểu tượng ">>>" gọi là prompt.

Để chạy code trong cửa sổ shell, nhập mã đó vào bên cạnh prompt và nhấn enter. **ĐẦU RA** của code sẽ xuất hiện ở dưới phía dưới, nếu code của bạn không chạy đúng cách, tin nhắn báo lỗi sẽ xuất hiện.

ĐẦU RA
Kết quả của code, thứ mà bạn sẽ thấy khi code chạy.

VÍ DỤ, bạn gõ vào code print("Hello,World!") và đầu ra sẽ là "Hello, World!" như sau:

```
>>> print("Hello, World!")
Hello, World!
>>>
```

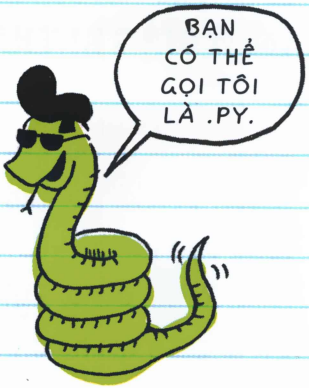
Cửa sổ shell rất phù hợp để chạy những đoạn code ngắn, nhưng không phù hợp để viết cả chương trình và bạn không thể lưu sản phẩm.

Cửa sổ editor dùng để viết toàn bộ chương trình và lưu trữ nó dưới dạng file. Cửa sổ editor khởi động là file hoàn toàn trống trơn, không có tiêu đề. File python có đuôi là .py, và bạn có thể lưu nó ở mọi chỗ trên máy tính. Đây là ví dụ về một file **firstproject.py**.

Đây là phần mở rộng của file
Nó đặt tên cho định dạng của file.

Khi làm việc với Python::

- Đặt tên file liên quan đến dự án của bạn.
- Lưu nó ở nơi bạn nhớ.
- Thường xuyên lưu sản phẩm của bạn.



MẸO LƯU FILE

Tạo một file mới gọi là "Dự án_Python" bên trong file "Tài liệu của tôi" (My documents) để lưu toàn bộ file dự án Python. Việc này sẽ giúp bạn ngăn nắp và nhớ chỗ tìm công việc của bạn.

CODE

Chỉ dẫn phải được viết bằng code. Mọi phần của code phải được viết theo luật của ngôn ngữ lập trình Python.

Luật viết chương trình gọi là syntax.

Python đi kèm hàm cài sẵn (những code thực hiện nhiệm vụ cụ thể). Một trong những hàm đó gọi là **Print**. Ở cửa sổ shell, hàm **print()** sẽ hiển thị văn bản mà bạn thêm vào là **GIÁ TRỊ THAM SỐ.**

GIÁ TRỊ THAM SỐ

Văn bản nằm trong dấu ngoặc của hàm.

Hello, Python

"Hello, World!" là một trong những chương trình cơ bản nhất bạn có thể viết bằng bất kỳ ngôn ngữ máy tính nào. Có thể dùng Python tạo chương trình "Hello, World!" bằng cách gọi hàm print và gõ "Hello, World!" là giá trị tham số:

Hàm print() ra lệnh cho máy tính hiển thị giá trị tham số.

Gọi hàm nghĩa là chỉ dẫn máy tính sử dụng công cụ cụ thể.

Cẩn thận khi gõ code. Nếu bạn đánh vần sai, chương trình sẽ không chạy đúng cách. Đừng thêm khoảng trống hoặc biểu tượng. Cần chú ý viết hoa. Ví dụ, gõ "Print" thay vì "print" sẽ khiến chương trình không hoạt động.

HÃY NGĂN NẮP

BÌNH LUẬN code là những lời nhắn cho bản thân và những lập trình viên khác. Bình luận không chạy như một phần chương trình nhưng bất kỳ ai đọc code đều có thể đọc

được bình luận. Chúng giống như những mảnh giấy nhớ nhỏ và giúp bạn sắp xếp suy nghĩ của mình. Bình luận có thể là lời nhắc, lời giải thích về code, và câu hỏi.

Để tạo bình luận trong Python, dùng biểu tượng # đối với bình luận một dòng, ba dấu ngoặc đơn (''') hoặc ba dấu ngoặc kép (""") đối với nhiều dòng.

Bình luận 1 dòng chỉ cần # đứng đầu, nhưng bình luận nhiều dòng mở đầu và kết thúc với dấu ngoặc kép.

```
# Comment on one
line
'''Multiline
comment can take
many rows.'''
```

Những màu sắc hữu ích

Cả cửa sổ shell và cửa sổ editor đều tự động thay đổi màu sắc của văn bản để giúp bạn đọc code.

Màu tím là cho hàm cài sẵn. Ví dụ, `print()` là hàm cài sẵn.

Màu cam là cho những từ khóa có ý nghĩa đặc biệt, ví dụ như "**if**" (nếu) hoặc "**True**" (đúng).

Màu xanh lá là cho những văn bản nằm trong dấu ngoặc kép. Dấu ngoặc đơn hoặc ngoặc kép đều dùng màu xanh lá miễn là chúng trùng khớp - nói cách khác, đừng bắt đầu với dấu ngoặc đơn và kết thúc với dấu ngoặc kép.

Màu xanh biển là cho văn bản đầu ra khi chương trình chạy.

Màu đỏ là cho tin nhắn báo lỗi xuất hiện khi chương trình không chạy đúng cách.

Màu đen là cho toàn bộ phần còn lại của chương trình.

KIỂM TRA KIẾN THỨC

1. Tại sao cửa sổ shell và cửa sổ editor tự động thay đổi màu sắc của văn bản bạn gõ?

2. Chương trình IDLE có nhiệm vụ gì? Tại sao nó hữu dụng?

3. Điền vào chỗ trống. Bạn nên dùng cửa sổ nào cho mỗi trường hợp dưới đây:

A. _____ Bạn muốn viết một chương trình lớn bằng Python.

B. _____ Bạn muốn lưu công việc.

C. _____ Bạn muốn chạy nhanh một đoạn code nhỏ.

4. Nối màu sắc ở cột phải với văn bản tương ứng ở cột trái.

A. Đầu ra	Màu tím
B. Văn bản trong ngoặc kép	Màu cam
C. Từ khóa	Màu xanh lá
D. Lỗi sai	Màu xanh biển
E. Hàm cài sẵn	Màu đỏ

5. Python (là/không phải là) ngôn ngữ lập trình dùng bởi những người chuyên nghiệp để tạo chương trình như Youtube hoặc Google.

6. Tại sao bạn nên bình luận vào code?

7. Bạn nền dùng hàm nào để hiển thị văn bản trong cửa sổ shell?

8. Thêm biểu tượng cạnh văn bản dưới đây để biến câu thành bình luận một dòng hoặc nhiều dòng.

Hãy biến câu này thành bình luận 1 dòng.

Hãy biến câu này
thành bình luận nhiều dòng.

ĐÁP ÁN

1. Màu sắc giúp bạn và những lập trình viên khác đọc code dễ hơn.

2. IDLE dùng để viết và chạy chương trình Python. Nó có thể dùng để thử nghiệm những đoạn code ngắn hoặc chạy file .py.

3. **A.** Editor
   B. Editor
   C. Shell

4. Màu tím: 5 Màu xanh lá: 2 Màu đỏ: 4
   Màu cam: 3 Màu xanh biển: 1

5. Là

6. Bình luận giúp bạn sắp xếp code ngăn nắp, và chúng giúp bạn và những lập trình viên khác hiểu nhiệm vụ của những phần code mà không cần phải đọc phần code đó.

7. print()

8. #Hãy biến câu này thành bình luận 1 dòng.

'''Hãy biến câu này
thành bình luận nhiều dòng.'''

hoặc bạn có thể dùng dấu ngoặc kép:

"""Hãy biến câu này
thành bình luận nhiều dòng."""

TÔI RẤT HỢP VỚI MÀU SẮC, BẠN CÓ NGHĨ VẬY KHÔNG?

Chương 25

BIẾN TRONG PYTHON

BIẾN dùng để lưu thông tin. Lưu thông tin hữu ích khi bạn muốn tái sử dụng thông tin nhiều lần hoặc dùng nó tại nhiều thời điểm khác nhau trong khi chạy chương trình. Chúng giúp bạn không phải gõ một khối code nhiều lần.

Trong Python, **DẠNG DỮ LIỆU** cơ bản có thể được lưu bằng biến bao gồm:

DẠNG DỮ LIỆU
Kiểu giá trị của dữ liệu.

- Chuỗi.
- Số nguyên.
- Danh sách.
- Giá trị Boolean.

Biến có thể lưu giá trị (nội dung hoặc dữ liệu), ví dụ như **CHUỖI**, số, **DANH SÁCH**, hoặc **BOOLEAN**.

DANH SÁCH
Một loạt các giá trị có thứ tự.

CHUỖI
Dạng dữ liệu thể hiện văn bản.

BOOLEAN
Dạng dữ liệu thể hiện "Đúng" hoặc "Sai".

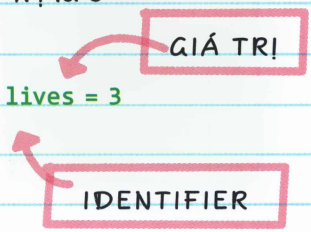
SỬ DỤNG BIẾN

Để gán giá trị cho biến, sử dụng định dạng sau:

```
identifier = value
```

IDENTIFIER là tên của biến.

VÍ DỤ, trong trò chơi mà người chơi có 3 mạng, bạn có thể đặt tên biến là "lives" (mạng) và gán cho nó giá trị là 3:



Trong Python, dấu "=" nghĩa là **gán cho**. "lives = 3" nghĩa là "gán 3 cho lives".

Sau khi tạo biến và gán cho nó giá trị bằng 3, bạn có thể sử dụng lại biến "lives" trong suốt chương trình.

Quy tắc đặt tên cho biến Python

Khi nghĩ ra tên, hay identifier, bạn phải tuân theo những quy tắc sau:

TỐT

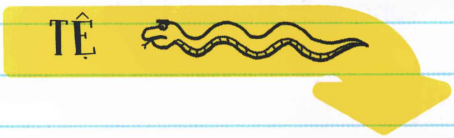
- Sử dụng số, chữ cái viết hoa, chữ cái thường và dấu gạch chân.
- Bắt đầu identifier với một chữ cái.
- Giữ identifier ngắn gọn và đơn giản. Identifier nên miêu tả được biến của nó.

CHUỘT_4

ĐIỂM

MẠNG

Khi nghĩ ra tên, hay identifier, **đừng làm điều sau:**



- Thêm khoảng cách hoặc biểu tượng (ví dụ như -, /, #, và @).
- Sử dụng từ khóa đã có nghĩa trong Python (ví dụ như "print").
- Sử dụng chữ cái O hoặc I viết hoa hoặc chữ I viết thường, vì chúng rất dễ gây nhầm lẫn - chữ O viết hoa trông như số 0.
- Bắt đầu tên biến với số hoặc biểu tượng.

Từ khóa trong Python (không được dùng làm tên biến):

and	def	finally	in	pass	while
as	del	for	is	print	with
assert	elif	from	lambda	raise	yield
break	else	global	not	return	
class	except	if	or	try	
continue	exec	import			

Khi đặt tên biến, hãy cố miêu tả biến.

VÍ DỤ. Khi bạn đặt tên biến lưu số mạng của người chơi là "j", bạn có thể sẽ quên "j" có nghĩa là gì khi bạn thêm nhiều biến khác.

Nhưng nếu bạn đặt tên biến là **"lives"**, bạn sẽ nhớ rằng đó là biến lưu số mạng của người chơi.

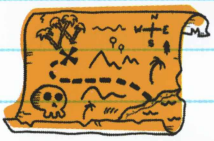
Hơn nữa, hãy giữ tên biến ngắn gọn. Tên biến dài dễ dẫn đến lỗi gõ máy và tốn thời gian viết các kỹ tự.

VÍ DỤ. mặc dù biến `the_number_of_lives_the_player_has_left` miêu tả dữ liệu nó lưu, bạn nên đặt tên biến ngắn hơn, ví dụ như `lives`.

Định dạng tên biến

Lập trình viên thường dùng **QUY ƯỚC ĐẶT TÊN** để khiến code của họ dễ đọc hơn. Quy ước đặt tên là định dạng của tên như tên biến. Python sẽ chạy mượt cho dù bạn chọn quy ước đặt tên nào. Tuy nhiên, tốt nhất nên cố định một cách để khiến code của bạn dễ đọc.

Những quy ước đặt tên thông dụng bao gồm:

MÔ TẢ	VÍ DỤ
Một chữ viết thường	x
Một chữ viết hoa	X 
Tất cả chữ thường	treasure
Viết thường và gạch dưới	x_position
Tất cả chữ viết hoa	CAPTAIN

Trong Python, nhiều người thích sử dụng **lowercase_and_underscores** để đặt tên biến.

Viết hoa và gạch dưới.	SHIP_NAME
Viết hoa chữ cái (Pascal case) Viết hoa chữ cái đầu tiên của mọi từ trong tên.	JollyRoger
Trường hợp lẫn lộn (camel case) Chỉ viết hoa chữ cái đầu tiên của từ thứ hai trong tên. Tưởng tượng nó giống như bướu lạc đà (camel) ở giữa tên vậy.	buriedTreasure
Viết hoa chữ cái đầu của từ và gạch dưới	Polly_Want_A_Cracker

KIỂM TRA KIẾN THỨC

1. Khai báo biến có tên là winning_number và cho nó giá trị là 1001.

2. Đáp án nào sau đây không phải là tên biến Python?
   A. @UserName
   B. handle
   C. Num_posts
   D. FriendCount

3. Trong Python, ký hiệu "=" nghĩa là gì?

4. Viết tên biến gồm 2 từ và sử dụng camel case.

5. Tại sao bạn nên chọn một quy ước đặt tên trong chương trình bạn đang làm?

6. Tên biến sau có gì sai?

date/Time

7. Tên biến sau có gì sai?

continue

8. Tên biến sau có gì sai?

1001I1

ĐÁP ÁN

1. `winning_number = 1001`

2. A

3. Nó có nghĩa là "gán cho".

4. Có thể trả lời là: exampleVariable.

5. Sử dụng cùng một quy ước đặt tên sẽ giúp bạn ngăn nắp và khiến code của bạn dễ đọc hơn vì mọi biến đều có chung một định dạng.

6. Python không cho phép dấu gạch chéo trong tên biến.

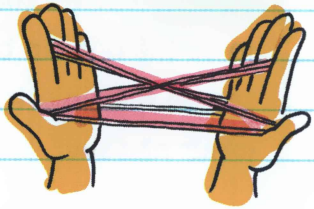
7. "coutinue" là từ khóa trong Python. Từ khóa không thể sử dụng làm tên biến.

8. Nó khó đọc vì 1s, 0s, *0s*, *Is*, và *ls* đều trông giống nhau.

Chương 26

CHUỖI

CHUỖI là một nhóm các ký tự được sắp xếp, nằm trong dấu ngoặc kép hoặc ngoặc đơn. Bạn dùng chuỗi để hiển thị tin nhắn hoặc văn bản trên màn hình trong chương trình của bạn. Khi Python thấy văn bản đặt trong dấu ngoặc kép, nó nhận diện đó là chuỗi.



VÍ DỤ: Tất cả những cụm từ sau đều là chuỗi và nó nằm trong dấu ngoặc kép/đơn:

"Game Over"

'time: 1:06'

"Lives left:"

"Welcome LOL. Can I tell you a joke?"

Đừng nhầm lẫn dấu ngoặc đơn và ngoặc kép. Ví dụ, **'board game night"** không đúng.

Để gán giá trị chuỗi cho biến, đặt tên biến, thêm toán tử gán (dấu bằng), sau đó thêm chuỗi nằm trong dấu ngoặc kép.

VÍ DỤ, đối với chương trình kể chuyện cười, chúng ta có thể đặt tên biến là "welcome, thêm toán tử gán, sau đó thêm giá trị "Welcome LOL. Can I tell you a joke?" như sau:

```
welcome = "Welcome LOL. Can I tell you a joke?"
```

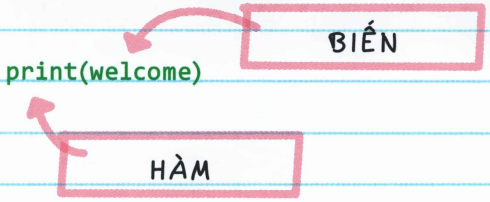
SỬ DỤNG HÀM PRINT VỚI CHUỖI

Để hiển thị giá trị biến, sử dụng hàm print:

- Đầu tiên, gán giá trị cho biến.

```
welcome = "Welcome LOL. Can I tell you a joke?"
```

- Sau đó, gõ tên biến vào trong dấu ngoặc của hàm print.

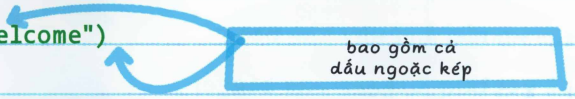


Cửa sổ shell sẽ hiển thị tin nhắn chào mừng:

```
Welcome LOL. Can I tell you a joke?
```

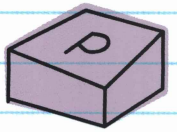
QUAN TRỌNG: Nếu bạn dùng dấu ngoặc kép xung quanh tên biến, giống như code dưới đây, kết quả sẽ khác.

```
welcome = "Welcome LOL. Can I tell you a joke?"
print("welcome")
```



Điều này sẽ khiến cửa sổ shell hiển thị: `welcome`

Khi bạn đặt văn bản trong dấu ngoặc kép, Python nhận diện nó là chuỗi thay vì biến. Vì vậy thay vì tìm giá trị biến, nó sẽ hiển thị từ "welcome" là chuỗi.



Hiển thị văn bản nhiều dòng

Để khiến chuỗi tự động hiển thị thành nhiều dòng, sử dụng 3 dấu nháy đơn (''') và gõ văn bản theo cách bạn muốn nó hiển thị.

VÍ DỤ:

```
joke = '''I was wondering why the ball
kept getting bigger and bigger,
and then it hit me.'''

print(joke)
```

Chương trình sẽ hiển thị:

```
I was wondering why the ball
kept getting bigger and bigger,
and then it hit me.
```

Đầu ra sẽ xuống dòng giống như chuỗi ban đã nhập.

Thay đổi giá trị của biến chuỗi

Bạn có thể thay đổi giá trị của biến bằng cách thêm dòng code gán giá trị mới cho biến, giống như cách bạn gán giá trị ban đầu.

VÍ DỤ. để thay thế "LOL" với "meh":

```
reaction = "lol"
```

Giá trị của reaction: lol

```
reaction = "meh"
```

Giá trị của reaction: meh

"meh" thay thế "lol" làm giá trị của reaction.

```
print(reaction)
```

Giá trị mới của reaction: meh

Chương trình sẽ hiển thị: `meh`

Ví dụ trên hiển thị "meh" vì nó là giá trị mới nhất của reaction.

Một khi giá trị của biến đã đổi, giá trị cũ bị xóa bỏ.

ĐỊNH DẠNG CHUỖI

Bạn có thể điều chỉnh chuỗi để chúng có dấu câu và cách trình bày như văn bản được nhập thông thường.

Dấu ngoặc kép

Mọi chuỗi đều có dấu ngoặc kép. Vậy nên nếu bạn muốn đặt dấu ngoặc kép vào trong chuỗi, bạn cần **THOÁT**[1] dấu ngoặc kép bên trong bằng cách đặt dấu gạch chéo quanh nó.

VÍ DỤ:

DẤU GẠCH CHÉO

```
tornado_joke = "What do you get if you say \"tornado"\
ten times backward and forward? . . . A real tongue
twister!"

print(tornado_joke)
```

Chương trình hiển thị:

```
What do you get if you say "tornado" ten times backward
and forward? . . . A real tongue twister!
```

Khi bạn **thoát một ký tự**, bạn đang bảo Python coi ký tự đó là một phần chuỗi thay vì là một ký tự đặc biệt.

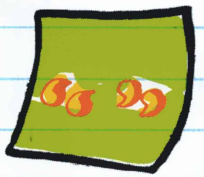
[1] escape.

NGOẠI LỆ! Nếu bạn dùng dấu ngoặc đơn ở đầu và cuối chuỗi, bạn có thể dùng dấu ngoặc kép làm 1 phần của chuỗi mà không cần thoát nó. (Ngược lại, nếu bạn dùng dấu ngoặc kép ở đầu và cuối, bạn có thể dùng dấu ngoặc đơn làm 1 phần của chuỗi).

VÍ DỤ, bạn có thể gán cho biến tornado_joke biến sau đây và có cùng kết quả trên:

```
tornado_joke = 'What do you get if you say "tornado" ten
times backward and forward? . . . A real tongue twister!'
```

Nhưng bạn vẫn nên biết cách thoát ký tự, để phòng trường hợp bạn cần dùng cả dấu ngoặc đơn và ngoặc kép hoặc các ký tự khác trong cùng một chuỗi.



Ngắt dòng

Để hiển thị dòng mới, sử dụng ký tự "n" để thoát, như sau:

```
print("\n")
```

VÍ DỤ, bạn có thể ngắt dòng câu đùa sau để tăng hiệu quả của trò đùa:

```
print("I had a talking parrot.\n But it didn't say it
was hungry,\n\n so it died.")
```

Chương trình sẽ hiển thị:

```
I had a talking parrot. But it
didn't say it was hungry,

    so it died.
```

TOÁN TỬ[1]

TOÁN TỬ là ký tự đại diện cho hành động điều khiển giá trị. Nhiều toán tử xuất phát từ toán học.

* là toán tử toán học đại diện cho phép nhân.
+ là toán tử toán học đại diện cho phép cộng.

Bạn có thể dùng toán tử với chuỗi cũng như số.

VÍ DỤ, để viết "Hello!" 3 lần, bạn có thể dùng:

```
print("Hello!" * 3)
```

TOÁN TỬ PHÉP NHÂN

"Hello!" * 3 nghĩa là "Hello!" x 3

Chương trình sẽ hiển thị: `Hello!Hello!Hello!`

Để thêm khoảng cách giữa các chữ "Hello!", thêm khoảng cách vào cuối chuỗi:

```
print("Hello! " * 3 )
```

KHOẢNG CÁCH

Chương trình sẽ hiển thị: `Hello! Hello! Hello!`

[1] Operator.

Cộng chuỗi

Toán tử có thể dùng để cộng chuỗi lại thành một câu.

VÍ DỤ:

KHOẢNG CÁCH

```
part1 = "Why did the chicken cross the road? "

part2 = "To get to the other side."

print(part1 + part2)
```

Chương trình sẽ hiển thị: `Why did the chicken cross the road? To get to the other side.`

Một cách khác để cộng chuỗi là tạo biến mới là tổng của của hai biến.

```
part1 = "Why did the chicken cross the road? "
part2 = "To get to the other side."
whole_joke = part1 + part2
```

Cách này sẽ gán cho biến whole_joke giá trị của `Why did the chicken cross the road? To get to the other side.`

Toán tử phép cộng có thể kết hợp cả biến và chuỗi thành giá trị biến chuỗi mới như sau:

```
animal = "alligator"
joke = "Why did the " + animal + " cross the road?"
print(joke)
```

Khoảng cách xung quanh dấu cộng giúp code dễ đọc hơn. Vì chúng không phải là một phần của chuỗi, máy tính sẽ bỏ qua nó. Bạn cũng có thể gõ như sau cũng đúng:

```
"Why did the "+animal+" cross the road?"
```

Và nó sẽ hiển thị cùng một dạng:

```
Why did the alligator cross the road?
```

Biến rất hữu dụng vì chúng thay đổi giá trị, giúp chương trình linh hoạt hơn.

Đôi khi bạn muốn người dùng điền thông tin.

Hàm **input()** hiển thị văn bản bạn thêm vào giữa dấu ngoặc và chờ người dùng gõ gì đó.

```
input("add your text here")
```

Cửa sổ shell sẽ hiển thị "add your text here" và chờ người dùng gõ gì đó vào cửa sổ và nhấn enter.

Bạn có thể lưu vào biến thông tin người dùng gõ trong cửa sổ shell.

Để làm điều này, trước hàm input, gõ tên biến bạn muốn dùng và thêm toán tử gán:

Biến lưu câu trả lời của người dùng.

```
variable = input("add your text here")
```

Văn bản hoặc câu hỏi hiển thị cho người dùng thấy.

VÍ DỤ. bạn dùng hàm input( ) để hỏi người dùng đưa giá trị mới cho biến "animal" ở ví dụ trên:

```
animal = input("What's your favorite animal? ")
joke = "Why did the " + animal + " cross the road?"
print(joke)
```

Chương trình sẽ hỏi, sau đó chờ người dùng gõ câu trả lời và nhấn ENTER.

```
What's your favorite animal? Alligator
```

Sau khi người dùng điền câu trả lời, dòng sau sẽ hiển thị với thông tin lưu trong biến "animal".

```
Why did the Alligator cross the road?
```

Đây là câu trả lời của người dùng.

Hàm chuỗi

Python có hàm cài sẵn có thể thay đổi chuỗi theo những cách hữu dụng. Có hàm viết hoa chữ cái đầu tiên của từ hoặc câu, ghép nhiều chuỗi thành một chuỗi, chuyển cả một chuỗi thành chữ viết thường, thay đổi một phần của

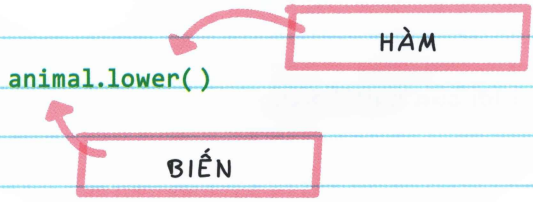
chuỗi, định dạng chuỗi thành kết quả đầu ra dễ nhìn, và nhiều hơn nữa.

HÀM CHUỖI thay đổi dữ liệu thành định dạng dễ sử dụng hơn. Ví dụ, nếu bạn hỏi người dùng động vật yêu thích của họ là gì và họ gõ viết hóa toàn bộ từ "ALLIGATOR" (cá sấu), hàm **lower()** có thể thay đổi chữ cái viết hoa thành viết thường.

Sử dụng hàm chuỗi bằng định dạng sau:

```
variable.function()
```

Nếu bạn có biến tên là "animal" và bạn muốn đảm bảo biến được viết thường, sử dụng hàm **lower()** với **tên** biến:



Bạn có thể dùng hàm biến với chuỗi ở hầu hết mọi nơi trong chương trình.

Ví dụ về các vị trí khác nhau để đặt hàm lower( ):

```
animal = input("What's your favorite animal? ").lower( )
print("Why did the " + animal + " cross the road?")

animal = input("What's your favorite animal? ")
animal = animal.lower( )
print("Why did the " + animal + " cross the road?")

animal = input("What's your favorite animal? ")
print("Why did the " + animal.lower( ) + " cross the road?")
```

Tất cả những ví dụ trên đều in ra cùng kết quả là:

```
Why did the alligator cross the road?
```

MỘT SỐ HÀM CHUỖI THÔNG DỤNG NHẤT

capitalize( ): Thay đổi chữ cái đầu thành viết hoa và viết thường phần còn lại.

lower( ): Thay đổi toàn bộ chữ viết hoa thành viết thường.

swapcase( ): Thay đổi chữ viết hoa thành viết thường và chữ viết thường thành viết hoa.

upper( ): Chuyển chữ viết thường thành viết hoa.

KIỂM TRA KIẾN THỨC

1. Đáp án nào sau đây không phải là chuỗi:

   A. "string cheese"

   B. 'shoelace'

   C. "She said, \"keep pulling the thread\""

   D. 5 + 9

2. Code Python dưới đây sẽ cho ra kết quả là gì?

```
art = "painting"
print("my favorite kind of art is: " + art)
```

3. Giải thích cách biến chuỗi thành nhiều dòng.

4. Nếu bạn muốn hỏi người dùng màu sắc yêu thích của họ là gì, bạn nên sử dụng hàm nào? Bạn viết nó như thế nào?

5. Thoát dấu ngoặc kép/đơn trong chuỗi là gì?

6. Bạn thêm code gì vào hàm print( ) dưới đây để nó ngắt dòng?

```
print(_____)
```

7. Kết quả của code sau đây là gì?

```
flavor1 = "chocolate"
flavor2 = "strawberry"
flavor3 = "vanilla"
print("My favorite sundae has these 3 flavors: " +
flavor1 + ", " + flavor2 + ", and " + flavor3 + ".")
```

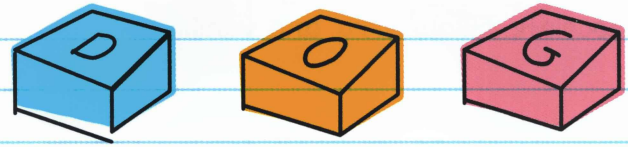
8. Giả sử bạn muốn hỏi người dùng điền tên chú chó của họ, bạn lưu biến tên là "dog". Bạn muốn viết hoa chữ cái đầu tiên của tên chú chó. Code nào sau đây sẽ KHÔNG viết hoa tên chú chó?

A. `capitalize( ) = dog`

B. `dog = dog.capitalize( )`

C. `dog = input("What's your dog's name?").capitalize( )`

D. `print("Your dog's name is: " + dog.capitalize( ))`



ĐÁP ÁN

1. D

2. my favorite kind of art is: painting.

3. Bạn có thể đặt chuỗi trong 3 dấu ngoặc đơn (''') và viết văn bản theo cách bạn muốn nó hiển thị. Hoặc bạn có thể chèn "\n" vào chỗ nào bạn muốn dòng mới bắt đầu.

4. Hàm input, viết như sau: input("What 's your favorite color?").

5. Khi bạn thoát dấu ngoặc kép/đơn, bạn biến nó thành một phần của chuỗi thay vì dùng nó như ký tự đặc biệt đánh dấu khởi đầu và kết thúc của chuỗi.

6. "\n"

7. My favorite sundae has these three flavors: chocolate, strawberry, and vanilla.

8. A

Chương 27

SỐ LÀM BIẾN

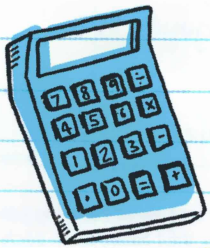
Biến cũng có thể là số. Khi gán một giá trị số cho biến, KHÔNG đặt số trong dấu ngoặc kép.

```
number = 5
string = "five"
string = "5"
```

Số có thể là chuỗi nếu bạn đặt nó trong dấu ngoặc kép. Nhưng bạn không thể làm toán với chuỗi.

TÍNH TOÁN VỚI BIẾN SỐ

Trong Python, bạn có thể dùng số để tính toán. Ví dụ, bạn tạo một chương trình hỗ trợ làm bài tập để Python tính toán.



Bạn có thể tạo 2 biến số, cộng chúng với nhau, và sau đó xuất kết quả là tổng của chúng:

> Hãy nhớ gán giá trị cho "num1" và "num2" *trước khi* bạn dùng chúng trong phép tính "sum". Bạn không thể dùng biến *cho đến khi* bạn gán giá trị cho chúng.

```
num1 = 4
num2 = 5
sum = num1 + num2
print(sum)
```

Kết quả sẽ là: 9

CÁC DẠNG SỐ

Python sử dụng các dạng số khác nhau. Hai dạng trong số đó là **SỐ NGUYÊN** và **PHẨY ĐỘNG**[1].

Số Nguyên

Trong Python, **Số Nguyên** là số nguyên dương hoặc số nguyên âm và không sử dụng dấu thập phân.

Bạn có thể tạo biến gọi là "num1" và gán cho nó giá trị số nguyên là 3:

```
num1 = 3
```

Số nguyên

[1] Float hoặc Floating point number. Dạng số này chứa dấu chấm tại Mỹ và dấu phẩy tại Việt Nam. Cuốn sách này sẽ dịch là "số dấu phẩy động" - chú thích của BTV.

Biến chuỗi dù có số bên trong cũng không được Python hiểu là số nguyên.

VÍ DỤ, khi bạn cộng hai chuỗi lại với nhau, chúng biến thành 1 chuỗi:

```
num1 = "4"
num2 = "3"
sum = num1 + num2
print(sum)
```

Số trong dấu ngoặc kép nghĩa là chuỗi.

Kết quả nhận được sẽ là 43, là chuỗi kết hợp giữa chuỗi 3 và chuỗi 4.

Đôi khi bạn cần chuyển một biến thành dạng **dữ liệu số nguyên**.

VÍ DỤ, nếu bạn hỏi người dùng nhập số bằng hàm **input( )**, Python sẽ lưu câu hỏi của họ là chuỗi.

```
width = input("Enter the rectangle's width. ")
height = input("Enter the rectangle's height. ")
```

Trong chương trình trên, nếu người dùng nhập 5 và 8, số sẽ được lưu là chuỗi - "5" và "8" - chứ không phải số nguyên.

Chuyển câu trả lời của người dùng từ chuỗi thành số nguyên bằng cách dùng hàm int(). Bạn có thể thực hiện phép tính với chúng.

Nhân chiều dài và chiều rộng của hình chữ nhật để biết diện tích, làm như sau:

Chiều dài và chiều rộng đang là chuỗi.

```
width = input("Enter the rectangle's width. ")
height = input("Enter the rectangle's height. ")
width = int(width)
height = int(height)
print(width*height)
```

Chiều dài và chiều rộng được chinh lại là số nguyên.

Giá trị số cho chiều dài và chiều rộng đã có thể dùng để tính toán, ví dụ như tính diện tích.

Số dấu phẩy động

Số dấu phẩy động là số có dấu thập phân.

VÍ DỤ, giá trị của những biến sau là số dấu phẩy động:

```
num1 = 3.3
num2 = 6.0
num3 = 2.9564576
```

SỐ DẤU PHẨY ĐỘNG

Để chuyển một giá trị biến từ số nguyên hoặc chuỗi sang số dấu phẩy động, sử dụng hàm **float()** như sau:

```
num1 = 6
```
num1 đang là số nguyên.

```
num1 = float(num1)
```
num1 được gán lại thành số dấu phẩy động.

```
sum = num1 + 3.3
```
Phép tính có số dấu phẩy động thì để chữ số ở bên phải dấu thập phân.

```
print(sum)
```
Sẽ hiển thị: 9.3 (tổng của 6 + 3.3).

CHUYỂN ĐỔI SỐ

int(): Chuyển một giá trị thành số nguyên. Hữu dụng để chuyển câu trả lời của người dùng từ chuỗi sang số nguyên để làm toán.

float(): Chuyển một giá trị sang số dấy phẩy động (số có dấu thập phân). Hữu dụng khi bạn muốn dùng giá trị không phải là số nguyên.

BIỂU THỨC TOÁN HỌC

Số và toán được dùng trong chương trình để thể hiện vị trí của nhân vật trong trò chơi, tính toán ngày, đặt thời gian, hiển thị màu sắc, tính điểm, và nhiều hơn nữa.

GIÁ TRỊ và **TOÁN TỬ** tạo thành **BIỂU THỨC**.

BIỂU THỨC

Sự kết hợp của giá trị và toán tử mà có thể tạo thành một giá trị duy nhất. Ví dụ, 2 + 3 là biểu thức có kết quả (bằng) 5.

VÍ DỤ VỀ BIỂU THỨC:

```
4 * 3
3 / 4 * 5 - 5 / 2
(7 * 3) - (5 - 7) / (4 + 5)
```

GIÁ TRỊ

Một mảnh thông tin. Trong biểu thức, giá trị là số.

Biến có thể chứa giá giá trị số và dùng trong biểu thức toán học. Ví dụ, nếu một biến gọi là "x" chứa giá trị là 4, nó có thể dùng trong biểu thức:

`x + 6` Vì x chứa giá trị là 4, biểu thức này có kết quả là 10.

Có 7 toán tử toán học thông dụng:

TÊN	BIỂU TƯỢNG	VÍ DỤ
Cộng	+	5 + 2
Trừ	-	5 - 2
Nhân	*	5 * 2
Chia	/	5 / 2
Phép toán Modulo[1]	%	5 % 2
Số mũ	**	5 ** 2
Phép chia làm tròn xuống[2]	//	5 // 2

Phần còn lại của thương số. Ví dụ, 5 ÷ 2 = 2 với phần dư hoặc modulo là 1.

Chia hai số và bỏ qua phần dư. Ví dụ, 5 ÷ 2 = 2 (phần dư, 1, được bỏ qua).

Biểu tượng "x" và "÷" không được dùng trong Python.

[1] Modulus: Phép toán modulo là phép toán tìm số dư của phép chia 2 số (trong ví dụ là tìm số dư của phép chia 5 : 2, tức là bằng 1).

[2] Floor division: Phép chia làm tròn xuống. Ví dụ, 5 : 2 = 2 (số dư là 1 bị bỏ qua).

THỨ TỰ TOÁN TỬ

Sử dụng **THỨ TỰ TOÁN TỬ** khi giải phương trình:

THỨ TỰ TOÁN TỬ
Quy luật quyết định thứ tự toán tử được thực hiện.

1. Giải bất kỳ phần nào nằm trong ngoặc của phương trình. Nếu có nhiều cặp ngoặc, giải cặp ngoặc ở trong cùng trước.

Ngoặc có thể được *lồng ghép*. Nghĩa là có một cặp ngoặc nằm trong một cặp ngoặc khác, ví dụ như thế này:
((2 + 18) × 5) 10.

Cặp ngoặc trong cùng là 2 + 18 cần được giải trước.

2. Tính số mũ, giá trị tuyệt đối và căn bậc hai từ trái sang phải.
3. Giải tất cả phép nhân và chia từ trái sang phải.
4. Giải phép cộng và trừ từ trái sang phải.

Để nhớ thứ tự trên, hãy nhớ "Please Excuse My Dear Aunt Sally", hoặc PEMDAS (Parentheses, Exponents, Multiplication, Division, Addition, Subtraction[1]). Hãy cẩn thận, PEMDAS vẫn có thể sai sót. Ví dụ, bạn nên làm phép trừ trước phép cộng nếu bạn tính toán từ trái sang phải – và tương tự với phép chia và phép nhân.

[1] Dấu ngoặc đơn, Số mũ, Nhân, Chia, Cộng, Trừ.

VÍ DỤ. để giải biểu thức `2 * (6 - 3)`:

Đầu tiên, giải phương trình trong ngoặc: `6 - 3 = 3` → `2 * (6 - 3)` → `2 * (3)`

Sau đó, thực hiện phép nhân: `2 * 3 = 6`

Nếu bạn có biểu thức tương tự nhưng không có dấu ngoặc: `2 * 6 - 3`

Đầu tiên, nhân: `2 * 6 = 12` → `12 - 3`

Sau đó, thực hiện phép trừ để ra kết quả: `12 - 3 = 9`

Cho dù có cùng số, hai phương trình ra kết quả khác nhau vì thứ tự toán tử. Python sẽ luôn tuân theo thứ tự toán tử khi trả giá trị.

Xuất biểu thức toán học

Nếu biểu thức ở trong hàm function, thì Python sẽ xuất giá trị của biểu thức ở cửa sổ shell.

VÍ DỤ:

```
print(5 * 3)
```

Kết quả xuất ra là: `15`

KIỂM TRA KIẾN THỨC

1. Trong Python, toán tử ** có nhiệm vụ gì?

2. Để tìm số dư của thương số, bạn dùng toán tử _____.

3. Kết quả của mỗi biểu thức sau là?

   A. 5 // 4

   B. 3 * (5 + 2) - 6

   C. 2 * 5

   D. 4 / 2

   E. 6 % 2

   F. 6 / 3 + (2 - 1) + 2 * 3

4. Viết code xuất kết quả của 5 nhân 2.

5. Theo thứ tự của toán tử, số mũ hay phép cộng đứng trước?

6. Biểu thức là gì?

7. Trong biểu thức toán học, giá trị luôn là (số/biểu tượng).

8. Nếu bạn muốn có giá trị số nguyên của thương số và bỏ qua số dư, bạn nên dùng toán tử _____.

9. Code sau sẽ xuất kết quả là?

```
width = 4
height = 12
perimeter = width * 2 + height * 2
print(perimeter)
```

10. Lỗi sai của chương trình sau nằm ở đâu?

```
dog_years = human_years * 7
human_years = 3
print(dog_years)
```

11. Sự khác biệt giữa biến số dấu phẩy động và số nguyên?

12. Chương trình sau sẽ xuất ra số nguyên hay số dấu phẩy động?

```
d = 4.2
t = 3
speed = d / t
print(speed)
```

13. Bạn dùng hàm nào để chuyển một biến thành số nguyên?

ĐÁP ÁN

1. ** tượng trưng cho toán tử số mũ.

2. Modulo hoặc %

3. **A.** 1
   B. 15
   C. 10
   D. 2
   E. 0
   F. 9

4. print(5 * 2)

5. Số mũ

6. Biểu thức là sự kết hợp của giá trị và toán tử, có thể cho ra kết quả là một giá trị duy nhất.

7. Số

8. Floor hoặc //

9. 32

10. Vấn đề là giá trị 3 được gán cho biến "human_years" sau khi biến được dùng ở dòng đầu (human_years * 7).

11. Số nguyên là số không có dấu thập phân, và số dấu phẩy động là số có dấu thập phân.

12. Số dấu phẩy động, vì "d" là số dấu phẩy động và dùng để tính toán "speed".

13. int()

Chương 28

DANH SÁCH VÀ BIỂU THỨC BOOLEAN

DANH SÁCH

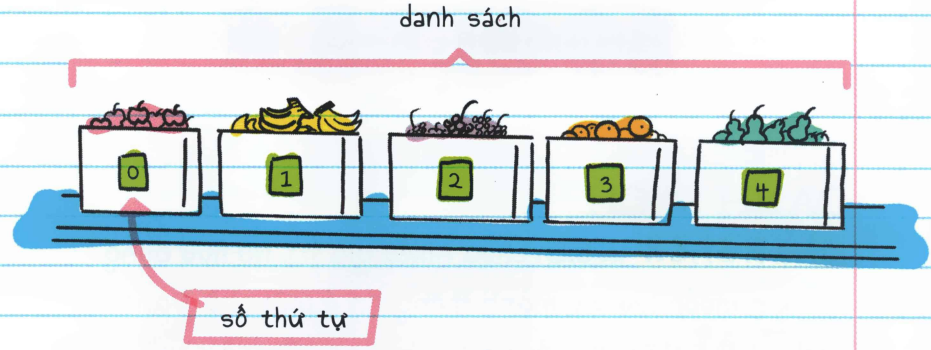
DANH SÁCH là biến chứa nhiều giá trị. Nó hữu dụng khi bạn muốn lưu vài mảnh thông tin ở một chỗ. Danh sách có thể chứa mọi dạng giá trị bao gồm số, chuỗi và các danh sách khác.

Trong Python, danh sách được định dạng bằng ngoặc vuông (đôi khi là ngoặc kép), với dấu phẩy giữa mỗi phần tử. Ví dụ, chúng ta có thể tạo một danh sách gọi là "fruit" và gán cho nó giá trị của apples, bananas, grapes, oranges, and pears:

```
fruits = ['apples', 'bananas', 'grapes', 'oranges', 'pears']
```

> Lưu ý: Mỗi phần tử được chia cách bởi dấu phẩy, nhưng hãy chắc chắn rằng dấu phẩy để chúng không bị thêm vào chuỗi.

Hãy coi danh sách như một hàng hộp, mỗi hộp chứa một giá trị. Mỗi hộp trong danh sách có một số, bắt đầu từ 0. Những số này gọi là **SỐ THỨ TỰ**[1].



Số thứ tự dùng để tìm hoặc thay đổi giá trị hoặc phần tử cụ thể trong danh sách. Ở ví dụ biến fruits, apples có số thự tự là 0, và số thứ tự của pears là 4.

Một phần tử cụ thể trong danh sách được đặt dưới định dạng `list[#]`, trong đó "list" là tên danh sách và "#" là số thứ tự phần tử.

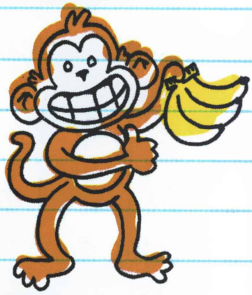
Vậy `fruits[0]` là apples và `fruits[4]` là pears.

[1] Index.

Để hiển thị phần tử thứ hai trong danh sách, sử dụng hàm print, điền "fruits" là tên danh sách, và "1" vào vị trí số thứ tự phần tử:

```
print(fruits[1])
```

Vì số thứ tự bắt đầu từ 0, phần tử thứ hai sẽ là 1.



Code này sẽ xuất kết quả là: `bananas`

Bạn cũng có thể xuất một phần danh sách bằng sách điền khoảng số thứ tự cần xuất, như thế này:

số thứ tự kết thúc

```
print(fruits[1:4])
```

số thứ tự bắt đầu

Code này sẽ xuất kết quả là: `['bananas', 'grapes', 'oranges']`

Số thứ tự cuối cùng (4 : pears) không được tính.

Khi bạn dùng hàm **print()** để hiển thị nhiều hơn một phần tử trong danh sách, đầu ra sẽ bao gồm ngoặc vuông và dấu ngoặc đơn. Đó là vì bạn đang xuất một danh sách, không phải là một phần tử trong danh sách.

Có nhiều cách để cập nhật danh sách.

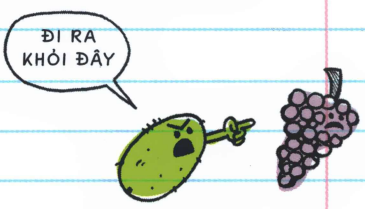
Bạn có thể thay thế phần tử trong danh sách sau khi danh sách đã được tạo.

Phần tử cuối cùng không được liệt kê – chỉ những phần tử ở trước số thứ tự kết thúc mới được tính vào danh sách.

- Để thay thế một giá trị của một phần tử trong danh sách.
  1. Nhắc đến số thự tự của phần tử
  2. Gán giá trị mới

Để thay thế grapes với kiwi: `['apples', 'bananas', 'grapes', 'oranges', 'pears']`

Số thứ tự của grapes là 2. Bạn có thể thay thế grapes bằng cách gán số thứ tự 2 của danh sách fruits cho kiwi:




```
fruits[2] = "kiwi"
```

Bây giờ danh sách là:

```
['apples', 'bananas', 'kiwi', 'oranges', 'pears']
```

- Để thêm phần tử vào cuối danh sách, sử dụng hàm **append()**.

Append nghĩa là thêm.

Nếu danh sách fruits là danh sách mua sắm, hàm **append()** sẽ thêm phần tử vào cuối nó.

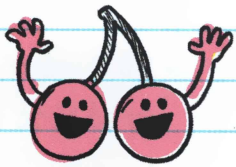
Bạn có thể thêm "cherries" vào danh sách bằng cách dùng hàm append với danh sách fruits như sau:

```
fruits.append("cherries")
```

Dấu phẩy được thêm tự động khi bạn thêm phần tử mới vào danh sách.

Bây giờ danh sách là: `['apples', 'bananas', 'kiwi', 'oranges', 'pears', 'cherries']`

- Để chèn một phần tử giữa hai giá trị trong danh sách, sử dụng hàm **insert()**. Bên trong ngoặc, cho hàm biết nơi chèn giá trị mới và giá trị mới là gì.



Chèn "peaches" vào vị trí số thứ tự 2 của danh sách fruits bằng cách dùng hàm **insert()** như sau:

```
fruits.insert(2, "peaches")
```

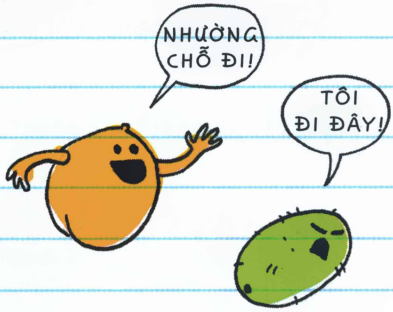
vị trí số thứ tự của phần tử mới.

giá trị cần chèn.

Danh sách giờ là: `['apples', 'bananas', 'peaches', 'kiwi', 'oranges', 'pears', 'cherries']`

- Để xóa một phần tử trong danh sách, sử dụng hàm **remove()**. Bên trong ngoặc, cho hàm biết bạn muốn xóa phần tử nào bằng cách điền giá trị (tên của phần tử, không phải số thứ tự):

```
fruits.remove("kiwi")
```

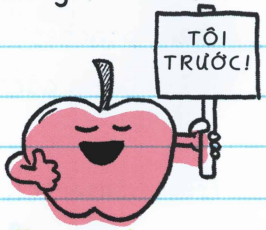


Danh sách giờ là: `['apples', 'bananas', 'peaches', 'oranges', 'pears', 'cherries']`

- Để sắp xếp danh sách,

  - Sử dụng hàm **sort()** nó sẽ xếp danh sách theo thứ tự số hoặc thứ tự chữ cái (A-Z), hoặc
  - Sử dụng hàm **reverse()**, nó sẽ đảo ngược thứ tự danh sách.

```
fruits.sort( )
```



Thứ tự danh sách giờ là: `['apples', 'bananas', 'cherries', 'oranges', 'peaches', 'pears']`

```
fruits.reverse( )
```

Thứ tự danh sách giờ là: `['pears', 'peaches', 'oranges', 'cherries', 'bananas', 'apples']`

- Để biết số phần tử trong danh sách, sử dụng hàm `len( )`:

```
fruits = ['pears', 'peaches', 'oranges', 'cherries',
'bananas', 'apples']
fruit_length = len(fruits)
print(fruit_length)
```

Điền tên danh sách vào dấu ngoặc.

Kết quả xuất ra: `6`

- Để thêm một danh sách vào cuối danh sách khác, sử dụng toán tử `"+"`:

```
fruits = ['apples', 'bananas', 'kiwi']
vegetables = ['carrots', 'peas', 'onions']
produce = fruits + vegetables
print(produce)
```

Đặt tên cho biến mới (danh sách kết hợp).

Kết quả: `['apples', 'bananas', 'kiwi', 'carrots', 'peas', 'onions']`

DANH SÁCH HÀM

list[#]: Nhắc đến một phần tử cụ thể trong danh sách, trong đó "#" là số thứ tự của phần tử và "list" là tên danh sách.

append("item"): Thêm phần tử vào cuối danh sách.

insert(#, "item"): Chèn phần tử vào vị trí số thứ tự cụ thể trong danh sách.

remove("item"): Xóa một phần tử trong danh sách.

sort( ): Sắp xếp danh sách theo thứ tự số hoặc thứ tự bảng chữ cái.

reverse( ): Đảo ngược thứ tự danh sách.

len( ): Cho biết số phần tử có trong dánh sách.

Danh sách trong danh sách

Danh sách có thể ở trong danh sách khác. Danh sách thứ hai được gọi là **DANH SÁCH TRONG**. Bạn có thể sử dụng danh sách này để thể hiện các danh mục phụ - một tổ hợp chuyên biệt hơn. Định dạng để tạo danh sách trong danh sách là cặp ngoặc vuông bên trong cặp ngoặc vuông.

Danh sách ngoài — Danh sách ngoài tiếp tục

```
list = ["A", "B", "C", ["D1", "D2", "D3"], "E"]
```

Danh sách trong, nằm ở vị trí phần thứ tư của danh sách ngoài.

VÍ DỤ, để liệt kê những loại táo (apples) yêu thích trong danh sách hoa quả yêu thích (fruits):

1. Mở ngoặc vuông
2. Thêm những loại apples vào trong dấu ngoặc kép, và đóng ngoặc vuông
3. Tiếp tục với phần còn lại của danh sách

Danh sách ngoài — Danh sách trong

```
fruits = ["bananas", ["Gala", "Empire", "McIntosh",
"Golden Delicious"], "kiwi", "oranges", "pears"]
```

Danh sách trong — Danh sách ngoài tiếp tục

Danh sách apples được chứa ở phần tử thứ hai của danh sách fruits.

Để chỉ xuất apples yêu thích (danh sách trong) từ danh sách fruits, sử dụng hàm print với phần tử thứ hai (vị trí số thứ tự là 1) của danh sách fruits:

```
print(fruits[1])
```

Phần tử thứ hai

Tên danh sách

Xuất: `["Gala", "Empire", "McIntosh", "Golden Delicious"]`

Bạn có thể chỉ xuất tên một loại apple bằng cách dùng vị trí số thứ tự 1 để nhắc đến danh sách apples và sau đó thêm vị trí số thứ tự của loại apple cụ thể.

Ví dụ, để xuất "McIntosh" trong danh sách apple, sử dụng code sau:

Apple cụ thể

```
print(fruits[1][2])
```

Danh sách apple

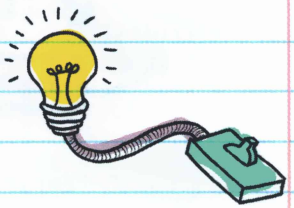
Xuất: McIntosh

BIỂU THỨC BOOLEAN

Trong lập trình, bạn thường muốn biết một điều là đúng hay sai. Ví dụ, ta muốn biết trò chơi kết thúc chưa hay nó vẫn tiếp tục, hoặc ta muốn biết người chơi trả lời đúng hay sai.

Python tự động nhận biết biến dạng Boolean khi giá trị được quy về "True" (Đúng) hoặc "False" (Sai), (biểu thức Boolean). Khi bạn quy giá trị về "True" or "False", hãy chắc chắn bạn viết hoa chữ T ở True và F ở False. Ví dụ: my_value = True.

Biến Boolean giống như bóng đèn. Chỉ có hai lựa chọn: bật hoặc tắt (True hoặc False).



Khi bạn gán biểu thức Boolean cho biến, Python sẽ quy giá trị về "True" hoặc "False" dựa trên liệu biểu thức Boolean đúng hay sai. Ví dụ:

```
height = 58
meet_limit = height > 50
print(meet_limit)
```

Gán giá trị "58" cho "height".

Gán giá trị "height > 50" cho "meet_limit".

Kết quả xuất ra: `True`

Biểu thức đúng vì chiều cao (height) được cho là 58, và 58 > 50. Có nghĩa là **meet_limit** là **đúng**.

> Biến số chứa câu trả lời của phép tính không chứa biểu thức toán học. Biến Boolean tương đồng vì chúng chứa "True" hoặc "False" làm câu trả lời cho biểu thức so sánh thay vì mệnh đề so sánh.

Ở ví dụ này, ta gán biểu thức Boolean **2 bằng 4** cho biến **test1**, và nó là **sai**.

```
test1 = 2 == 4
print(test1)
```

Kết quả xuất ra: `False`

TOÁN TỬ SO SÁNH đánh giá thông tin là đúng hay sai. Chúng so sánh hai giá trị với nhau.

Toán tử so sánh là:

TOÁN TỬ	Ý NGHĨA
==	Bằng
!=	Không bằng
<	Nhỏ hơn

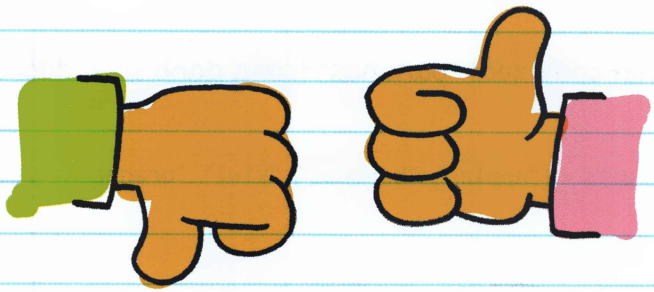
BIỂU TƯỢNG	Ý NGHĨA
>	Lớn hơn
<=	Nhỏ hơn hoặc bằng
>=	Lớn hơn hoặc bằng

Ví dụ về biểu thức được đánh giá là True (đúng):

BIỂU THỨC	Ý NGHĨA	GIÁ TRỊ
2 == 2	2 bằng 2	True
2 != 3	2 không bằng 3	True
2 < 3	2 nhỏ hơn 3	True
4 > 3	4 lớn hơn 3	True
2 <= 2	2 nhỏ hơn hoặc bằng 2	True
5 >= 3	5 lớn hơn hoặc bằng 3	True

Ví dụ về biểu thức được đánh giá là Fasle (sai):

BIỂU THỨC	Ý NGHĨA	GIÁ TRỊ
2 == 5	2 bằng 5	False
2 != 2	2 không bằng 2	False
3 < 3	3 nhỏ hơn 3	False
2 > 3	2 lớn hơn 3	False
5 <= 2	5 nhỏ hơn hoặc bằng 3	False
2 >= 3	2 lớn hơn hoặc bằng 3	False



KIỂM TRA KIẾN THỨC

1. _____ dùng để chứa nhiều giá trị.

2. Bạn dùng hàm nào để thêm phần tử vào cuối danh sách?

3. Làm thế nào để thay thế phần tử thứ hai trong danh sách "cars" với "Porsche"?

4. Bạn dùng hàm nào để thêm phần tử vào giữa hai phần tử đã có trong danh sách?

5. Làm thế nào để chứa một danh sách trong danh sách?

6. Giải thích nhiệm vụ của hàm sort( ).

7. Viết code xuất "bananas" trong danh sách dưới đây.

```
fruits = ["apples", "bananas", "kiwi", "oranges", "pears"]
```

8. Đáp án nào sau đây không đánh giá được theo giá trị Boolean?

A. True

B. 3 ** 2

C. False

D. 3 > 2

9. Chương trình sau sẽ xuất kết quả là gì?

```
score = 3
game_over = score > 5
print(game_over)
```

10. Code sau sẽ xuất kết quả là gì?

```
print(100 == 25)
```

11. Trong Python, "==" khác "=" như thế nào?

ĐÁP ÁN

ĐÁP ÁN

1. Danh sách

2. append()

3. cars[1] = "Porsche"

4. insert()

5. Thêm một cặp ngoặc vuông thứ 2 bao quanh danh sách bên trong danh sách khác. Danh sách trong sẽ chiếm một chỗ của danh sách ngoài.

6. Hàm sort() sắp xếp lại danh sách theo thứ tự chữ cái hoặc chữ số.

7. print(fruits[1])

8. B 9. False 10. False

11. Trong Python, "==" là toán tử so sánh để kiểm tra hai giá trị có bằng nhau hay không, nhưng "=" là toán tử gán và có nhiệm vụ gán giá trị cho biến.

Chương 29

VÒNG LẶP FOR

Python dùng các dạng **VÒNG LẶP** khác nhau để thực hiện nhiệm vụ khác nhau. VÒNG LẶP *FOR* lặp lại code theo số lần cụ thể.

VÒNG LẶP
Code dùng để lặp lại một chuỗi lệnh.

ĐỊNH DẠNG VÒNG LẶP *FOR*

Vòng lặp *For* đi theo một định dạng rất đặc biệt; thậm chí cả số khoảng cách bạn dùng cho một dòng cũng quan trọng.

Các bước tạo vòng lặp *For*:

BƯỚC 1: Gõ từ khóa "for". Vòng lặp *for* luôn luôn bắt đầu với "for" để thông báo rằng code sau là vòng lặp *For*.

BƯỚC 2: Đặt tên cho **BIẾN ĐẾM**. Giá trị của biến này tăng lên mỗi lần vòng lặp lặp lại. Ví dụ, nếu bạn dùng vòng lặp *For* 10 lần, bắt đầu với 0 và tăng 1 mỗi lần vòng lặp bắt đầu lại, biến đếm sẽ hiển thị 0 trong lần chạy đầu tiên, sau đó là 1 vào lần chạy tiếp theo, sau đó là 2, 3, và hơn nữa, tăng 1 vào mỗi lần vòng lặp bắt đầu lại.

Bạn có thể đặt tên biến đếm tùy ý. Nhiều lập trình viên đặt tên biến đếm của họ là "i" theo cách truyền thống.

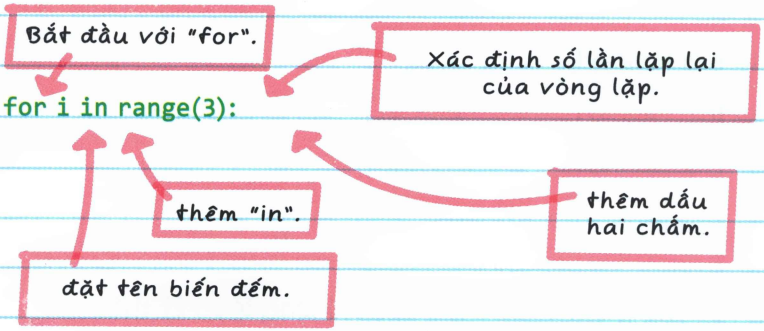
BƯỚC 3: Thêm từ khóa "in" để thông báo rằng bạn sắp chỉ định số lần vòng lặp lặp lại.

BƯỚC 4: Đặt số lần vòng lặp lặp lại. Sử dụng hàm `range()` để làm điều này. Hàm range sẽ đếm từ 0 trở lên, nhưng không đến số được ghi trong ngoặc của hàm.

Range( ) cho biết nơi bộ đếm bắt đầu, dừng lại, và bước đếm. Tham số bao gồm (bắt đầu, dừng lại, bước). Tham số **bắt đầu** và **dừng lại** cho bộ đếm biết số bắt đầu và kết thúc. Tham số **bước** cho bộ đếm biết cần đếm theo bao nhiêu bước. Ví dụ, range(4, 13, 4) sẽ đếm từ 4 đến 12 (số trước số "dừng") theo **số gia của** 4; các số được đếm theo khoảng này là: 4, 8, 12.

BƯỚC 5: Kết thúc dòng đầu tiên của vòng lặp *For* với dấu hai chấm (:).

Dưới đây là ví dụ về vòng lặp *For*, với biến đếm tên *i*, đếm bước 1 từ 0 tới 3.



Dưới dòng đầu tiên của vòng lặp *For*, thêm code được lặp.

Thụt lề code được lặp lại bằng cách nhấn phím **TAB** trên bàn phím một lần hoặc dùng phím cách để thêm 4 khoảng cách. Trong Python, thụt lề cho biết rằng code ở dòng đó là một phần của dòng trên. Bạn có thể thêm

hàm print vào vòng lặp bằng cách bắt đầu một dòng mới và thụt lề nó theo cách tương tự.

VÍ DỤ. bạn có thể thêm hàm print hiển thị "Hip hip hooray!"

```
for i in range(3):
    print("Hip hip hooray!")
```

Điền code bạn muốn lặp lại

Thụt vào 1 tab (4 khoảng cách)

Kết quả là:

```
Hip hip hooray!
Hip hip hooray!
Hip hip hooray!
```

Bạn có thể thêm code chạy sau khi lòng lặp kết thúc.

Để làm vậy, bắt đầu một dòng mới không phải thụt lề, như sau:

```
for i in range(3):
    print("Hip hip hooray!")
print("Congratulations")
```

Trong vòng lặp

Ngoài vòng lặp

Kết quả là:

```
Hip hip hooray!
Hip hip hooray!
Hip hip hooray!
Congratulations
```

Mỗi lần bạn tạo một vòng lặp *For*, bạn cũng tạo một biến mới đếm số lần vòng lặp được lặp lại. Nghĩa là biến đó có thể dùng trong code được lặp.

Ở ví dụ trước, biến "i" dùng trong vòng lặp *For*. Xuất giá trị "i" trước "Hip hip hooray!" cho biết giá trị tăng mỗi lần vòng lặp chạy:

```
for i in range(3):
    print(i, "Hip hip hooray!")
```

NHẮC NHỞ: Hàm range( ) bắt đầu đếm ở 0 và dừng trước số được đặt trong ngoặc.

Kết quả là:

```
0 Hip hip hooray!
1 Hip hip hooray!
2 Hip hip hooray!
```

Trong kết quả hiển thị, giá trị biến "i" ở vòng lặp đầu là 0, ở vòng lặp cuối "i" có giá trị là 2.

Biến đếm có thể dùng theo cách khác.

VÍ DỤ, bạn có thể tạo chương trình đếm ngược. Để làm vậy, trừ biến đếm từ 10, và máy tính sẽ hiển thị đếm ngược từ 10 tới 1:

```
for i in range(10):
    print(10 - i)
```

Kết quả là:

```
10
9
8
7
6
5
4
3
2
1
```

Hàm range( ) được đặt ở 10, nhưng có thể đặt cao hơn nếu bạn muốn.

Bạn có thể dùng hai tham số khác trong hàm range( ) để đếm những số chia hết cho 2 giữa 2 và 10:

```
for i in range(2, 10, 2):
    print(i)
```

Bắt đầu ở số này.

Đếm theo số bước này.

Kết thúc trước số này.

Kết quả là:

```
2
4
6
8
```

Để xuất mọi phần tử trong danh sách (ví dụ như "fruits) sử dụng code sau:

```
fruits = ['pears', 'oranges', 'mangos', 'cherries', 'bananas', 'apples']

for i in range(6):
    print(fruits[i])
```

Phần tử trong danh sách

Tên danh sách

Kết quả là:

```
pears
oranges
mangos
cherries
bananas
apples
```

Có cách dễ hơn để lặp lại toàn bộ danh sách mà không cần biết chính xác độ dài danh sách.

Thay vì dùng hàm **range( )**, hãy sử dụng danh sách đó, và vòng lặp sẽ chạy cùng số lần với số phần tử trong danh sách. Thay vì là một số, giá trị biến đếm sẽ đặt là mỗi phần tử của danh sách như các bước qua toàn bộ danh sách.

Để xuất mọi hoa quả trong danh sách "fruits", thay hàm range( ) với tên danh sách:

```
fruits = ['pears', 'oranges', 'mangos', 'cherries',
'bananas', 'apples']

for i in fruits:
    print(i)
```

Dùng tên danh sách thay vì range( ).

Giá trị biến của I là phần tử danh sách, không phải là số.

KIỂM TRA KIẾN THỨC

1. Giải thích nhiệm vụ của range( ) trong vòng lặp *For*.

2. Phần này của code vòng lặp *For* cần phải chú ý?

3. Hai cách để xuất các phần tử trong danh sách sau đây là gì?

```
Seasons = ["Winter", "Spring", "Summer", "Fall"]
```

Lật trang để biết câu hỏi số 4

4. Viết kết quả của mỗi chương trình sau đây:

TÊN CHƯƠNG TRÌNH	CHƯƠNG TRÌNH	KẾT QUẢ
A	`for i in range(4):` `  print(i)`	
B	`for i in range(10):` `  print(i * 5)`	
C	`for i in range(25, 101, 25):` `  print(i)`	

D	`colors = ["red", "blue",`
	`"yellow"]`
	`for i in colors:`
	`  print(i)`
	`print("Those are the primary`
	`colors")`	

E	`multiplier = 4`
	`for i in range(11):`
	`  print(i * multiplier)`	

F	`for i in range(7):` `  print(i)`	

ĐÁP ÁN

1. Hàm range( ) định nghĩa nơi vòng lặp bắt đầu, kết thúc và số bước nó nên đếm.

2. Phần code thụt vào 4 khoảng cách (hoặc 1 tab) là phần code chạy mỗi lần vòng lặp *for* lặp lại.

3. Một cách:

```
for i in range(4):
    print(Seasons[i])
```

Cách khác:

```
for i in Seasons:
    print(i)
```

4. Kết quả:

A: 0
1
2
3

B: 0
5
10
15
20
25
30
35
40
45

C: 25
50
75
100

D: red
green
blue
yellow
Those are the primary colors

E: 0
4
8
12
16
20
24
28
32
36
40

F: 0
1
2
3
4
5
6

Chương 30

VÒNG LẶP WHILE VÀ VÒNG LẶP LỒNG GHÉP

VÒNG LẶP *WHILE*

VÒNG LẶP *WHILE* lặp lại miễn là câu lệnh điều kiện trong nó là đúng. Vòng lặp *while* trông tương tự với vòng lặp for, nhưng nó thay phần đếm bằng CÂU LỆNH ĐIỀU KIỆN.

> **LƯU Ý:**
> Câu lệnh điều kiện chỉ chạy code khi một điều kiện nhất định được đáp ứng.

Vòng lặp *while* luôn bắt đầu với từ khóa "while" và theo sau bởi biểu thức Boolean và cuối cùng là dấu hai chấm (:). Phần code lặp lại được viết dưới dòng đầu theo cách có chủ ý.

Bạn có thể tạo một chương trình kiểm tra mật khẩu và sẽ liên tục lặp lại đến khi người dùng gõ đúng mật khẩu. Biểu thức Boolean kiểm tra xem nếu mật khẩu được nhập là đúng (True). Làm như sau:

Tạo biến "password" và gán cho nó giá trị **None**.

"None" là từ khóa Pythone, nghĩa là trống rỗng.

Tạo biến "password" trước khi bạn dùng nó trong vòng lặp *while*.

```
password = None
while password != "myPassword1234":
    password = input("Enter the password: ")
    if password != "myPassword1234":
        print("Your password is incorrect.")
print("Correct password. Welcome."
```

Vòng lặp while sẽ tiếp tục khi mật khẩu nhập vào không phải là *"myPassword123"*.

Điều kiện này sẽ chỉ chạy hàm print( ) ở dưới nếu giá trị mật khẩu không phải là *"myPassword1234"*.

Hàm print( ) sẽ chạy sau khi giá trị biến bằng với "myPassword1234".

Kết quả ví dụ:

```
Enter the password: rememberMe
Your password is incorrect.
Enter the password: CantRemember
Your password is incorrect.
Enter the password: OhNowIDo
Your password is incorrect.
Enter the password: myPassword1234
Correct password. Welcome.
```

Chương trình sẽ liên tục lặp lại nhắc người dùng nhập mật khẩu đến khi người dùng nhập "myPassword1234". Khi điều đó xảy ra, vòng lặp kết thúc và phần còn lại của chương trình sẽ chạy (hiển thị "Correct password. Welcome").

Vòng lặp vô tận

Trong Python, vòng lặp **VÔ TẬN** (vòng lặp forever) là vòng lặp *while* sử dụng biểu thức Boolean không bao giờ sai.

Những ví dụ sau là vòng lặp vô tận vì các biểu thức Boolean không thể sai:

```
while True:
    print("This is the song that never ends.")

while 4 > 3:
    print("This is the song that never ends.")
```

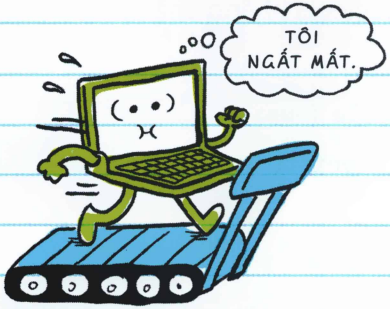
```
while "hello" == "hello":
    print("This is the song that never ends.")

while 5 <= 5:
    print("This is the song that never ends.")
```

Không vòng lặp nào trên đây có câu lệnh điều kiện có thể sai, vì vậy những vòng lặp này sẽ tiếp diễn mãi mãi. Câu lệnh print( ) ở mỗi ví dụ trên sẽ lặp đi lặp lại, mãi mãi.

> Bạn có thể thoát vòng lặp vô tận bằng cách ấn Crtl + C đối với máy Windows, và command + C đối với máy Mac.

Đôi khi bạn cố ý tạo ra một phần chương trình là vòng lặp vô tận. Ví dụ, trò chơi điện tử thường dùng vòng lặp vô tận để hoạt hóa nhân vật, liên tục thay đổi chuyển động của nhân vật khi người dùng chơi trò chơi. Nhưng cũng có lúc bạn vô tình viết vòng lặp vô tận, và nó có thể khiến máy tính của bạn sập vì chương trình quá nặng hoặc khiến máy tính xử lý quá nhiều thông tin.



VÒNG LẶP LỒNG GHÉP

VÒNG LẶP LỒNG GHÉP là vòng lặp đặt trong vòng lặp khác. Chúng giúp tạo ra những code lặp phức tạp hơn. Ví dụ, bạn có thể dùng chương trình mật khẩu làm vòng lặp *while* ở ngoài và lồng vòng lặp *for* vào trong để xuất ra tất cả những đáp án người dùng nhập sai trước khi họ nhập đúng.

```
password = None
attempts = []
while password != "myPassword1234":
    password = input("Enter the password:")
    attempts.append(password)
    if password != "myPassword1234":
        print("Your password is incorrect. You have
        already guessed:")
        for i in attempts:
            print(i)
print("Correct password. Welcome.")
```

Tạo biến mới dạng danh sách và gán cho nó giá trị rỗng.

Thêm thông tin đầu vào của người dùng vào danh sách attempts (thử).

Vòng lặp For xuất ra tất cả những phần tử trong danh sách attempts (những mật khẩu mà người dùng đã thử).

Có phần thụt lề thêm cho hàm print( ) này để thể hiện rằng code này nằm trong vòng lặp *for* được lồng ghép, không phải là vòng lặp *while* ở ngoài.

KIỂM TRA KIẾN THỨC

1. Sự khác biệt giữa vòng lặp *while* và vòng lặp *for*?

2. Vòng lặp *while* dùng câu lệnh điều kiện luôn đúng gọi là _____.

3. Bạn dùng phím gì để thoát khỏi vòng lặp vô tận?

4. Mô tả cách phím tab (hoặc 4 khoảng cách) được dùng trong vòng lặp lồng ghép.

5. Viết một vòng lặp cho những kết quả sau đây.

TÊN CHƯƠNG TRÌNH	CHƯƠNG TRÌNH	KẾT QUẢ
A		Chương trình xuất ra: Go! Go! Go! Miễn là biến "x" lớn hơn 50
B		Chương trình xuất ra: Good morning, Steve Miễn là biến "name" bằng với "Steve"

TÊN CHƯƠNG TRÌNH	CHƯƠNG TRÌNH	KẾT QUẢ
C		Chương trình đếm 3 lần từ 7 tới 11 theo bước 2
D		Chương trình đếm tới 5 mãi mãi
E		Chương trình xuất ra: hip hip hooray hip hip hooray hip hip hooray
F		Chương trình xuất ra: 1 2 3 4 5
G		Chương trình xuất ra: Hello, friend Mãi mãi, hoặc đến khi chương trình bị dừng lại bởi tổ hợp phím Ctrl + C

ĐÁP ÁN

1. Vòng lặp *for* sẽ chạy theo số lần đã đặt, vòng lặp *while* sẽ chạy miễn điều kiện là đúng (True).

2. Vòng lặp vô tận

3. Crt + C đối với máy Windows hoặc command + C đối với máy Mac.

4. Tab (hoặc 4 khoảng cách) cho biết vòng lặp nào trong vòng lặp nào. Tab thêm hoặc 4 khoảng cách nữa cho biết code được lồng trong vòng lặp trong.

5. Đáp án

A.
```
while x > 50:
    print("Go! Go! Go!")
```

B.
```
while name == "Steve":
    print("Good morning, Steve")
```

C.
```
for i in range(3):
    for j in range(7, 12, 2):
        print(j)
```

Tên biến có thể khác.

D.
```
count = True
while count:
    for i in range(1, 6, 1):
        print(i)
```

Tên biến có thể khác.

E.
```
for i in range(3):
    for j in range(2):
        print("hip", end=" ")
    print("hooray")
```

Tên biến có thể khác.

F.
```
num = 0
while num < 5:
    num = num + 1
    print(num)
```

Tên biến có thể khác.

G.
```
while True:
    print("hello, friend")
```

999, 1000 VÒNG!
BẠN CỨ NÓI VÀ
TÔI CỨ CHẠY!

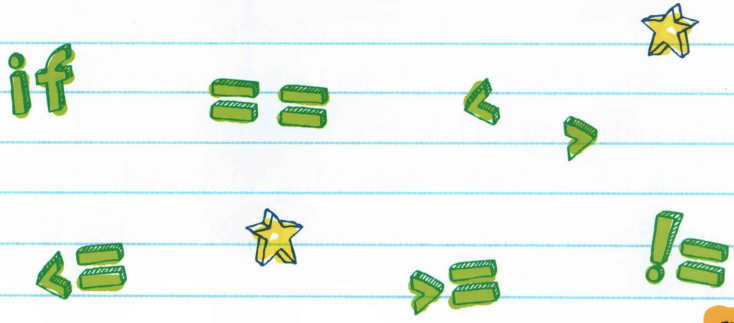
Chương 31

CÂU LỆNH ĐIỀU KIỆN

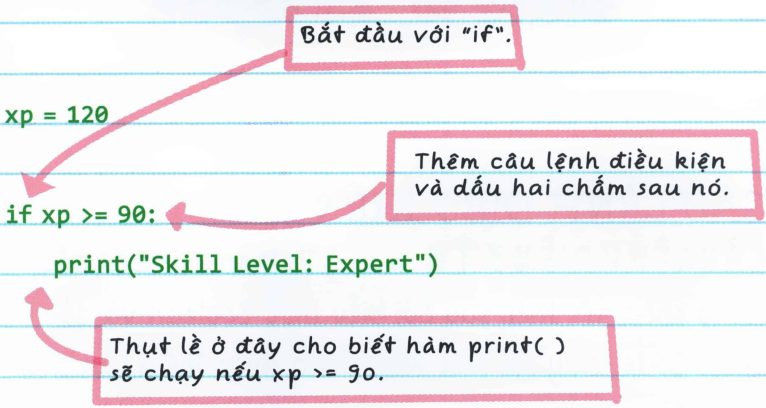
CÂU LỆNH ĐIỀU KIỆN

CÂU LỆNH ĐIỀU KIỆN trong Python luôn bắt đầu với từ khóa "if", theo sau bởi biểu thức Boolean và dấu hai chấm. Code chạy khi điều kiện đúng được thêm bên dưới dòng đầu và chú ý thụt vào 1 tab (4 khoảng cách).

> Biểu thức Boolean sử dụng ==, !=, <, >, <=, và >=.



VÍ DỤ, bạn có thể viết một chương trình hiển thị thông điệp nếu người chơi đạt mức độ expert (chuyên nghiệp). Code xuất kết quả "Skill Level: Expert" sẽ chỉ chạy nếu như điểm kinh nghiệm (xp - experience points) của người chơi lớn hơn hoặc bằng 90. Nếu xp nhỏ hơn 90, không có gì xảy ra.



`Skill Level: Expert` sẽ được hiển thị vì 120 > 90.

Câu lệnh Else

Câu lệnh **ELSE** là câu lệnh chạy khi giá trị Boolean là sai (False). Để dùng câu lệnh *else*, gõ "else" ở dòng mới. Sau đó, ở dòng tiếp theo, thụt vào 1 tab (4 khoảng cách) và thêm code chạy khi biểu thức Boolean là sai. Ví dụ, bạn có thể nói: nếu xp lớn hơn hoặc bằng 90, hãy hiển thị "Skill Level: Expert", nhưng nếu xp của người chơi dưới 90, hiển thị "Skill Level: Novice".

```
xp = 50
if xp >= 90:
    print("Skill Level: Expert")
else:
    print("Skill Level: Novice")
```

Cho biết nên chạy code nào khi biểu thức Boolean sai.

Ví dụ này sẽ hiển thị "Skill Level: Novice" vì xp là 50, nó không lớn hơn hoặc bằng 90.

Elif

ELIF dùng để kết hợp câu lệnh *else* với câu lệnh điều kiện khác để kiểm tra thông tin thêm. *Elif* chỉ dùng sau câu lệnh "if" hoặc câu lệnh "elif" khác. Bên cạnh *elif*, thêm biểu thức Boolean và dấu hai chấm.

Ví dụ, bạn có thể thêm bậc kinh nghiệm nữa vào trò chơi bằng cách thêm câu lệnh *elif* vào code.

Điều kiện đầu tiên hiển thị: "Skill Level: Expert[1]" nếu xp lớn hơn hoặc bằng 90.

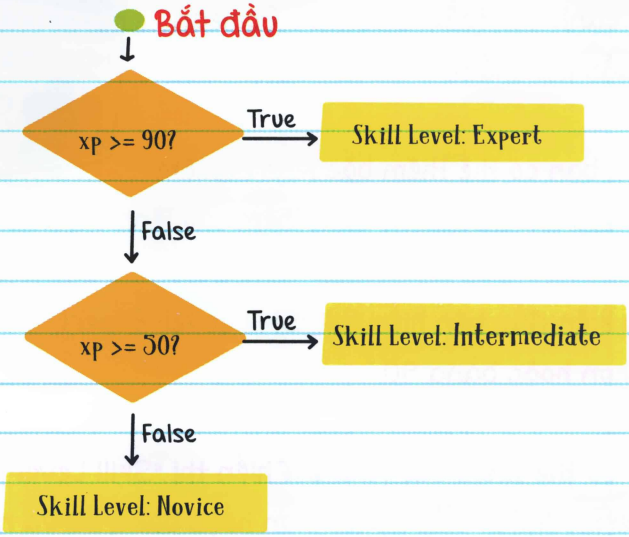
Bạn có thể thêm câu lệnh *elif* hiển thị "Skill Level: Intermediate[2]" nếu xp lớn hơn hoặc bằng 50 và nhỏ hơn

[1] Expert: bậc thầy.
[2] Intermediate: kinh nghiệm.

90. Nếu xp nhỏ hơn 50, câu lệnh else sẽ chạy code hiển thị "Skill Level: Novice[1]".

```
xp = 50
if xp >= 90:
    print("Skill Level: Expert")
elif xp >= 50:
    print("Skill Level:
    Intermediate")
else:
    print("Skill Level: Novice")
```

Dưới đây là sơ đồ của chương trình, thể hiện các nhánh khác nhau cho mỗi điều kiện.



[1] Novice: lính mới.

Câu lệnh điều kiện phức hợp

CÂU LỆNH ĐIỀU KIỆN PHỨC HỢP là câu lệnh điều kiện chứa hai biểu thức Boolean. Ví dụ, bạn có thể tạo trò chơi tặng điểm thưởng cho người chơi nếu điểm của họ lớn hơn 50 và họ chơi ở mức "khó" (hard).

TOÁN TỬ LOGIC kết hợp nhiều biểu thức Boolean hoặc giá trị được quy về giá trị Boolean.

Python dùng 3 toán tử logic:

AND: Nếu cả hai biểu thức là True, vậy cả điều kiện là True. Nếu một trọng cả hai biểu thức là False, vậy cả điều kiện là False.

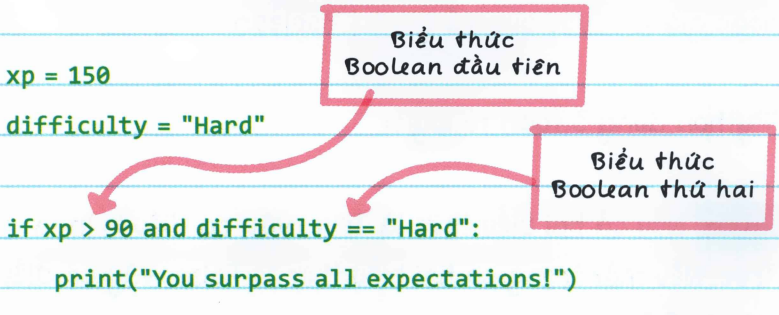
OR: Nếu một hoặc cả hai biểu thức là True, vậy điều kiện là True. Không quan trọng điều kiện kia là True hay False.

NOT: Đảo ngược giá trị biểu thức (True sang False và False sang True).

> Khi chúng ta nói hoặc viết về **AND**, **OR**, hoặc **NOT**, chúng ta viết hoa tất cả. Tuy nhiên trong chương trình Python, toàn bộ các từ được viết thường.

VÍ DỤ, chúng ta có thể hiển thị thông báo đặc biệt nếu người chơi thắng ở mức khó nhất và xp lớn hơn 90.

Câu lệnh điều kiện phức hợp bắt đầu với "if", và sau đó chúng ta có thể thêm biểu thức Boolean theo sau bởi toán tử logic "AND" và biểu thức Boolean khác:




```
xp = 150
difficulty = "Hard"

if xp > 90 and difficulty == "Hard":
    print("You surpass all expectations!")
```

BẠN ĐÃ VƯỢT KỲ VỌNG!

Ví dụ về cách hoạt động của toán tử:

```
3 < 4 and 6 == 6
```

True AND **True** được đánh giá là **True**

```
4 != 4 and 6 > 2
```

False AND **True** được đánh giá là **False**

```
5 == 5 or 6 < 3
```

True OR **False** được đánh giá là **True**

```
4 > 12 or 7 != 7
```

False OR **False** được đánh giá là **False**

```
not(6 < 13)
```

True được đánh giá là **False**

TRUTH TABLES liệt kê toàn bộ kết quả khả dĩ của toán tử logic. Dưới đây là Truth table cho AND, OR và NOT:

BẢNG CHÂN TRỊ AND

Boolean 1	AND	Boolean 2	Kết quả
True	AND	True	True
True	AND	False	False
False	AND	True	False
False	AND	False	False

BẢNG CHÂN TRỊ OR

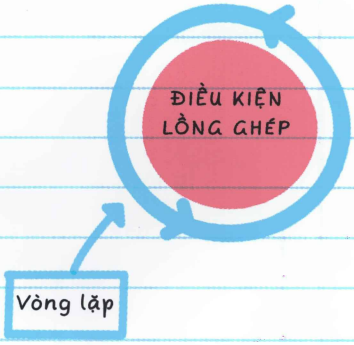
Boolean 1	OR	Boolean 2	Kết quả
True	OR	True	True
True	OR	False	True
False	OR	True	True
False	OR	False	False

BẢNG CHÂN TRỊ NOT

NOT	Boolean	Kết quả
NOT	True	False
NOT	False	True

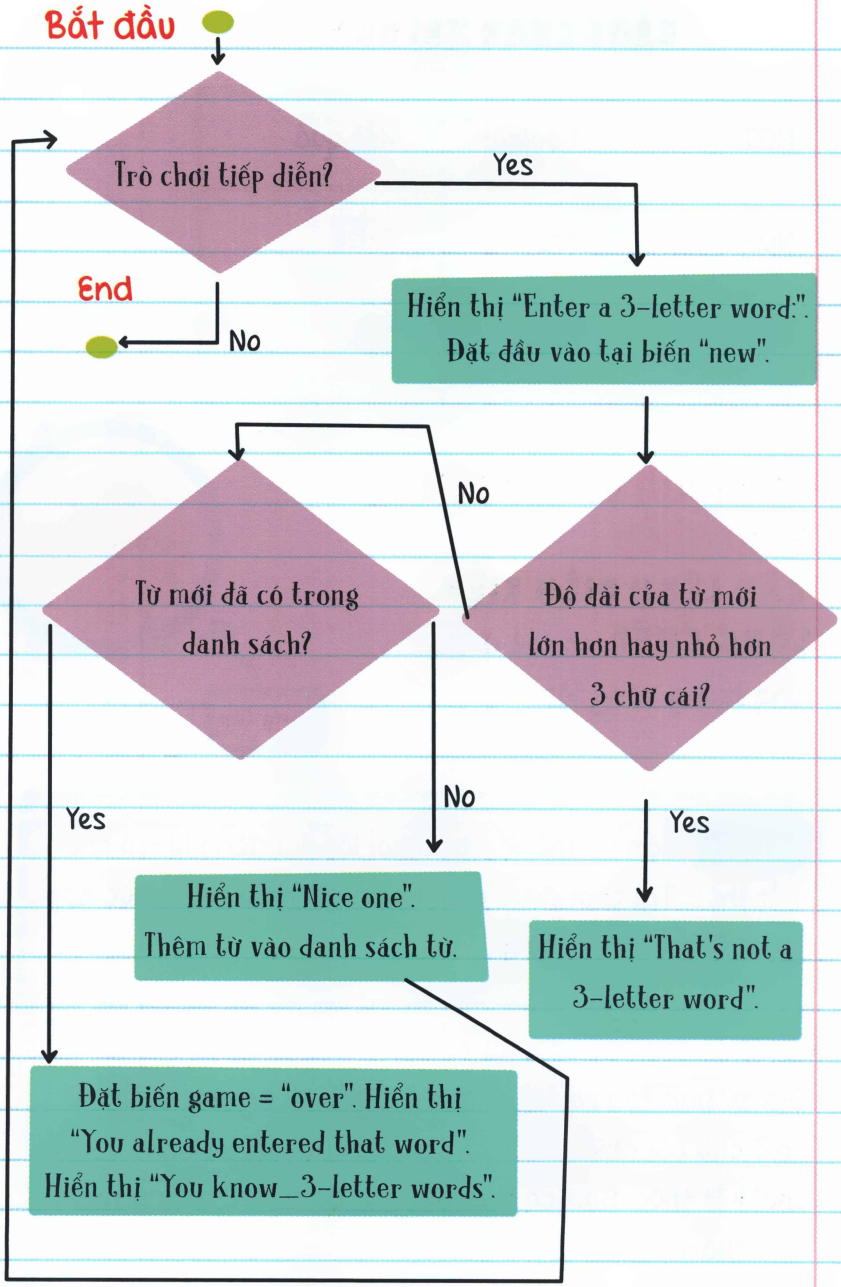
Câu lệnh điều kiện lồng ghép

CÂU LỆNH ĐIỀU KIỆN LỒNG GHÉP có thể được lồng trong vòng lặp.



VÍ DỤ, bạn có thể tạo trò chơi lặp lại đến khi trò chơi kết thúc. Trò chơi được điều khiển bởi điều kiện phức hợp và điều kiện lồng ghép.

Giả sử bạn tạo ra trò chơi mà người dùng phải điền từ có 3 chữ cái nhiều nhất có thể. Nếu họ lặp lại một từ, trò chơi kết thúc. Bạn có thể dùng sơ đồ để vẽ ra thứ tự và các nhánh.


Bắt đầu
Trò chơi tiếp diễn?
Yes
End
No
Hiển thị "Enter a 3-letter word:".
Đặt đầu vào tại biến "new".
No
Từ mới đã có trong danh sách?
Độ dài của từ mới lớn hơn hay nhỏ hơn 3 chữ cái?
Yes
No
Yes
Hiển thị "Nice one".
Thêm từ vào danh sách từ.
Hiển thị "That's not a 3-letter word".
Đặt biến game = "over". Hiển thị "You already entered that word".
Hiển thị "You know_3-letter words".

Sử dụng sơ đồ làm hướng dẫn, bạn tạo chương trình như sau:

```
words = []

game = "play"

while game == "play":

    new = input("Enter a 3-letter word: ")

    if len(new) > 3 or len(new) < 3:

        print("That's not a 3-letter word.")

    else:

        if new in words:

            game = "over"

            print("You already said that word. Game over.")

        print("You know", len(words), "3-letter words.")
```

Tạo danh sách trống gọi là "words".

Bắt đầu vòng lặp while cho trò chơi.

Đặt biến "new" cho câu trả lời của người chơi.

Kiểm tra xem từ có 3 chữ cái.

Kết quả xuất ra nếu từ không có 3 chữ cái.

Nếu từ có 3 chữ cái, chuyển sang code dưới đây.

Nếu từ đã nằm trong danh sách, vòng lặp bị phá vỡ.

Nếu người dùng điền từ đã có trong danh sách, xuất thông báo kết thúc trò chơi.

Cho người chơi biết có bao nhiêu từ trong danh sách.

```
else:
    print("Nice one.")

    words.append(new)
```

Nếu từ đó đặc biệt, xuất kết quả này.

Cũng thêm từ này vào danh sách.

Điều kiện lồng ghép là:

Nếu từ có 3 chữ cái AND đặc biệt, trò chơi tiếp tục.

KIỂM TRA KIẾN THỨC

1. Những code dưới đây thuộc dạng câu lệnh điều kiện nào (if, if/else, if/elif/else):

A.

```
if at_dance == True:
    print("Dance the night away.")
elif busy != True:
    print("Go to the dance.")
else:
    print("Stay home.")
```

B.

```
if hands != "sweaty":
    print("hold")
else:
    print("wipe hands")
```

C.

```
if courage > 50:
    print("Say hello to your crush.")
elif courage < 10:
    print("Don't make eye contact.")
else:
    print("Smile at your crush.")
```

D.

```
if breath != "sweet" and crush_distance < 10:
    print("Go brush your teeth, now!")
```

E.

```
if crush == "at movies" or friends ==
"at movies":
    print("Go to the movies.")
else:
    print("Stay home and chill.")
```

2. Viết kết quả đầu ra của các chương trình dưới đây.

A.

```
num = 3
if num >= 3:
    print("Greater than 3")
```

B.

```
shape = "square"
if shape == "circle":
    print("you win")
else:
    print("no such luck")
```

C.

```
color1 = "red"
color2 = "blue"
if color1 == "red" and
color2 == "yellow":
    print("orange")
elif color1 == "red" and
color2 == "blue":
     print("purple")
else:
    print("green")
```

D.

```
fact = not(5 == 4)
print(fact)
```

3. Toán tử logic AND có nhiệm vụ gì?

4. Đánh giá mỗi câu lệnh điều kiện phức hợp sau là True hay False.

A. Bạn sống ở Trái Đất AND mặt trăng:

B. Bạn có thể hít thở không khí OR hít thở nước

C. NOT (bạn là người)

D. Người nhện là thật OR người dơi là thật

5. Vẽ sơ đồ cho mỗi mô tả sau đây.

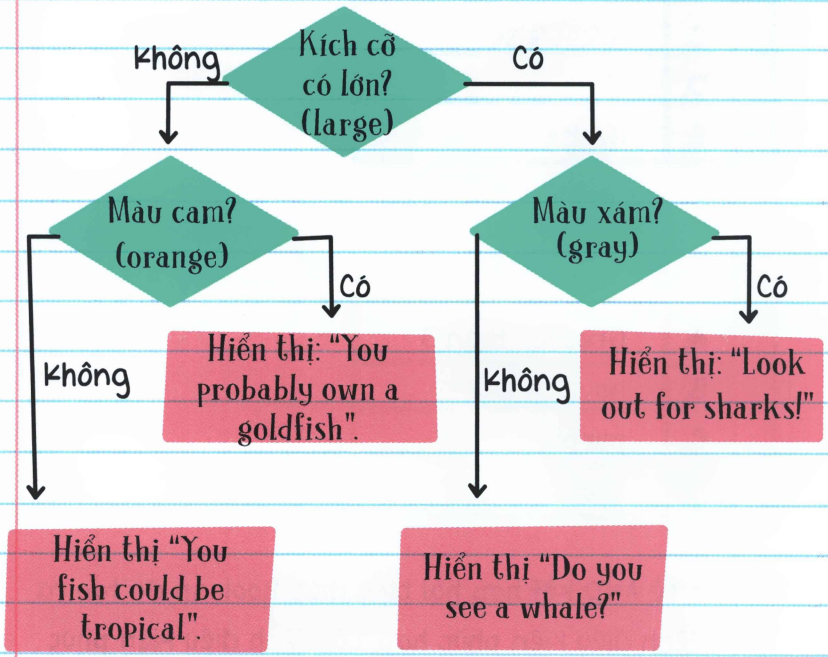
A. Đầu tiên chương trình sẽ hỏi người dùng có muốn xem phim Harry Potter. Nếu họ muốn, gợi ý phim Harry Potter họ nên xem dựa trên việc người dùng thích xem Harry Potter hồi nhỏ (phần 1 hoặc 2) hay Harry Potter lúc trưởng thành (phần 3 tới 7).

B. Đầu tiên chương trình sẽ hỏi người chơi có thích mũ. Nếu họ không, bảo người dùng thử kính che mặt. Nếu họ thích mũ, và nếu họ thích những thứ đơn giản, gợi ý mũ bóng chày, không thì gợi ý mũ chóp.

C. Đầu tiên chương trình sẽ hỏi người dùng có thích rảnh rỗi không. Nếu người dùng thích, gợi ý họ đừng làm gì. Nếu người dùng không thích rảnh

rồi, hỏi họ có thích đọc không. Nếu họ thích đọc, hỏi xem họ muốn đọc nhiều không. Nếu họ muốn đọc nhiều, gợi ý rằng họ nên đọc tiểu thuyết, nếu không bảo họ đọc tạp chí.

6. Viết một chương trình dựa trên sơ đồ sau.



7. Chương trình ở câu 6 sẽ xuất kết quả gì nếu biến "color" có giá trị "orange" và biến "size" có giá trị "small"?

ĐÁP ÁN

1.

A.	if/elif/else
B.	if/else
C.	if/elif/else
D.	if
E.	if/else

2.

A.	Greater than 3
B.	no such luck
C.	purple
D.	True

3. Toán tử AND kết hợp hai biểu thức Boolean để tạo ra câu lệnh điều kiện phức hợp. Câu lệnh điều kiện phức hợp sử dụng AND và chỉ đúng khi cả hai biểu thức Boolean trong câu lệnh là đúng.

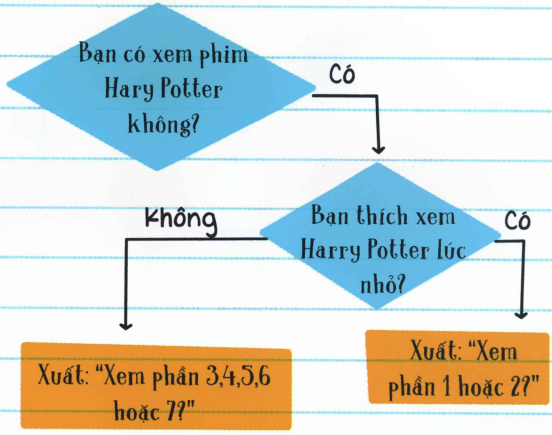
4. **A.** False

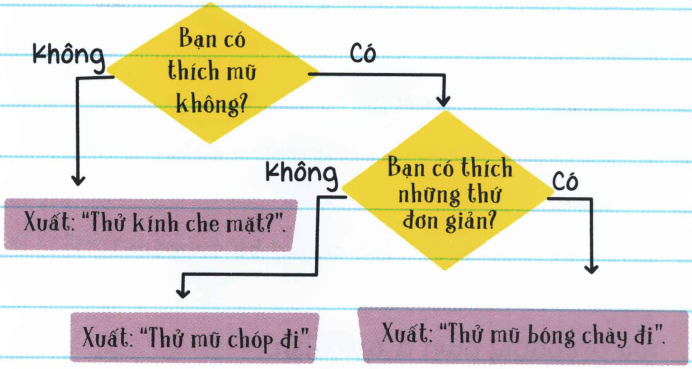
B. True

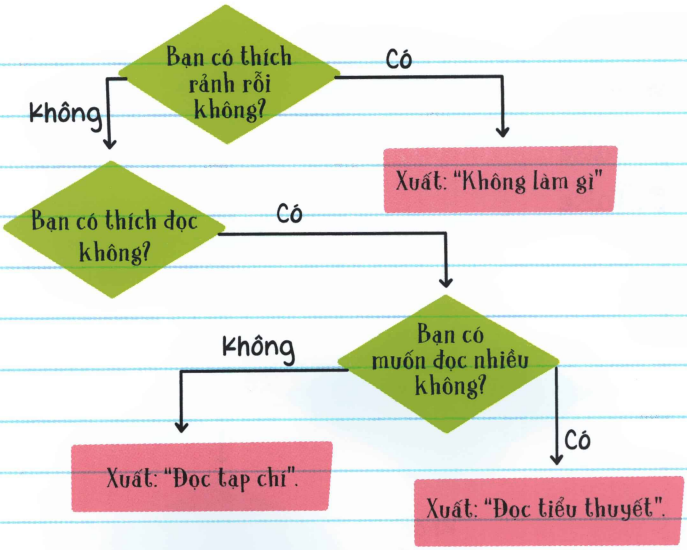
C. False

D. False

5. **A.**







6.

```
if size == "large":
    if color == "gray":
        print("Look out for sharks!")
    else:
        print("Do you see a whale?")
else:
    if color == "orange":
        print("You probably own a goldfish.")
    else:
        print("Your fish could be tropical.")
```

7. You probably own a goldfish.

Chương 32

HÀM

Hàm là những khối code bạn có thể dùng để xây chương trình lớn hơn. Bạn có thể thêm những hàm xây sẵn bằng cách nhập **modules**.

Một kiểu thư viện

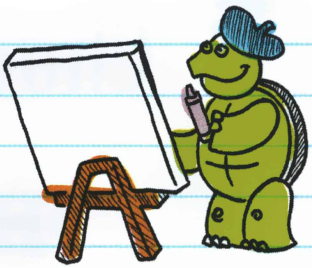
Từ **hàm** và **thủ tục** gần như cùng nghĩa. Khi nói đến Python, chúng ta hay dùng từ hàm (một số người dùng từ thủ tục).

ĐỒ HỌA CHÚ RÙA

MODULE CHÚ RÙA bao gồm những hàm cho phép bạn coi màn hình như bảng vẽ và dùng biểu tượng chú rùa để vẽ lên đó.

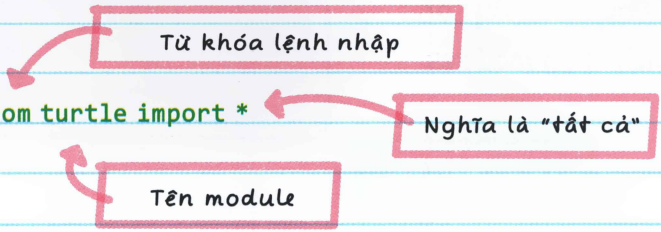
Thư viện là bộ sưu tập những hàm và code được viết sẵn, có thể nhập vào chương trình.

Nhập module



Bạn có thể nhập module bằng cách sử dụng **lệnh nhập**. Bắt đầu với từ "from", sau đó nhập tên module, và sau đó gõ "import". Hành động này có nghĩa là bạn muốn nhập module được gọi tên (trong trường hợp này, "rùa").

Bạn có thể gọi tên hàm cụ thể mà bạn muốn nhập từ module. Để nhập toàn bộ hàm, kết thúc lệnh nhập với "*".



Điều này có nghĩa là nhập tất cả các hàm từ module chú rùa.

Di chuyển chú rùa

Sau khi nhập module chú rùa, bạn có thể dùng hàm của nó để di chuyển chú rùa quanh màn hình. Ban đầu chú rùa hướng sang phải. Bạn có thể khiến chú rùa di chuyển bằng hàm tiến lên (forward) và lùi lại (backward).

Chế độ mặc định "chú rùa" thực ra là hình tam giác chỉ theo hướng nó đối mặt.

VÍ DỤ, bạn có thể khiến chú rùa di chuyển 100 **PIXEL** quanh màn hình với lệnh:

```
forward(100)
```

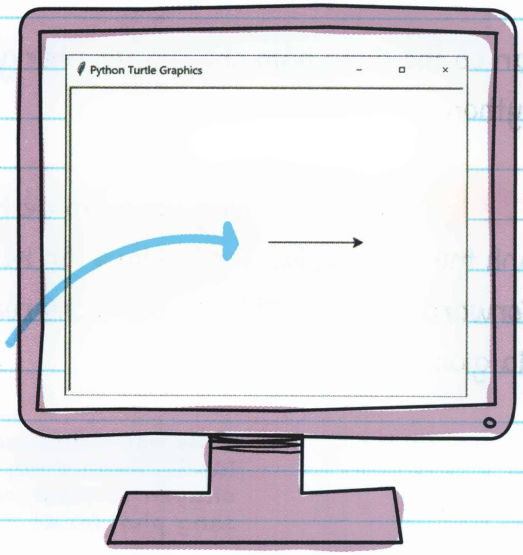
Tham số/Khoảng cách

Hàm

Code trông như sau:

```
from turtle import *
forward(100)
```

Lệnh sẽ mở một cửa sổ mới và vẽ một đường từ trái sang phải dài 100 pixel (Tam giác ở cuối đường thẳng là chú rùa).

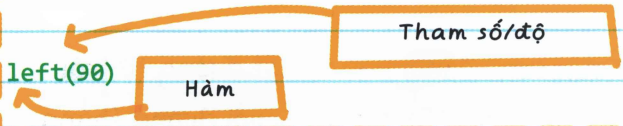


PIXEL

Một **pixel** (từ **phần tử hình ảnh**) là một điểm sáng trên màn hình. Các điểm nhỏ tạo thành hình ảnh máy tính hiển thị.

Hàm **left**() và **right**() quay chú rùa sang trái hoặc phải tùy theo hướng nó đang đứng trên màn hình. Cả hai hàm đều cần tham số cho biết số độ cần quay.

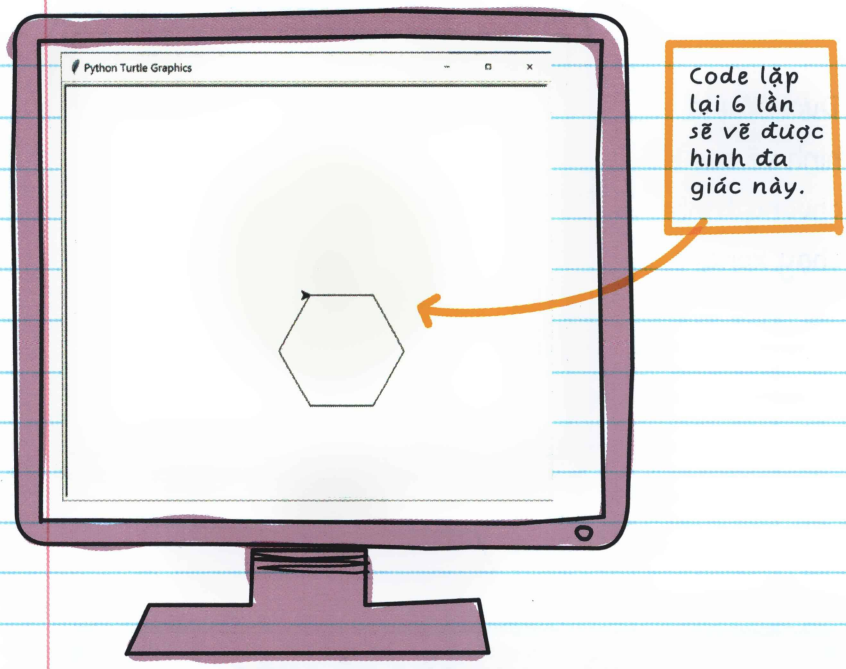
VÍ DỤ. quay 90 độ sang trái, sử dụng:



Bạn có thể dùng hàm chú rùa với những phần khác của Python.

VÍ DỤ. bạn có thể dùng vòng lặp để tạo ra hình ảnh thú vị. Vòng lặp **for** lặp lại 6 lần kết hợp với hàm **forward** và hàm **right** từ module chú rùa sẽ vẽ được hình đa giác.

Code này điều khiển chú rùa lặp lại hành động 6 lần: di chuyển 70 pixel, sau đó quay sang phải 60 độ.

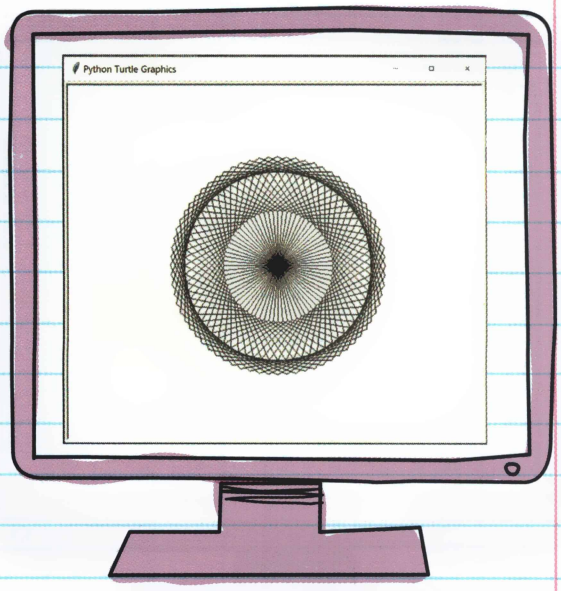


Bạn có thể lồng vòng lặp đa giác vào bên trong một vòng lặp khác và quay 5 độ mỗi lần một đa giác được vẽ.

Bằng cách lặp vòng lặp ngoài 72 lần, bạn có thể vẽ 72 hình đa giác, mỗi hình lệch 5 độ sang bên phải so với hình trước.

```
from turtle import *
for i in range(72)
        for j in range(6):
                forward(70)
                right(60)
        right(5)
```

Dưới đây là hình vẽ sau khi chương trình chạy xong:



CÁC HÀM CHÚ RÙA BỔ SUNG

Trong module chú rùa có nhiều hàm có thể dùng để tạo hình hoạt vui nhộn.

HÀM	MÔ TẢ
forward()	Di chuyển chú rùa về phía trước, tham số là số pixel cần di chuyển.
backward()	Di chuyển chú rùa về phía sau, tham số là số pixel cần di chuyển.
right()	Quay chú rùa theo chiều kim đồng hồ, tham số là số độ cần quay.
left()	Quay chú rùa theo ngược chiều kim đồng hồ, tham số là số độ cần quay.
penup()	Nhấc bút (ngừng vẽ theo chuyển động của chú rùa), không có tham số.

HÀM	MÔ TẢ
pendown( )	Đặt bút (bắt đầu vẽ theo chuyển động của chú rùa), không có tham số.
pencolor( )	Thay đổi màu sắc của bút, tham số là tên màu sắc.
tiêu đề( )	Cho biết hướng hiện tại - hữu dụng khi bạn cần biết chú rùa đang hướng về đâu trên màn hình, không có tham số.
position( )	Cho biết vị trí x và y - hữu dụng khi bạn muốn biết chú rùa đang ở đâu trên màn hình, không có tham số.
goto( )	Di chuyển chú rùa tới một vị trí cụ thể, tham số là tọa độ x và y.
fillcolor( )	Thay đổi màu chú rùa dùng để tô hình đa giác; tham số giống với tham số ở hàm pencolor( ).
begin_fill( )	Nhớ điểm bắt đầu tô màu đa giác - dùng trước khi hình được vẽ, không có tham số.
end_fill( )	Kết thúc đa giác và tô màu hiện tại - sử dụng sau khi hình được vẽ, không có tham số.
dot( )	Vẽ một chấm ở vị trí hiện tại, không có tham số.
stamp( )	In hình chú rùa trên màn hình, bất kể chú rùa đứng ở đâu, không có tham số.
shape( )	Thay đổi hình dạng chú rùa, tham số là "arrow," "classic," "turtle," "circle," và "square" (mũi tên, cơ bản, chú rùa, vòng tròn, và hình vuông).

Mặc dù một số hàm không sử dụng tham số, bạn vẫn cần sử dụng bao gồm dấu ngoặc đơn ().

HÀM

Nếu bạn dùng một đoạn code nhiều lần trong Python, tốt nhất nên tự tạo hàm của riêng mình, việc này gọi là **ĐỊNH NGHĨA MỘT HÀM**.

Để định nghĩa một hàm, sử dụng định dạng sau:

Bắt đầu với từ "def" – viết tắt của "define" (định nghĩa).

```
def functionName( ):
    print("hello")
```

Đặt tên hàm tùy ý (sử dụng quy tắc tượng tự như đặt tên biến), thêm dấu ngoặc, kết thúc bằng dấu hai chấm.

Thụt lề vào và thêm code của bạn.

GỌI HÀM là dùng hàm đã được định nghĩa. Khi bạn gọi hàm, Python tìm định nghĩa hàm và chạy code tìm được ở thân hàm.

Bạn chỉ có thể gọi hàm sau khi nó được định nghĩa. Để gọi hàm, sử dụng định nghĩa sau:

TÊN HÀM

```
functionName()
```

DẤU NGOẶC

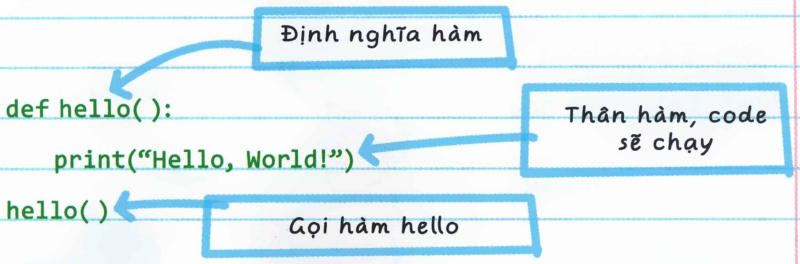
Khi bạn gọi hàm, chương trình sẽ nhảy lại nơi hàm được định nghĩa, chạy tất cả code trong thân hàm, sau đó quay lại nơi chương trình đang chạy khi hàm được gọi.

Thân hàm là phần được viết thụt vào. Thân hàm sẽ chạy khi hàm được gọi.

```
def functionName():
    ••• •• •••
    ••• •• •••

••• •• •••
••• •• •••
functionName()
••• •• •••
••• •• •••
```

Dưới đây là ví dụ về hàm hiển thị "Hello, World!":

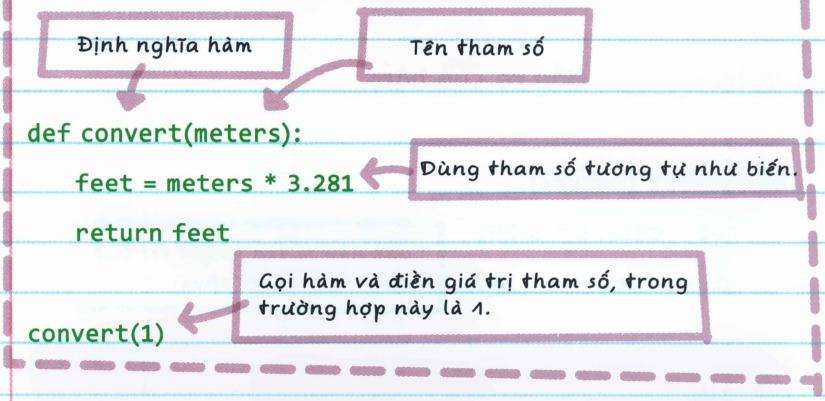


Kết quả: `Hello, World!`

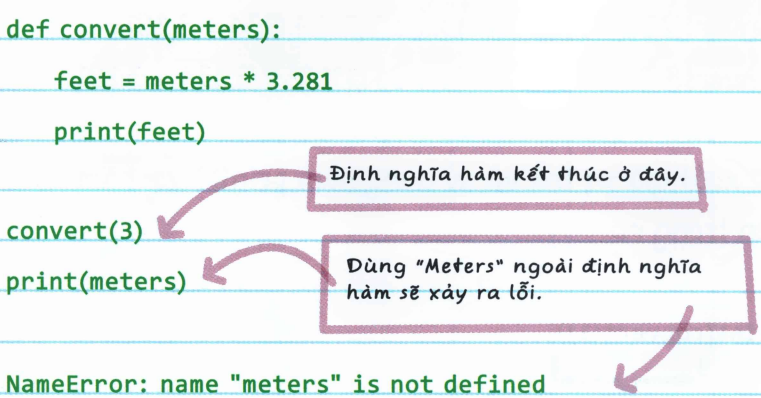
THAM SỐ VÀ GIÁ TRỊ TRẢ VỀ

Tham số và biến tương đồng vì chúng đều dùng để chứa thông tin. Tuy nhiên, **THAM SỐ** khác ở chỗ nó không thể dùng ngoài hàm. Tham số là dữ liệu được cung cấp như thông tin đầu vào từ người dùng. Hàm chỉ có thể dùng tham số trong thân hàm. Vì tham số trong hàm chỉ được nhận diện trong chính hàm đó.

VÍ DỤ, bạn có thể tạo hàm chuyển đổi đơn vị mét sang feet. Bên trong hàm chuyển đổi, bạn nhân tham số mét (meters) với 3.281 (số feet trong 1 mét) để biết bằng bao nhiêu feet trong số mét cụ thể.



Nếu bạn cố dùng tham số mét bên ngoài định nghĩa hàm, bạn sẽ gặp lỗi:



GIÁ TRỊ TRẢ VỀ CỦA HÀM là thông tin bạn có thể chuyển từ hàm ra chương trình chính. Đây là đầu ra của hàm. Để chuyển thông tin ra khỏi hàm, viết từ khóa "return" theo sau bởi dữ liệu đầu ra.

Giá trị trả về = đầu ra của hàm

```
def convert (meters):
    feet = meters * 3.281
    return feet
```

Code này sẽ trả về giá trị của feet mỗi lần hàm chạy.

Với hàm chuyển đổi được định nghĩa, bạn có thể gọi hàm bao nhiêu lần cũng được và mỗi lần nhận giá trị trả về khác nhau.

convert(5)	returns 16.405 (5 * 3.281).
convert(234)	returns 767.754 (234 * 3.281).
convert(5,234)	returns 17,172.754 (5.234 * 3.281).

Để xuất giá trị trả về, sử dụng hàm **print()** với hàm gọi bên trong nó:

```
print(convert(1))
```

Hàm gọi

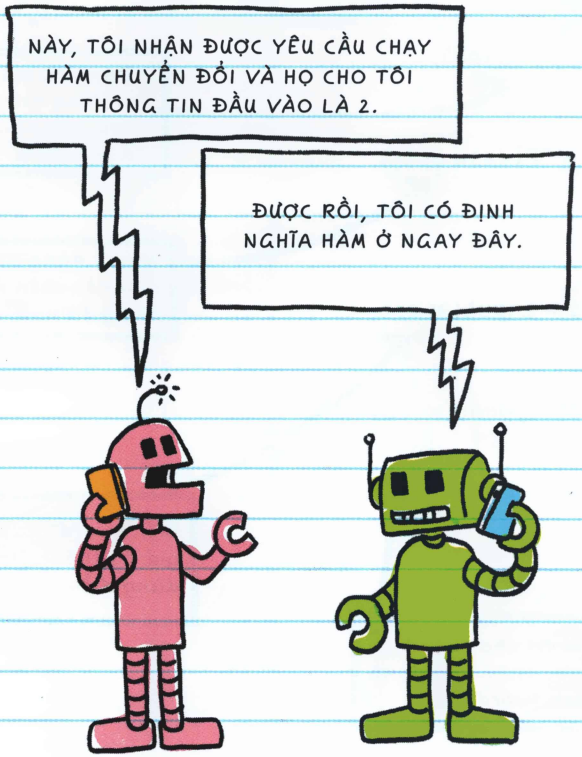
Chương trình sẽ xuất kết quả: `3.281`

Bạn cũng có thể thêm văn bản quanh hàm gọi để giải thích code, ví dụ như sau:

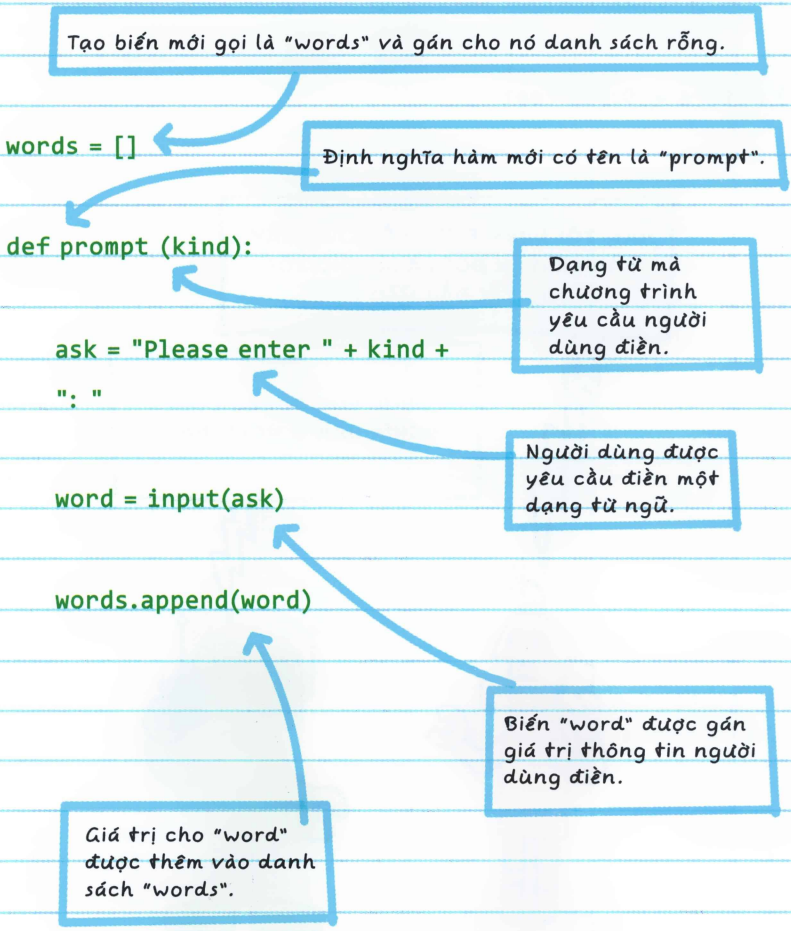
```
print("3 meters = ", convert(3), "feet")
```

Chương trình xuất ra:

```
3 meters =  9.843 feet
```



Bạn có thể lặp lại một đoạn code đồng thời dùng tham số để thay đổi một chút mỗi lần dùng nó. Ví dụ, trong chương trình trò chơi từ ngữ, thay vì liên tục gõ "Please enter a noun" hoặc "Please enter a adjective", bạn có thể tạo một hàm thực hiện hành động này cho bạn:



```
prompt("an adjective")
prompt("a nationality")
prompt("a person")
prompt("a plural noun")
prompt("an adjective")
prompt("a plural noun")
```

Dạng từ ngữ người dùng điền.

Hàm "prompt" có thể gọi nhiều lần, yêu cầu người dùng điền các dạng từ ngữ khác nhau bằng cách đổi tham số cho mỗi hàm gọi.

Danh sách từ ngữ, tạo ra bởi thông tin người dùng điền, được chèn vào trong trò chơi:

```
print("Computers were invented by a", words[0],
words[1], "engineer named", words[2], ". To make a
computer, you need to take a lot of", words[3], ", melt
them down, and make", words[4], words[5], ".")
```

Đầu ra của chương trình này là một câu chuyện hoàn chỉnh sử dụng từ của người dùng để điền vào các chi tiết chính.

KIỂM TRA KIẾN THỨC

1. Bạn nhập tất cả các hàm trong module chú rùa vào chương trình Python như thế nào?

2. Viết hai dòng code để di chuyển chú rùa 45 pixel và quay phải 30 độ (giả sử bạn đã nhập module chú rùa).

3. Gọi hàm là gì? Bạn sẽ viết code gì để gọi hàm có tên là "BopIt", không cần thông tin đầu vào?

4. Khi bạn tạo hàm mới, bạn cần _____ nó.

5. Bạn có thể chuyển thông tin vào hàm bằng cách sử dụng _____.

6. Bạn có thể chuyển thông tin ra khỏi hàm vào chương trình bằng cách sử dụng _____.

7. Giải thích lỗi sai của chương trình bên dưới:

```
def distance (laps):
    meters = laps * 100
    return meters
print(laps)
```

8. Để sử dụng hàm sau khi bạn định nghĩa nó, bạn cần _____. Hành động này được thực hiện với _____.

9. Viết giá trị trả về của mỗi hàm dưới đây:

TÊN	CODE	GIÁ TRỊ TRẢ VỀ

A.	`name = "Max"`
	`def hello_you(person):`
	`  sentence = "Hello " +`
	`person`
	`  return sentence`
	
	`hello_you("Max")`	

B.	`def plotter(x, y):`
	`  instructions = "Plot a`
	`  course through " + str(x) +`
	`  " and " + str(y)`
	`  return instructions`
	
	`plotter(3, 5)`	

TÊN	CODE	GIÁ TRỊ TRẢ VỀ

C.	def absolute_value(num):
	    if num >= 0:
	        return num
	    else:
	        return num * -1
	
	absolute_value(-4)	

D.	def favorite(category, thing):
	    sentence = "My favorite "
	    + category + " is the " +
	    thing
	    return sentence
	
	favorite("snake", "Python")	

ĐÁP ÁN

1. From turtle import *

2. forward(45)
   right(30)

3. Khi bạn gọi hàm, bạn dùng tên hàm để báo chương trình chạy code tương ứng với tên hàm.

   Code có dạng như này: BopIt()

4. Định nghĩa

5. Tham số

6. Trả về

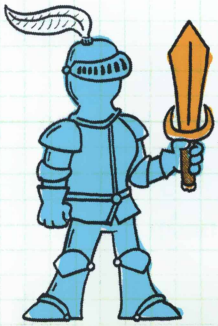
7. Tham số "laps" được dùng ngoài định nghĩa hàm.

8. Gọi, dấu ngoặc đơn

9.

A.	Hello Max
B.	Plot a course through 3 and 5
C.	4
D.	My favorite snake is the Python

BÀI 8



Phát triển Web

Chương 33

INTERNET LÀ GÌ?

MẠNG MÁY TÍNH

MẠNG[1] là một nhóm các máy tính được kết nối với nhau. Ví dụ, máy tính, điện thoại, và các thiết bị khác mà gia đình bạn sở hữu đều là một phần của mạng máy tính nhà bạn. Khi bạn có máy tính bảng mới và nhập mật khẩu wi-fi, bạn đang thêm máy tính bảng đó vào mạng máy tính nhà bạn.

MẠNG
Một nhóm các máy tính được kết nối, chia sẻ thông tin và tài nguyên.

Các thiết bị trong cùng mạng có thể dễ dàng giao tiếp và chia sẻ tài nguyên với nhau. Ví dụ, máy tính thuộc mạng không dây ở quán cà phê đều chia sẻ chung kết nối internet, và điện thoại của gia đình đều có thể điều khiển TV.

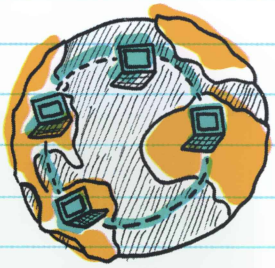
[1] network.

Mạng cục bộ[1]

MẠNG CỤC BỘ (LAN) là loại mạng được tạo nên bởi những máy tính lân cận. Mạng LAN có thể ở bất kỳ đâu từ một vài thiết bị (ví dụ như mạng gia đình) cho đến hàng trăm thiết bị (ví dụ như trường hoặc bệnh viện). Những tổ chức lớn như đại học có thể sở hữu vài mạng LAN - có thể là mỗi tòa một mạng cục bộ.

Mạng diện rộng[2] (WAN) được tạo nên bởi những mạng LAN ở xa nhau kết nối lại (ví dụ như ở những nơi khác nhau của đất nước hoặc thế giới). Mạng WAN kết nối hàng nghìn hoặc thậm chí hàng triệu thiết bị.

Giao tiếp trên mạng LAN



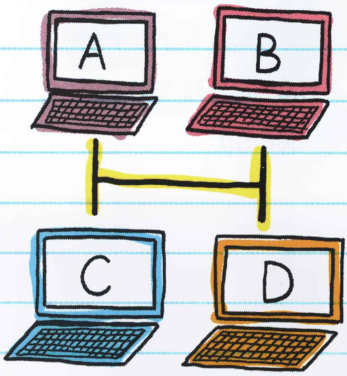
Máy tính trên thế giới giao tiếp với **ETHERNET** (có dây) hoặc **WIRELESS** (không dây).

Ethernet được phát triển vào thập niên 1970. Nó là **GIAO THỨC MẠNG**, hoặc quy tắc, về cách các thiết bị giao tiếp bằng cáp để kết nối vật lý. Wireless được phát triển vào cuối thập niên 1990.

GIAO THỨC MẠNG
Một bộ quy tắc được tiêu chuẩn hóa.

[1] Local Area Network.
[2] Wide Area Network.

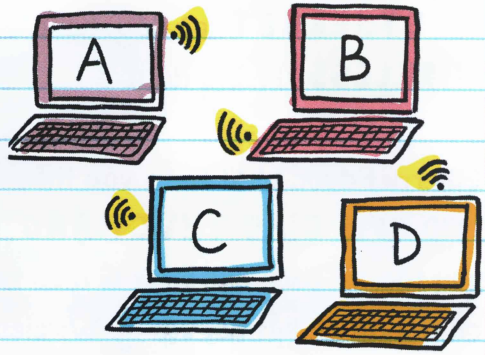
Nó là giao thức mạng sử dụng sóng radio gửi thông tin qua không khí để kết nối các thiết bị.



ETHERNET
Những quy tắc mà máy tính tuân theo để giao tiếp trên mạng có dây.

ETHERNET

WIRELESS
Những quy tắc mà máy tính tuân theo để giao tiếp trên mạng qua sóng radio.



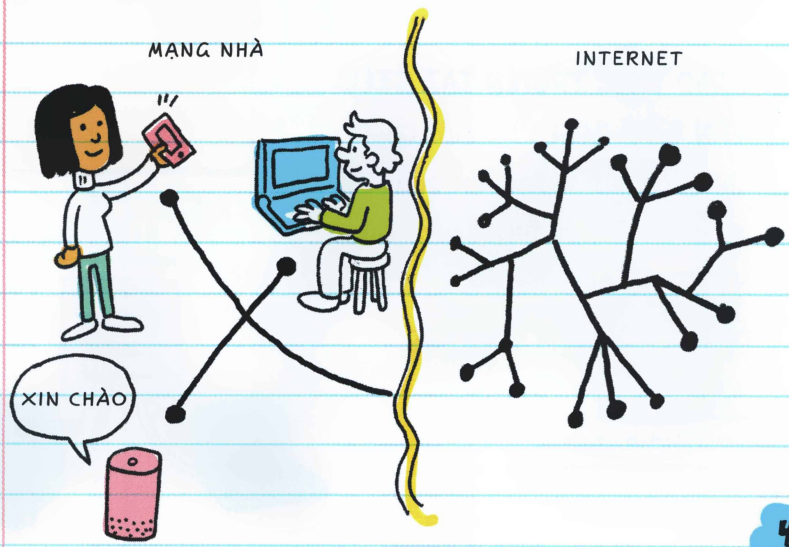
WIRELESS

Wi-Fi là một phương thức phổ biến để kết nối không dây với mạng. "Wi-Fi" được cho là viết tắt của "độ trung thực không dây", được lấy từ thuật ngữ "độ trung thực cao" – được sử dụng để mô tả âm thanh chất lượng cao.

INTERNET

INTERNET là hệ thống toàn cầu bao gồm những mạng máy tính được kết nối. Nó là một mạng khổng lồ bao gồm những mạng nhỏ hơn. Ở nhà, điện thoại, máy tính, và các thiết bị khác của gia đình bạn đều là một phần mạng máy tính nhà bạn, chia sẻ chung kết nối internet.

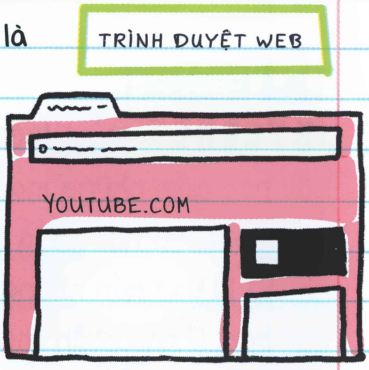
Nhiều nơi có mạng riêng ở chỗ đó. Trường học, cửa hàng cà phê, thậm chí khu trung tâm thương mại gần như luôn có mạng của riêng họ để kết nối với internet.



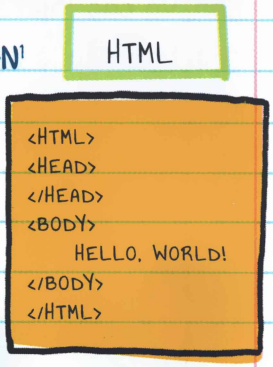
Câu chuyện về nguồn gốc của web

Internet hiện đại bắt đầu vào đầu những năm 1990, khi TIM BERNERS-LEE tạo ra công cụ và khung cho World Wide Web . Trong vài tháng, ông đã tạo ra:

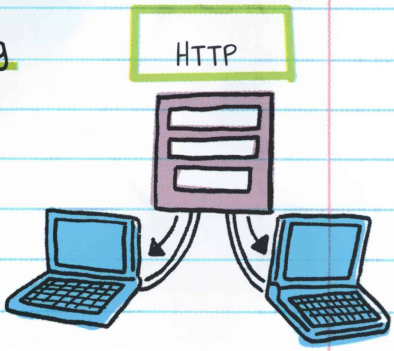
- Trình duyệt web đầu tiên, gọi là WORLDWIDEWEB và sau đó được đặt tên lại là NEXUS. (Bây giờ chúng ta có thể chọn giữa Chrome, Safari, Microsoft Edge, và các trình duyệt khác).



- NGÔN NGỮ ĐÁNH DẤU SIÊU VĂN BẢN[1] (HTML) - ngôn ngữ dùng để tạo cấu trúc của website.



- GIAO THỨC TRUYỀN TẢI SIÊU VĂN BẢN[2] (HTTP) - một cách chính để gửi thông tin qua internet. Trình duyệt web dùng HTTP để giao tiếp với những máy tính chứa các website.



[1] Hypertext Markup Language.
[2] Hypertext Transfer Protocol.

Mạng toàn cầu tạo ra một kỷ nguyên mới cho giao tiếp. Hiện nay có hàng triệu thiết bị và ứng dụng kết nối với nhau trên internet.

GỬI THÔNG TIN TRÊN INTERNET

Khi bạn tải một tài liệu trên internet, ví dụ như nhạc vào điện thoại, các bài hát có dung lượng quá lớn để gửi trong một lần. Thay vào đó, mỗi bài hát được phân nhỏ thành các **PACKETS**, chúng được gửi riêng và sau đó kết hợp lại trong điện thoại của bạn. Mỗi packet có địa chỉ đích đến và số packet để mạng biết nơi gửi nó và cách tập hợp lại tất cả các packet khi chúng đến đích cuối cùng.

PACKETS
Đơn vị dữ liệu nhỏ được gửi đi dưới dạng mã nhị phân qua mạng.

ĐỊA CHỈ IP dùng để hướng packet tới mạng đúng. Địa chỉ IP là những địa chỉ độc nhất dán nhãn các thiết bị trên mạng và internet. Mọi thiết bị đều có địa chỉ IP của riêng nó. Địa chỉ IP có thể đổi và được gán bởi **nhà cung cấp dịch vụ internet.**

IP (INTERNET PROTOCOL)
Quy tắc truyền thông tin trên mạng.

Mỗi máy tính đều có địa chỉ MAC[1] (Điều khiển truy cập phương tiện) riêng được tạo thành từ 16 chữ số và chữ cái, MAC dùng để xác định nhà sản xuất và loại máy cụ thể.

[1] Media Access Control.

NHÀ CUNG CẤP DỊCH VỤ INTERNET

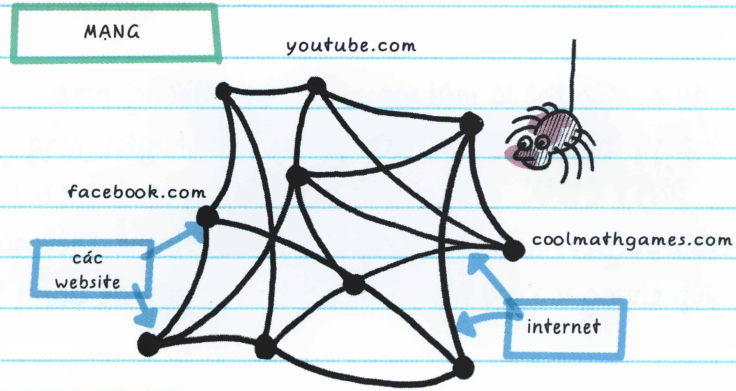
Công ty cung cấp quyền truy cập internet cho mọi người.

SỬ DỤNG INTERNET ĐỂ LƯỚT WEB

INTERNET ≠ MẠNG TOÀN CẦU

INTERNET là kết nối và mạng giữa các thiết bị. Bạn dùng internet để kết nối, lướt và khám phá các website.

MẠNG TOÀN CẦU[1], hay web, là tổ hợp các website được kết nối với nhau qua internet. Web là nội dung bạn có thể nhìn, đọc, nghe, phát, và tải. Hầu hết website có đường liên kết[2] kết nối chúng với website khác. Nếu bạn có thể vẽ hình ảnh của những liên kết này, nó sẽ trông như một mạng nhện tạo nên bởi những website trên khắp toàn cầu. Đó là lý do nó được gọi là Mạng Toàn Cầu.



[1] World Wide Web.
[2] Link.

Khi bạn mở một website trên máy tính, bạn đang mở một bản sao được tải xuống tạm thời của các **FILE HTML** tạo nên trang đó.

HTML là viết tắt của "ngôn ngữ đánh dấu siêu văn bản" và là ngôn ngữ dùng để tạo website. Phần siêu văn bản trong tên nghĩa là ngôn ngữ này được xây dựng để kết nối các website với nhau bằng đường liên kết. Vì website được liên kết (bởi internet), bạn có thể dùng HTML để nhảy từ trang này sang trang khác.

Mỗi website đều được chứa trong một máy tính. Khi bạn mở trình duyệt web và gõ website vào, bạn đang yêu cầu máy tính chứa website gửi bạn một bản sao của site. Máy tính chứa website gọi là **MÁY CHỦ**.

Đầu não có thể là một máy tính bình thường, nhưng thường thì đầu não là những máy tính chuyên dụng gọi là **MÁY CHỦ**[1]. Máy chủ có dung lượng lưu trữ lớn, kết nối internet nhanh và chạy phần mềm cụ thể. Điều này giúp chúng lưu trữ tất cả thông tin website dễ dàng hơn.

[1] Server.

Máy tính của bạn tải các website bằng cách sử dụng danh sách quy tắc gọi là **GIAO THỨC TRUYỀN SIÊU VĂN BẢN (HTTP)**. Địa chỉ website bắt đầu với HTTP hoặc HTTPS (phiên bản bảo mật hơn để truyền file). HTTP là cách chính để gửi các website trên qua internet.

GIAO THỨC TRUYỀN SIÊU VĂN BẢN (HTTP)

Quy tắc chia sẻ nội dung web. Trình duyệt web dùng HTTP để giao tiếp với máy chủ.

GIAO THỨC TRUYỀN SIÊU VĂN BẢN BẢO MẬT (HTTPS)

Chức năng giống HTTP nhưng thêm lớp bảo mật bằng mã hóa tất cả các dữ liệu truyền đi.

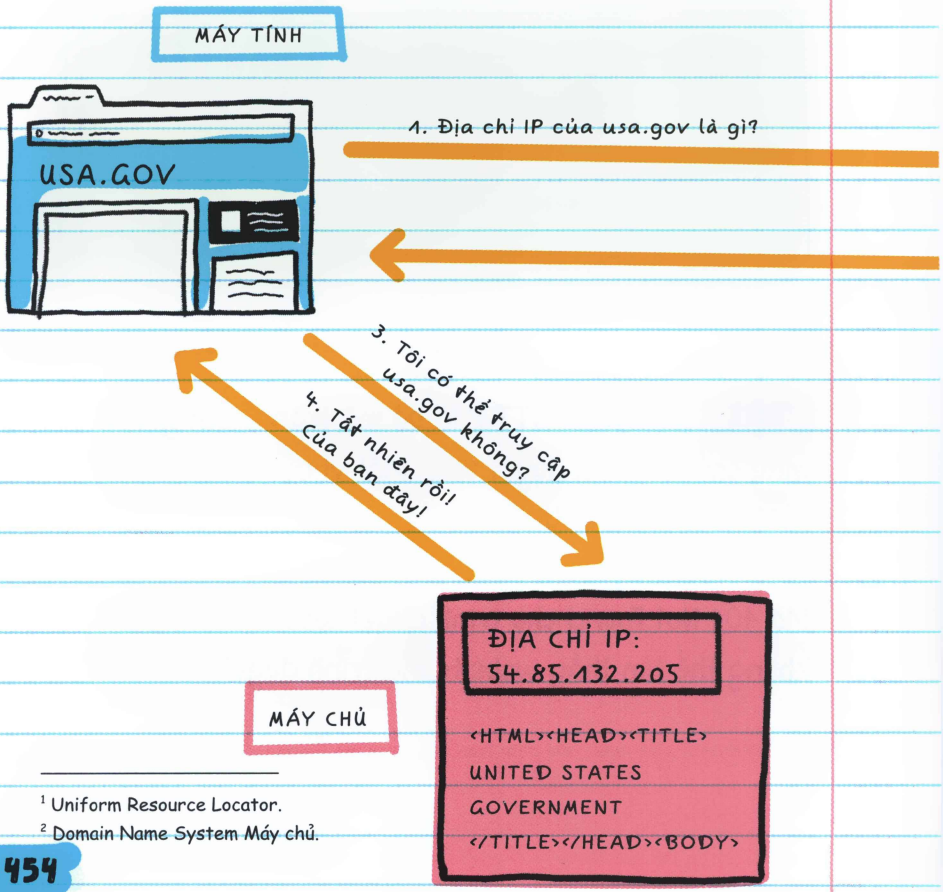
VÍ DỤ, một phần HTTP là bộ mã trạng thái gửi từ máy chủ tới trình duyệt web. Một ví dụ về mã trạng thái là 404 NOT FOUND.

Mã 404 Not Found cho trình duyệt web biết máy chủ không thể tìm thấy website mà trình duyệt web cần.

Tìm nơi website tồn tại

Mọi website trên internet đều có một **ĐỊNH VỊ TÀI NGUYÊN THỐNG NHẤT**[1] **(URL)** hoặc địa chỉ.

Trước khi bạn có thể truy cập một website, máy tính của bạn cần tìm ra máy chủ chứa trang bạn cần. Khi bạn gõ một địa chỉ web vào trình duyệt web, đầu tiên máy tính của bạn sẽ hỏi **MÁY CHỦ HỆ THỐNG TÊN MIỀN** cho

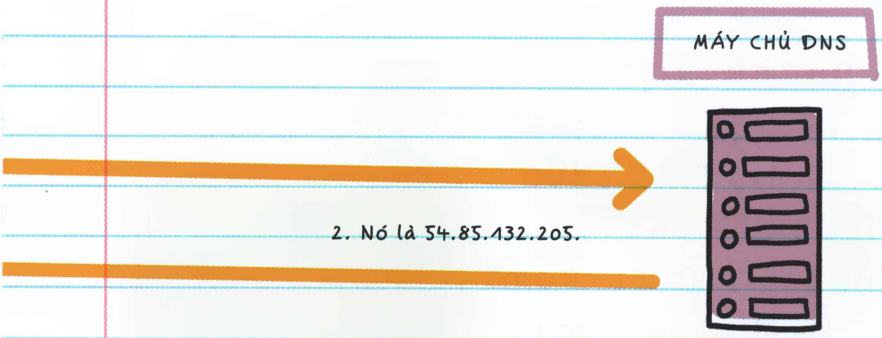


[1] Uniform Resource Locator.

[2] Domain Name System Máy chủ.

địa chỉ IP của máy chủ đang lưu site. Sau đó, máy tính của bạn có thể dùng địa chỉ IP đó để tìm chính xác máy chủ và hỏi máy chủ quyền truy cập file HTML tạo nên website. Tất cả những chuyện này diễn ra chưa đến 1 giây.

HỆ THỐNG TÊN MIỀN (DNS)
Cơ sở dữ liệu chuyển đổi URL của trang web sang địa chỉ IP.



Máy chủ DNS hoạt động giống như danh bạ trong điện thoại của bạn. Bạn chọn tên bạn bè trong danh bạ, và điện thoại sẽ gọi số kết nối với bạn của bạn, tương tự như việc chúng ta không phải nhớ địa chỉ IP của website. Nhưng bạn có thể gọi bạn bè bằng cách điền số điện thoại của họ, cũng như cách bạn có thể truy cập một website bằng cách điền địa chỉ IP của nó.

Những website nổi tiếng được quản lý bởi nhiều máy chủ trên khắp thế giới. Việc này giúp trang chạy nhanh khi có nhiều yêu cầu xem trang cùng một lúc. Nó cũng giúp ngăn ngừa sự cố mất điện nếu một máy chủ sập. Người dùng có thể được kết nối với máy chủ gần nhất.

Mỗi phút, có hàng triệu người sử dụng Internet xem các trang khác nhau. Đó là lý do tại sao Internet cần các giao thức để giữ mọi thứ được ngăn nắp.

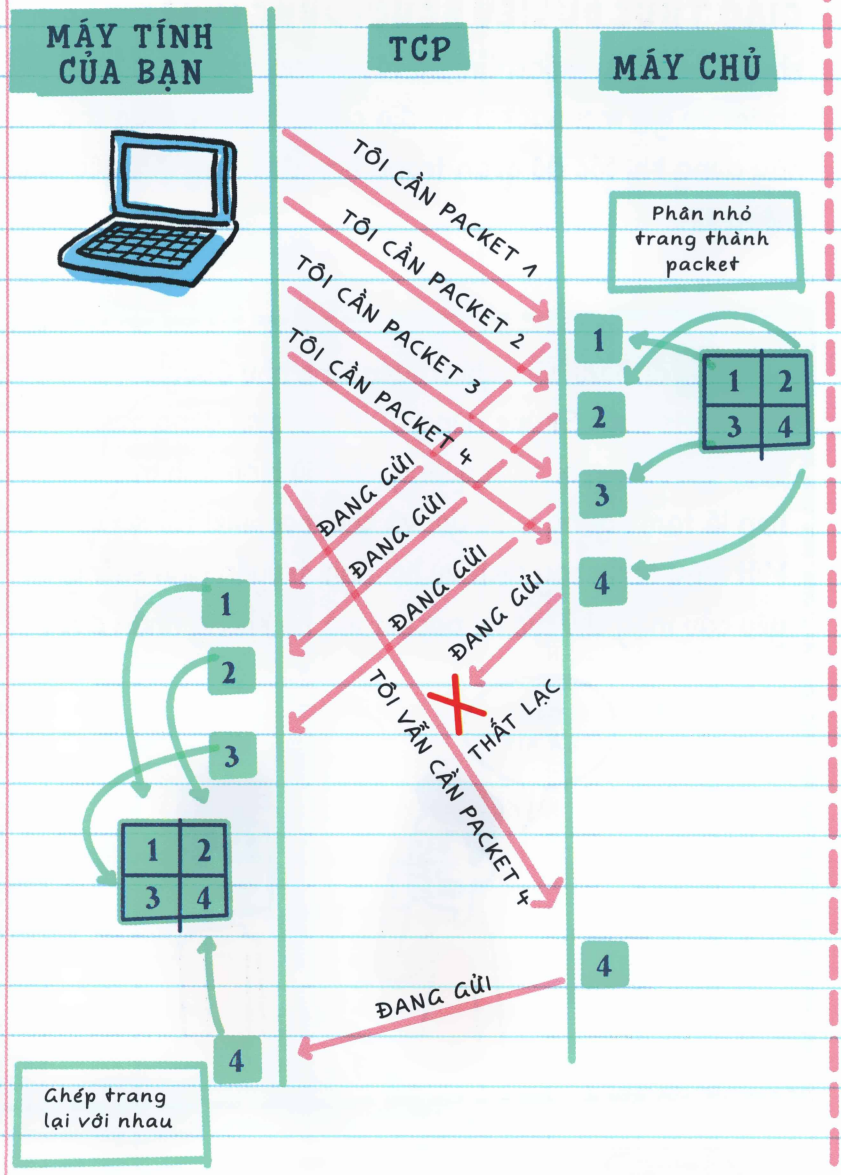
Giao thức điều khiển truyền vận[1]

Một khi máy tính của bạn kết nối với máy chủ, máy chủ sẽ tuân theo **GIAO THỨC ĐIỀU KHIỂN TRUYỀN VẬN (TCP)**. Thiết bị sử dụng TCP khi *tất cả* các thông tin được yêu cầu (website, ảnh, video) cần được truyền đi.

TCP
Loạt quy tắc máy tính dùng khi truyền thông tin qua internet.

VÍ DỤ, khi bạn tải một website, máy chủ gửi cho bạn thông tin trang dưới dạng packet nhỏ. Nếu packet bị lạc trên đường (thường là do kết nối kém), trình duyệt web của bạn sẽ gửi một yêu cầu mới cho packet bị mất. Nó sẽ tiếp tục gửi yêu cầu đến khi nhận được packet và cả trang được tải hoàn chỉnh.

[1] Transmission Control Protocol.


MÁY TÍNH CỦA BẠN
TCP
MÁY CHỦ
TÔI CẦN PACKET 1
TÔI CẦN PACKET 2
TÔI CẦN PACKET 3
TÔI CẦN PACKET 4
Phân nhỏ trang thành packet
1
2
3
4
ĐANG GỬI
ĐANG GỬI
ĐANG GỬI
ĐANG GỬI
THẤT LẠC
TÔI VẪN CẦN PACKET 4
ĐANG GỬI
Ghép trang lại với nhau

Giao thức dữ liệu người dùng[1]

GIAO THỨC DỮ LIỆU NGƯỜI DÙNG (UDP) là cách khác để truyền thông tin. Nó tập trung vào tốc độ hơn là nhiệm vụ gửi mọi packet tới địa chỉ cuối cùng. Giao thức này dùng khi tốc độ quan trọng hơn đáp ứng mọi yêu cầu thông tin.

VÍ DỤ, dịch vụ hội nghị truyền hình như Google Hangouts hoặc Skype dùng UDP vì người dùng muốn cuộc gọi không bị ngắt quãng, cho dù hình ảnh mờ, còn hơn là tạm ngừng cuộc gọi và chờ mọi pixel tải xong. Một máy tính chạy dịch vụ hội nghị truyền hình không yêu cầu máy chủ gửi lại packet mà nó không nhận được.

[1] User Datagram Protocol.

KIỂM TRA KIẾN THỨC

1. Sự khác biệt giữa internet và Mạng Toàn Cầu?

2. Packet là gì?

3. Tại sao giao thức mạng cần thiết để truyền thông tin và giao tiếp trên mạng?

4. Máy chủ DNS là gì?

5. Làm thế nào để máy tính biết thông tin gửi qua mạng là cho nó?

6. Giải thích giao thức mạng là gì. Tại sao giao thức mạng hữu dụng đối với giao tiếp mạng?

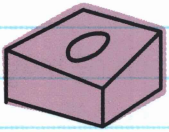
7. Máy chủ là gì và nó khác với máy tính cá nhân như thế nào?

8. "URL" là viết tắt của gì?

9. Tại sao những dịch vụ video như Skype ưu tiên dùng giao thức UDP?

10. Ai đã tạo những công cụ để lướt Mạng Toàn Cầu?

11. TCP là viết tắt của gì? Nó khác với UDP như thế nào?



ĐÁP ÁN

1. Internet là kết nối giữa các thiết bị, web là nội dung (bao gồm website, video, ảnh, v.v.).

2. Để gửi được qua mạng, file cần phải được phân nhỏ thành các packet. Packet là những mảnh nhỏ của file.

3. Nếu không có giao thức mạng, thông tin sẽ nhảy loạn lên và bị lạc. Giao thức mạng giúp đảm bảo thông tin được gửi đúng cách và tới đúng nơi.

4. Máy chủ DNS giống như danh bạ điện thoại. Nó là hệ thống dữ liệu tìm kiếm URL và cung cấp địa chỉ IP cho máy chủ chứa site.

5. Tất cả thông tin gửi qua mạng đều có địa chỉ đích đến đi kèm, như vậy thông tin sẽ tìm đúng máy tính nhận thông tin.

6. Giao thức mạng là một loạt các thủ tục hoặc quy tắc được tiêu chuẩn hóa. Giao thức mạng đảm bảo mọi người tuân theo cùng một hướng dẫn. Ethernet và Wireless được chạy bởi giao thức mạng, như vậy thông tin có thể được đi một cách gọn gàng.

7. Máy chủ là máy tính có kết nối internet rất nhanh và dung lượng lớn. Chúng chứa website, video, ảnh, và các thông tin mà người dùng có thể truy cập qua internet.

8. Định vị Tài nguyên Thống nhất (Uniform Resource Locator).

9. UDP là cách gửi thông tin nhanh hơn vì nó không đòi hỏi mọi mảnh thông tin được truyền đi một cách hoàn hảo. Mọi người thà có một cuộc gọi video hơi mờ còn hơn là bị ngắt quãng.

10. Tim Berners-Lee

11. Giao thức điều khiển truyền vận. TCP khác với UDP ở chỗ nó kiểm tra để chắc chắn mọi thông tin được yêu cầu đều đã được truyền đi, và gửi lại nếu thông tin chưa tới.

Chương 34

AN NINH MẠNG

AN NINH MẠNG LÀ GÌ?

AN NINH MẠNG là cần thiết để giữ thông tin kỹ thuật số an toàn. An ninh mạng là một loạt các kỹ thuật hoặc cách dùng để giữ thông tin bảo mật, giữ cho thông tin riêng tư, khả dụng, và không bị phá hoại.

Riêng tư: Thông tin không nên được chia sẻ với người khác cần được giữ bí mật.

Khả dụng: Những người có thẩm quyền có thể truy cập vào thông tin và file của họ lưu trên máy tính. Nếu ai đó cài vi-rút vào máy tính của bạn để ngăn không cho bạn dùng nó, họ đang khiến bạn không thể sử dụng thông tin của mình.

NGƯỜI CÓ THẨM QUYỀN
Người có sự cho phép.

Không bị phá hoại: File và thông tin trên máy tính không nên bị thay đổi

bởi người không có quyền. Nếu ai đó thay đổi mật khẩu của một trong những trang mạng xã hội của bạn và đăng nhập với tư cách của bạn, họ đã phá hoại tài khoản của bạn.

VI-RÚT
Phần mềm có hại và có thể lây lan sang máy tính khác.

KỸ THUẬT AN NINH MẠNG

Kỹ thuật an ninh mạng có thể:

- Ngăn người xấu tiếp cận thông tin.
- Khiến thông tin không thể sử dụng, ngay cả khi người không có quyền tác động vào nó.

Thông thường, nhiều kỹ thuật an ninh mạng được dùng để bảo vệ thông tin.

Xác thực

XÁC THỰC là quy trình mà máy tính dùng để xác nhận danh tính của một người trước khi cho họ quyền truy cập thông tin. Máy tính có thể dùng các kiểu xác thực khác nhau:

- Yêu cầu tên người dùng và mật khẩu
- Quét dấu vân tay
- Sử dụng máy ảnh để nhận diện khuôn mặt

XÁC THỰC HAI YẾU TỐ kết hợp hai phương pháp xác thực.

VÍ DỤ, để đăng nhập vào tài khoản thư điện tử có phương pháp xác thực hai yếu tố, bạn sẽ cần phải nhập mật khẩu và quét dấu vân tay trên điện thoại. Như vậy, những kẻ tấn công sẽ khó phá vỡ phương pháp xác thực hai yếu tố.

Kiểm soát truy cập

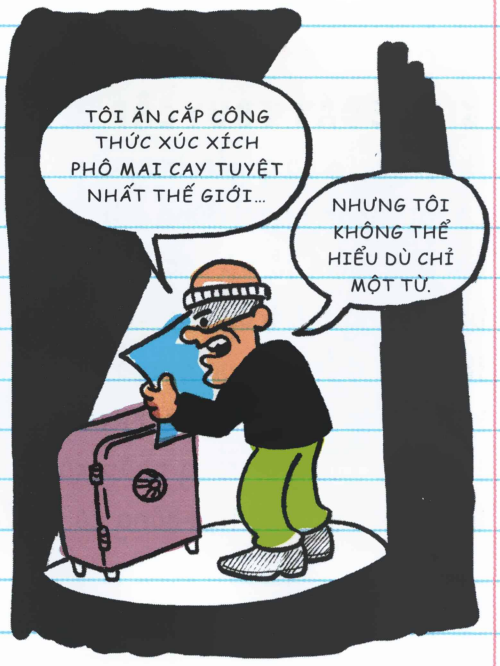
KIỂM SOÁT TRUY CẬP đề cập đến người dùng có thẩm quyền có quyền truy cập những file cụ thể. Nếu bạn có tài khoản người dùng ở trường, bạn có thể truy cập bài tập, điểm số, và các file khác mà bạn cần. Nhưng bạn không thể truy cập vào tài khoản của giáo viên để thay đổi điểm số của bạn, xem thông tin của học sinh khác, hoặc tải xuống bài kiểm tra tuần tới. Kiểm soát truy cập dùng phương pháp xác thực để ngăn người sai lấy sai dữ liệu mà họ không có quyền xem.

Sandboxing

SANDBOXING là đặt ứng dùng vào phần bảo mật của hệ thống, hạn chế quyền truy cập phần còn lại của hệ thống máy tính. Bằng cách này, nếu ứng dụng bị tấn công, chỉ ứng dụng bị ảnh hưởng, phần còn lại của hệ thống sẽ an toàn.

MẬT MÃ HỌC[1]

MẬT MÃ HỌC là chiến thuật bảo mật thông tin. Mục tiêu của mật mã học là đảo lộn thông tin và thông điệp đến mức không ai ngoài người có thẩm quyền mới có thể sắp xếp

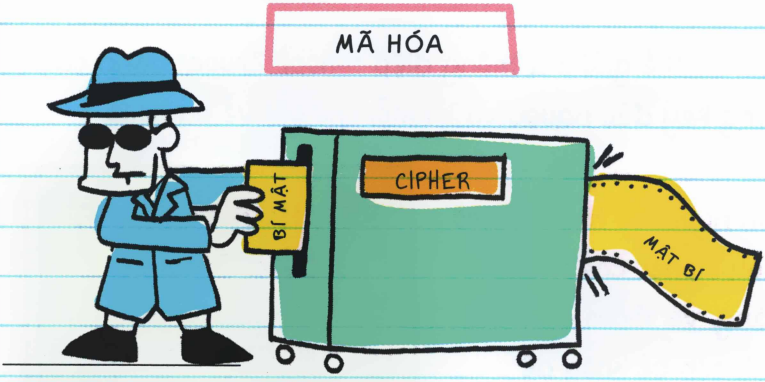


[1] Cryptography.

lại và đọc chúng. Việc này thêm một lớp bảo mật nữa bên cạnh kiểm soát truy cập, vậy nên ngay cả khi người không có thẩm quyền lấy được file bí mật, họ vẫn không thể đọc chúng.

Quá trình đảo lộn và sắp xếp lại thông tin gọi là mã hóa[1] và giải mã[2]. **MÃ HÓA** là khi thông tin đọc được chuyển hóa sang thông điệp bí mật, và **GIẢI MÃ** là quá trình biến thông điệp bí mật thành thông tin đọc được.

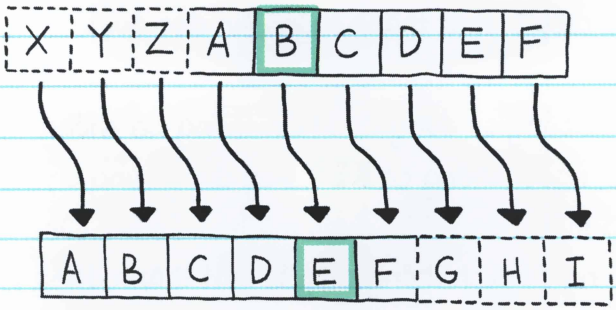
CIPHER là công cụ dùng để khiến thông tin không thể đọc đối với người không có **KEY**. Key dùng cùng cipher để giải mã một thông điệp được mã hóa. Cipher cũng là công cụ dùng để dịch thông điệp bí mật thành dạng ban đầu, có thể đọc được. Ví dụ, website của ngân hàng dùng cipher để dịch thông tin tài khoản khách hàng khi nó được chuyển giữa máy tính và máy chủ.



[1] Encryption.
[2] Decryption.

Cipher có tuổi đời lâu hơn máy tính nhiều. Kẻ độc tài người Roman, **JULIUS CAESAR** (100TCN – 44TCN) dùng một trong những cipher đời đầu nhất – nó được đặt tên theo hắn. Máy **CIPHER CAESAR** dịch chuyển mỗi chữ cái sang một vị trí cố định, ví dụ: ba bước trên bảng chữ cái.

Key của máy cipher Caesar là dịch mỗi chữ cái về phía trước theo số bước nhất định.



Ví dụ, dùng máy cipher Caesar, "Hello, World" sẽ được mã hóa thành "Khoor Zruog".

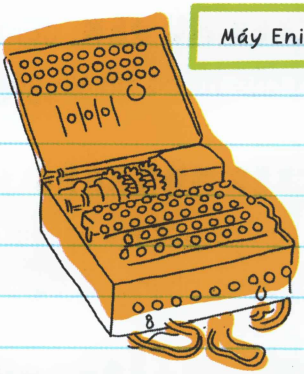
Bạn có thể giải mã thông điệp "Khoor Zruog" bằng cách dùng key đảo ngược và lùi mỗi chữ cái về ba bước.

Con người có thể bẻ khóa máy cipher Caesar dễ dàng, và thậm chí máy tính còn bẻ khóa dễ hơn nữa.

BẺ KHÓA MẬT MÃ HOẶC CIPHER

Khi ai đó không được cho biết key tìm cách giải mã thông điệp được mã hóa.

Trong Thế Chiến II (1939 – 1945), quân đội người Đức cải tiến máy cipher Caesar thành máy gọi là ENIGMA. Nó là sự kết hợp phức tạp của nhiều máy cipher khác nhau. Mỗi chữ cái được gõ sẽ đi qua nhiều lớp mã hóa, và sau đó key sẽ đổi cho chữ cái tiếp theo. Vậy nên hai chữ cái giống nhau trong thông điệp ban đầu sẽ được chuyển sang chữ cái khác nhau trong phiên bản được mã hóa.



Máy Enigma

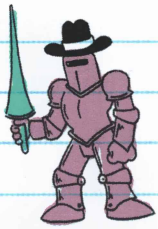
Alan Turing (người Anh, 1912 – 1954): Trong Thế Chiến II, Turing làm việc với một đội có nhiệm vụ giải mã cách máy Enigma của người Đức mã hóa. Thành quả của ông, máy BOMBE, là máy tính đời đầu thành công trong việc giải mã mật mã được mã hóa bởi Enigma.

HACKERS

HACKERS là những người xâm nhập vào máy tính và hệ thống máy tính lớn. Có nhiều loại hacker, không phải tất cả họ đều xấu.

Các công ty và chính phủ thuê hacker mũ trắng để giúp bảo vệ hệ thống và thông tin.

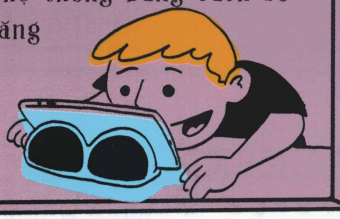
HACKER MŨ ĐEN đột nhập vào hệ thống để ăn cắp thông tin và gây thiệt hại.



HACKER MŨ TRẮNG truy tìm lỗi và sửa chúng trước khi hacker mũ đen có thể lợi dụng những lỗi đó để đột nhập.

NGƯỜI HOẠT ĐỘNG THEO CHỦ NGHĨA HACKER đột nhập vào hệ thống để đạt được mục đích chính trị hoặc xã hội.

Thuật ngữ "hacker" cũng có nghĩa là ai đó dùng đồ vật theo cách sáng tạo khác. Ví dụ, dùng kính râm để đỡ điện thoại gọi là hack, vì kính râm không được chủ đích dùng theo cách như vậy. Định nghĩa này xây dựng trên khái niệm hacker máy tính, người chiếm quyền truy cập hệ thống bằng cách sử dụng code, lỗi, và các tính năng công nghệ khác theo cách sáng tạo.



TẤN CÔNG

Hacker có thể đột nhập vào hệ thống bằng nhiều cách tấn công khác nhau.

Cách tấn công **BRUTE FORCE** là khi một hacker dùng

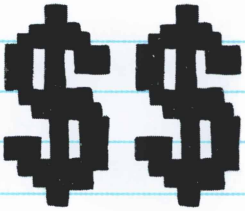
máy tính để đoán mọi khả năng xác thực. Ví dụ, đoán mọi sự kết hợp mật khẩu khả dĩ là cách tấn công brute force. Nếu bạn đặt mật khẩu là số có 4 chữ số, máy tính có thể đoán mọi số từ 0 đến 9999 để tìm ra mật khẩu của bạn trong vài giây. Đây là lý do tại sao website và ứng dụng thường yêu cầu bạn dùng ít nhất 8 ký tự bao gồm số, biểu tượng, và vài chữ cái viết hoa. Có hơn 600 nghìn tỷ sự kết hợp bạn có thể dùng với để tạo ra một mật khẩu như thế. Những mật khẩu như vậy dài hơn, phức tạp hơn và khó để bẻ khóa với cách tấn công brute force.

CÙNG LÀM TOÁN

Số định danh cá nhân (PINs) dài 4 ký tự và dùng 10 chữ số (0–9) có 10×10×10×10 hay 10^4 = 10.000 sự kết hợp khả dĩ. Nhưng một mật khẩu dài 8 ký tự và dùng 10 chữ số, 26 chữ cái viết thường, 26 chữ cái viết hoa, và 10 ký tự có 72^8 = 722 triệu sự kết hợp khả dĩ.

Một cách khác hacker dùng để truy cập vào hệ thống là sử dụng **MALWARE**. Malware bao gồm bất kỳ loại phần mềm độc hại nào được cài trên máy tính.

Malware bao gồm:

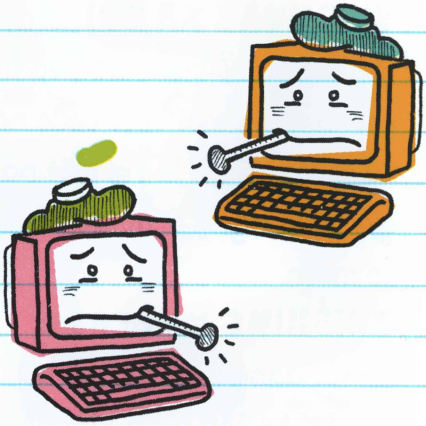


Ransomware: Không cho người dùng sử dụng máy tính cho đến khi họ trả tiền cho hacker.

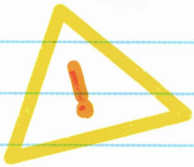
Spyware: Theo dõi người dùng để ăn cắp thông tin, ví dụ như tên người dùng hoặc mật khẩu.

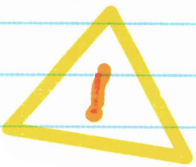
Trojan horse (hoặc Trojan): Bất kỳ loại phần mềm độc hại nào được ngụy trang như file bình thường.

Virus: Phần mềm lây lan như dịch bệnh đối với máy tính. Vi-rút cực kỳ nguy hiểm đối với mạng lưới vì chúng có thể lây lan tới tất cả các máy tính kết nối trong mạng lưới đó.



Worm: Phần mềm độc hại có thể tạo ra bản sao của chính nó và tự lây lan. Worm thường được lây lan bằng cách gửi qua thư điện tử số lượng lớn.





Không cần code

KỸ THUẬT XÃ HỘI là dạng tấn công phổ biến nhất, và nó không dùng một chút code nào. Kỹ thuật xã hội là khi một người giả vờ là người khác để có thứ họ muốn.

PHISHING là một trong những dạng kỹ thuật xã hội, khi một kẻ tấn công giả vờ có địa chỉ thư điện tử hoặc website hợp pháp để lấy tên người dùng và mật khẩu của bạn. .

VÍ DỤ. Kẻ tấn công có thể gửi một thư điện tử yêu cầu bạn đăng nhập vào một trong những tài khoản xã hội. Nhưng thay vì liên kết đến tài khoản thực sự của bạn, họ gửi cho bạn một trang họ xây nên để trông như bản thật. Khi bạn đăng nhập vào trang giả, bạn cho kẻ tấn công tên người dùng và mật khẩu của bạn, họ có thể dùng chúng để đăng nhập vào trang mạng xã hội thật với tư cách của bạn.

MẸO

Bảo vệ bản thân và không bao giờ để lộ thông tin được yêu cầu qua thư điện tử. Trang mạng xã hội bạn dùng đã biết mật khẩu của bạn – họ không bao giờ hỏi chúng (chỉ có những người dùng kỹ thuật xã hội). Khi nghi ngờ, đừng nhấn vào liên kết trong thư điện tử. Hơn nữa, hãy nhìn thật kỹ vào URL, www.amazon.com là trang hợp pháp, nhưng www.amaz0n.com có thể là trang phishing. Trang giả thay chữ o bằng số 0.

Tấn công từ chối dịch vụ phân tán (DDoS) là khi hacker dùng nhiều máy tính (thường bị nhiễm vi-rút) để gửi nhiều tin nhắn tới máy chủ cùng lúc để hệ thống bị quá tải và sập. Thông thường, hacker dùng tấn công DDoS vì mục đích ngẫu nhiên hoặc để thực hiện mục tiêu chính trị, ví dụ như làm sập website họ không đồng tình.

KIỂM TRA KIẾN THỨC

1. An ninh mạng là gì?

2. Kỹ thuật an ninh mạng nào xác nhận danh tính người dùng?

3. Kỹ thuật an ninh mạng nào cô lập ứng dụng phòng khi nó bị tấn công, những ứng dụng khác sẽ an toàn?

4. Dạng hacker nào đột nhập vào hệ thống một cách hợp pháp?

5. Tấn công brute force là gì?

6. Nối miêu tả với dạng walmare.

Ransomware	Có thể tự nhân bản và lây lan.
Spyware	Ngụy trang như file bình thường.
Trojan	Lây lan như dịch bệnh.

Virus	Ngăn người dùng sử dụng máy tính cho đến khi họ trả tiền.
Worm	Theo dõi người dùng để ăn cắp thông tin.

7. Cipher dùng để làm gì?

8. Thông điệp này được mã hóa bằng cipher Caesar bằng cách dịch mỗi chữ cái về phía trước 3 bước. Giải mã nó: **gdb**.

9. Mã hóa là gì?

10. Phishing là gì?

11. Tên cỗ máy của người Đức dùng để mã hóa thông điệp cho quân đội Đức trong Thế Chiến II?

ĐÁP ÁN

ĐÁP ÁN

1. Loạt kỹ thuật dùng để giữ thông tin an toàn.

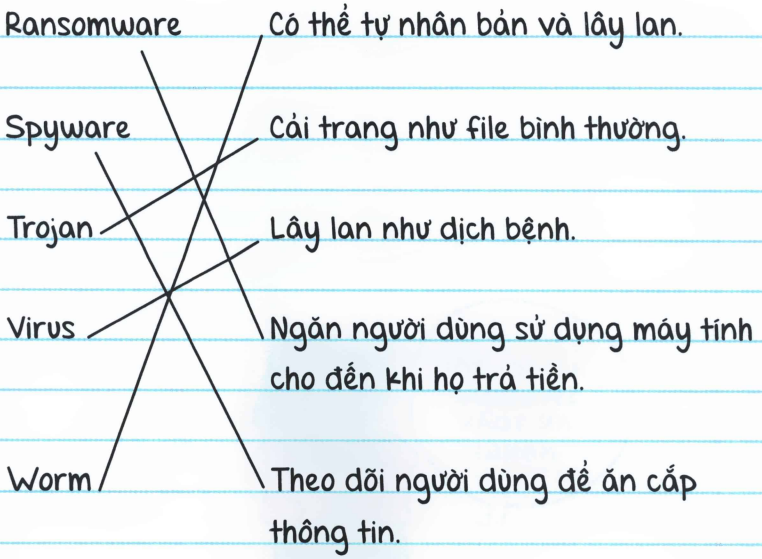
2. Xác thực

3. Sandboxing

4. Mũ trắng (cho dù ý định của người hoạt động theo chủ nghĩa hacker có thể là tốt, nhưng những gì họ làm vẫn là phạm pháp).

5. Dùng máy tính để đoán mọi khả năng xác thực.

6. Nối miêu tả với dạng malware.



7. Để mã hóa và giải mã thông điệp, khiến thông tin không thể đọc được đối với người không có key.

8. day

9. Mã hóa là quá trình đảo lộn thông tin để chỉ người có key mới có thể giải mã và hiểu được thông tin.

10. Phishing là dạng kỹ thuật xã hội mà ai đó giả vờ là người có thẩm quyền để lừa bạn chia sẻ thông tin cá nhân, ví dụ như tên người dùng và mật khẩu.

11. Enigma

HỌ GỌI TÔI LÀ CINDY AN TOÀN MẠNG!

Chương 35

TẠO MỘT TRANG WEB CƠ BẢN

XÂY DỰNG CÁC KHỐI WEBSITE

Tất cả website đều được tạo bằng **HTML CODE**. HTML được lập trình viên web chuyên nghiệp (người tạo website) dùng để sắp xếp và hiển thị văn bản, hình ảnh, video, và các thứ khác trên trình duyệt web. HTML dùng để thông báo cho trình duyệt web biết phải hiển thị gì.

CASCADING STYLE SHEETS (CSS) được website dùng để cho trình duyệt biết cách hiển thị trang. Bạn có thể dùng CSS để thay đổi màu văn bản, chèn hình nền, và nhiều hơn nữa.

HTML

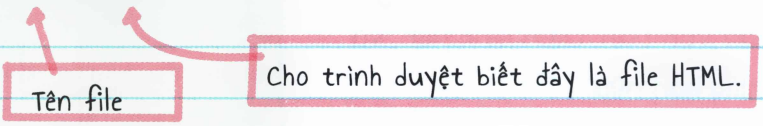
Viết tắt của **Ngôn ngữ Đánh dấu Siêu văn bản**. Phần "Siêu văn bản" nghĩa là nó kết nối các website lại với nhau, nên bạn có thể chuyển từ trang này sang trang khác.

Website đời đầu được tạo ra chỉ dùng ngôn ngữ HTML và phiên bản đời đầu của CSS, nghĩa là các website đó không có nhiều phong cách, sự tương tác hoặc đồ họa chuyển động. Hiện nay, có nhiều ngôn ngữ bên cạnh HTML và CSS để tạo website sống động và có tính tương tác. HTML và CSS dùng để tạo bố cục và giao diện của website, trong khi những ngôn ngữ khác như JavaScript và PHP dùng để chạy chương trình khiến trang có tính tương tác cao hơn.

CÔNG CỤ ĐỂ VIẾT WEBSITE

Lập trình viên web chuyên nghiệp đôi khi trả phí để dùng chương trình chỉnh sửa code đặc biệt, nơi họ viết code tạo nên website. Tuy nhiên, TEXT EDITOR được cài trước trên máy tính (Notepad hoặc TextEdit) cũng được sử dụng.

File HTML của website được lưu dưới dạng file ".html". Ví dụ, một website có tên là "index" nên được lưu là "index.html".



Bạn có thể đặt tên HTML tùy ý. "Index" thường là trang chủ của website được đặt tên. Hãy đảm bảo thêm đuôi ".html" sau tên file; nếu không trình duyệt web sẽ không đọc nó đúng cách.

XEM WEBSITE

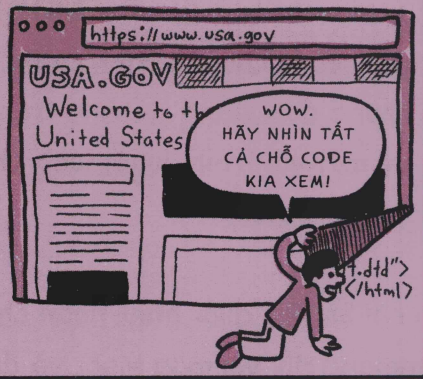
Để xem website, bạn cần **TRÌNH DUYỆT WEB**. Trình duyệt web là chương trình có thể đọc file HTML và hiển thị chúng dưới dạng website trên máy tính của bạn. Ví dụ về một vài trình duyệt web: Chrome, Safari, Microsoft Edge, và Firefox.

Tới nơi lưu file đuôi .html, nhấn chuột phải (hoặc nhấn command đối với máy Mac), và dưới menu "Open with", chọn một trình duyệt. Khi bạn mở file, trình duyệt sẽ hiển thị website.

TRANG WEB CƠ BẢN

Tất cả website đều chứa **NỘI DUNG**. Nội dung là văn bản, hình ảnh, và các phần khác của website mà người xem thấy khi họ nhìn vào trang trên **TRÌNH DUYỆT WEB.** Trình duyệt web là chương trình có thể đọc file HTML và hiển thị chúng dưới dạng website trên máy tính.

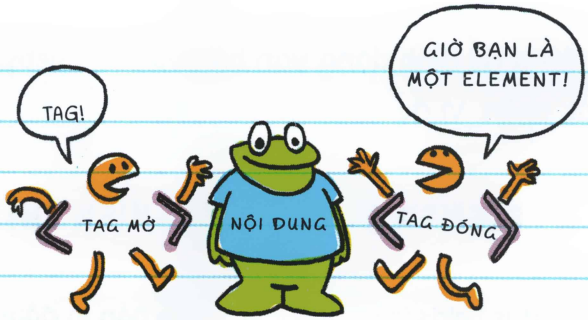
Bạn có thể đọc code HTML từ bất kỳ trang nào bằng cách nhấn chuột phải vào trang ở trình duyệt web và chọn "view page source" (xem nguồn trang). Một cửa sổ mới sẽ mở ra, cho thấy tất cả code của trang đó.



Tags và Elements

ELEMENTS là các phần của trang web, chúng giống như những khối xây dựng. Mỗi loại element thêm chi tiết, hoặc lớp, khác nhau vào thứ bạn muốn xây.

TAGS giống như dấu ngoặc kép. Dấu ngoặc kép mở cho bạn biết khi nào quote bắt đầu, và dấu ngoặc kép đóng cho bạn biết khi nào quote kết thúc. Tag mở và kết thúc hoạt động theo cách tương tự. Chúng bao quanh mỗi mảnh nội dung trong code cho website.



Tag quan trọng vì chúng cho máy tính biết khi nào element bắt đầu và kết thúc.

Element là tag và nội dung bên trong tag.

Element = Tag + Nội dung

Có nhiều dạng element, nhưng chúng đều bắt đầu với tag, theo sau bởi nội dung, sau đó kết thúc bằng tag:

Tag mở: Bạn đặt tên element trong dấu ngoặc nhọn.

```
<elementName>Nội dung ở đây . . . </elementName >
```

nội dung

Cả dòng này là một element.

Tag đóng: giống như tag mở nhưng thêm dấu gạch chéo.

Element dùng để định dạng văn bản và chia website thành các phần. Ví dụ:

ELEMENT	HIỂN THỊ LÀ
This text is bold	Văn bản in đậm
This is a heading	Văn bản tiêu đề
This text is emphasized	*Văn bản nhấn mạnh*
<button>This is a Button</button>	Đây là nút bấm

ELEMENT CHO WEBSITE

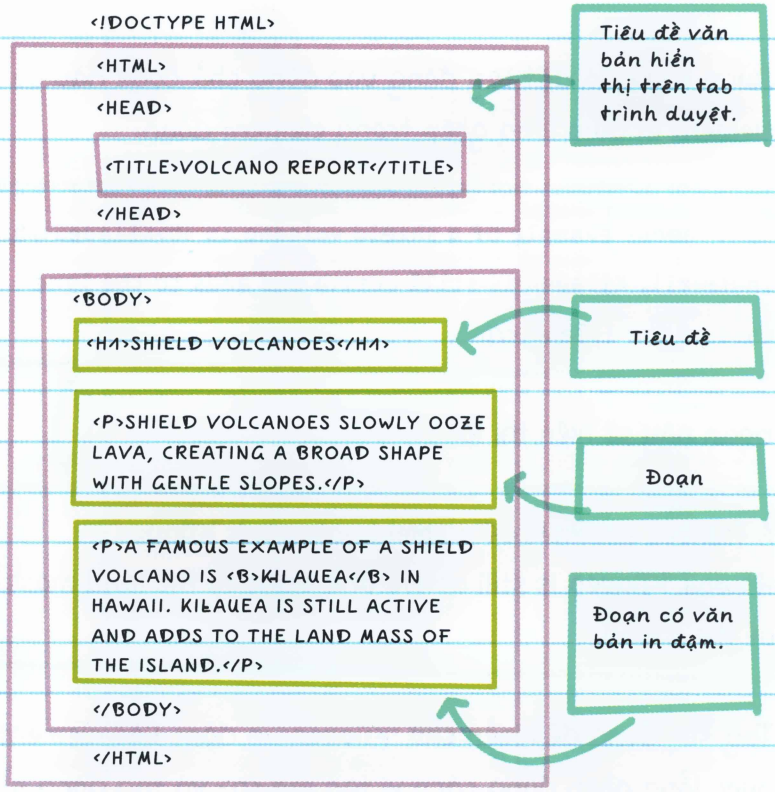
Mọi website nên bao gồm element <html>, <head>, và <body>.

ELEMENT <HTML> chứa tất cả những element khác giữa tag mở và kết thúc của nó. Element này đánh dấu nơi tệp HTML bắt đầu và kết thúc.

ELEMENT <HEAD> chứa element thêm thông tin về website. Hầu hết element trong **<head> element** không hiển thị nội dung trên website để người xem thấy. Những element này đưa thông tin cho trình duyệt web, bao gồm code CSS bổ trợ, tiêu đề trang, hoặc từ khóa để công cụ tìm kiếm tìm.

ELEMENT <BODY> là nơi chứa tất cả nội dung website mà bạn thấy. Văn bản, hình ảnh, video, hoạt họa, và bất kỳ thứ gì bạn muốn hiện thị trên trang đều nằm ở đây.

Phần đầu (head) và thân (body) của trang có thể trông như sau:



ĐỊNH DẠNG HTML

Thụt lề và khoảng cách không quan trọng trong HTML. Thụt lề element lồng ghép giúp sắp xếp gọn gàng mọi thứ. Thụt lề element <h1> trong đoạn văn bản dưới đây cho thấy rằng nó được lồng ghép trong element <body>.

```
<body>
    <h1> Shield Volcanoes</h1>
</body>
```

Quan trọng là đặt tag đóng vào đúng chỗ. Code này có element in đậm lồng ghép trong element đoạn:

```
<p>A famous example of a shield volcano is <b>Kīlauea</b>
in Hawaii. Kīlauea is still active and adds to the land
mass of the island.</p>
```

Đoạn này sẽ hiển thị là:

A famous example of a shield volcano is **Kīlauea** in Hawaii. Kīlauea is still active and adds to the land mass of the island.

Tag dùng để khiến chữ "Kīlauea" in đậm. Các element được lồng ghép đúng cách vì tag mở và tag đóng đều nằm trong tag mở và tag đóng .

Nếu tag của bạn được lồng ghép không đúng cách, nội dung có thể sẽ hiển thị sai. Sẽ khó mà giữ code trật tự và việc tìm kiếm lỗi sai sẽ khó hơn.

Lồng ghép sai cách:

```
<p>A famous example of a shield volcano is <b>Kīlauea in
Hawaii. Kīlauea is still active and adds to the land mass
of the island.</p></b>
```

Tag đóng đến sau tag đóng </p> làm rối trật tự lồng ghép và không hiển thị văn bản đúng cách.

HTML không bị ảnh hưởng bởi chữ viết hoa hay viết thường. Nghĩa là tất cả ví dụ dưới đây đều hoạt động giống nhau:

<BODY>	Viết hoa tất cả
<body>	Viết thường tất cả
<Body>	Viết hoa thường lẫn lộn

Nhưng thông thường, lập trình viên web sẽ gõ tag dưới dạng viết thường, hầu như là vì nó từng là một yêu cầu bắt buộc và giờ đã trở thành thói quen.

Bình luận

Bạn có thể thêm bình luận vào code HTML, trình duyệt sẽ không đọc được nó. Bình luận là lời nhắc cho chính bạn về chương trình. Chúng trông như sau:

```
<!-- Write your comments here -->
```

Dấu chấm than nằm ở đầu nhưng không nằm ở cuối.

Tag mở và kết thúc cần phải khác biệt để máy tính và lập trình viên không bị nhầm lẫn về nơi nội dung bắt đầu và kết thúc.

VÍ DỤ, tag mở <!-- khác tag đóng -->, và tag mở có dấu chấm than trong khi tag đóng thì không.

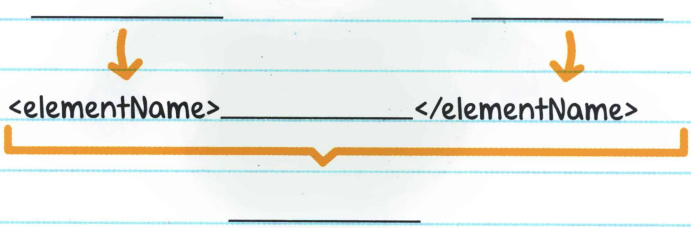
KHÔNG BÌNH LUẬN, LÀM ƠN. ĐÂY LÀ QUÀ TẶNG ĐẤY.
MEOW! MEOW! ME
OW! MEOW! MEOW! ME

KIỂM TRA KIẾN THỨC

1. Hai ngôn ngữ website được sử dụng nhiều nhất là gì? Ngôn ngữ kiểm soát phần nào của website?

2. HTML là viết tắt của từ gì?

3. Điền vào chỗ trống dưới đây để phân biệt mỗi phần.



4. Để code một file HTML, bạn có thể mua chương trình chuyên nghiệp, hoặc bạn có thể sử dụng một trong những chương trình cài sẵn: _____ hoặc _____.

5. Khi bạn lưu file HTML, đuôi tên file là gì?

6. Bạn cần chương trình gì để xem website?

7. Tên file thường dùng cho trang chủ của website là gì?

8. Tên của văn bản, hình ảnh, và các phần khác của website mà người xem thấy?

9. Phần bắt đầu và kết thúc của một element được đánh dấu bằng _____.

10. Viết ra một bình luận trong HTML.

11. Ba element HTML nào bạn nên dùng với mọi website bạn làm?

12. Tag HTML không bị ảnh hưởng bởi chữ viết hoa và chữ viết thường, điều này nghĩa là gì?

13. Bạn dùng element nào để đặt nội dung dễ thấy nhất mà bạn muốn người xem thấy?

14. Mệnh đề nào sau đây về khoảng cách đúng?

A. Khoảng cách trong HTML là yêu cầu cần thiết liên quan đến cách code chạy.

B. Khoảng cách trong HTML, ví dụ như thụt lề element, giúp lập trình viên đọc code, nhưng không ảnh hưởng đến cách code chạy.

15. Kiểu element nào nằm trong element <head>?

16. Mọi code nội dung website nên được thêm vào giữa tag _____ bắt đầu và kết thúc.

17. Cấu trúc của website được tạo nên bởi _____.

ĐÁP ÁN

1. HTML—kiểm soát bố cục
   CSS—kiểm soát thiết kế

2. Ngôn ngữ đánh dấu siêu văn bản.

3. Tag mở — Tag đóng

   <elementName> Nội dung </elementName>

   element

4. TextEdit đối với Mac, hoặc Notepad đối với máy tính khác.

5. .html

6. Trình duyệt web như Chrome, Microsoft Edge, Firefox, hoặc Safari.

7. index.html

8. Nội dung

9. Tag

10. <!-- đây là một bình luận -->

11. <html>, <head>, và <body>

12. Bạn có thể viết hoa tag tùy ý, và nó không ảnh hưởng đến code của bạn.

13. Element <body>

14. **B**

15. Element đưa thông tin thêm về website cho trình duyệt. Ví dụ, tiêu đề trang hay từ khóa được chứa trong element <head>.

16. <body>

17. Element HTML

Chương 36

ELEMENT VĂN BẢN CỦA HTML

ELEMENT TIÊU ĐỀ

Website dùng tiêu đề để chia văn bản và tiêu đề mỗi phần. **ELEMENT HEADING** khiến văn bản to hơn và đậm hơn.

Tối màu hơn

Có 6 tag tiêu đề: **<h1>**, **<h2>**, **<h3>**, **<h4>**, **<h5>**, và **<h6>**. Tiêu đề <h1> là bậc cao nhất - nghĩa là nó to và đậm nhất.

VÍ DỤ. tiêu đề <h1> dùng cho những tiêu đề trên cùng ví dụ như "Types of Volcanoes" (các loại núi lửa), và tiêu đề phụ <h2> có thể dùng cho các loại núi lửa ví dụ như "Shield," "Cinder Cone," và "Composite".

```
<h1>Types of Volcanoes</h1>
<h2>Shield</h2>
<h2>Cinder Cone</h2>
<h2>Composite</h2>
```

Code trên sẽ trông như sau trên trình duyệt web:

Types of Volcanoes

Shield

Cinder Cone

Composite

ELEMENT ĐỊNH DẠNG VÀ ELEMENT ĐOẠN

Element đoạn <p> dùng để tạo đoạn văn. HTML bỏ qua tất cả khoảng cách và dòng mới trong code, nên nếu bạn muốn văn bản hiển thị như một đoạn, sử dụng element <p>.

Bạn có thể dùng element <b> để khiến văn bản **in đậm** hoặc element <i> để kiến văn bản *in nghiêng*.

VÍ DỤ, bạn có thể đặt nội dung trong tag <p>, đặt văn bản cụ thể trong tag <b> để in đậm nó, và <i> để in nghiêng nó.

ĐOẠN

IN ĐẬM

```
<p>A famous example of a <b>shield volcano</b> is Kīlauea in Hawaii. <i>Kīlauea is still active</i> and adds to the land mass of the island.</p>
```

IN NGHIÊNG

Đoạn trên sẽ trông như sau:

A famous example of a **shield volcano** is Kīlauea in Hawaii. *Kīlauea is still active* and adds to the land mass of the island.

ĐƯỜNG KẺ NGANG VÀ TÁCH DÒNG

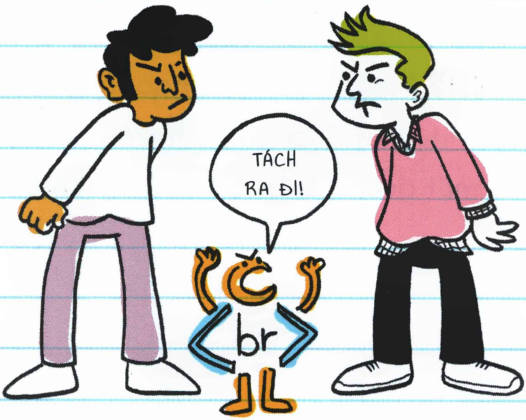
ELEMENT <HR> tạo đường kẻ ngang chia nội dung bằng cách hiển thị một đường mảnh trên trang web. Trông như vẽ một đường trên giấy để chuẩn bị cho một chủ đề mới.

Bạn có thể thêm element <hr> vào bên phải element tiêu đề <h1> để thể hiện sự chia cắt giữa các phần.

```
<h1>Types of Volcanoes</h1>
<hr>
<h2>Shield</h2>
<p>A famous example of a <b>shield volcano</b> is Kīlauea
in Hawaii. <i>Kīlauea is still active</i> and adds to the
land mass of the island.</p>
<h2>Cinder Cone</h2>
<h2>Composite</h2>
```

Đường kẻ ngang giữa tiêu đề <h1> và <h2>.

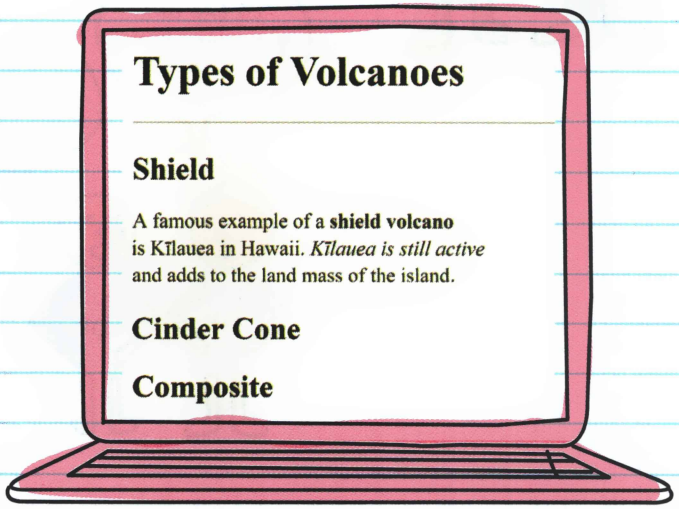
ELEMENT TÁCH DÒNG
 hiển thị một dòng trống giữa các element. Nó có thể dùng để chia cắt các phần hoặc thêm khoảng trống vào website.



```
<h1>Types of Volcanoes</h1>
<hr>
<h2>Shield</h2>
<p>A famous example of a <b>shield volcano</b> is Kīlauea
in Hawaii. <i>Kīlauea is still active</i> and adds to the
land mass of the island.</p>
<br>
<h2>Cinder Cone</h2>
<h2>Composite</h2>
```

Element tách dòng giữa element <p> và <h2>.

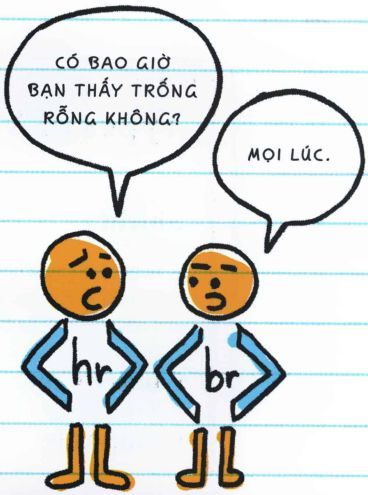
Với element tách dòng và đường kẻ ngang, trang web sẽ trông như sau:



ELEMENT RỖNG

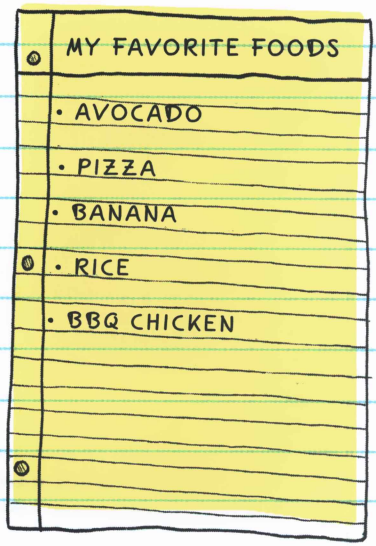
Hầu hết element HTML có tag mở và kết thúc với nội dung bên trong. Một số element không có nội dung và tag đóng. Chúng được gọi là **ELEMENT RỖNG**.

Element **<hr>** và **
** gọi là **element rỗng** vì chúng không có tag đóng.



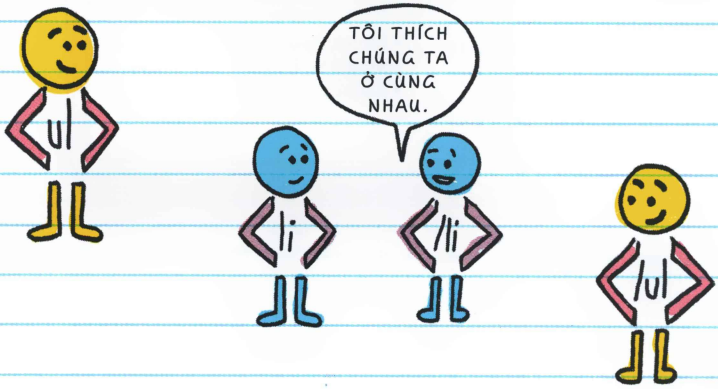
ELEMENT DANH SÁCH

ELEMENT DANH SÁCH định dạng văn bản thành danh sách được gạch đầu dòng hoặc đánh số. Danh sách dùng một tổ hợp element.



ELEMENT DANH SÁCH KHÔNG CÓ THỨ TỰ dùng để hiển thị danh sách các mục. Ví dụ, bạn có thể dùng danh sách không có thứ tự để tạo một danh sách bạn bè hoặc danh sách đồ ăn bạn thích ăn.

Để khiến mỗi mục chia cách (nằm ở dòng riêng biệt), bạn có thể dùng **LIST ITEM ELEMENT** cho mỗi mục trong danh sách. Luôn dùng element danh sách phần tử <li> bên trong element danh sách không có thứ tự <ul>.



VÍ DỤ, ta có thể tạo một danh sách không có thứ tự về những núi lửa nổi tiếng bằng cách sử dụng element <ul> và <li>:

```
<h1>Volcanoes</h1>
<hr>
<h2>Famous Volcanoes:</h2>
<ul>
    <li>Mount Vesuvius</li>
    <li>Mount St. Helens</li>
    <li>Kilauea</li>
</ul>
```

Mỗi phần tử đều có tag <li> riêng.

Cả danh sách được nằm trong tag <ul>.

Trang web sẽ trông như thế này:

DANH SÁCH CÓ THỨ TỰ dùng tag <ol> và hiển thị các mục được đánh số. Danh sách có thứ tự dùng để cho biết 10 bước nhảy được yêu thích nhất hoặc danh sách các bước theo trình tự. Mỗi phần tử đều cần element danh sách phần tử <li>.

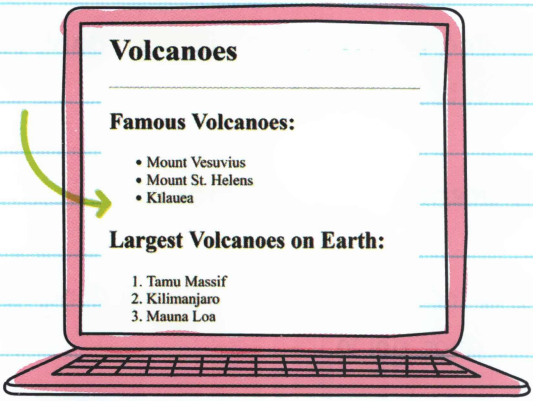
VÍ DỤ, bạn có thể tạo danh sách những núi lửa lớn nhất trên Trái Đất:

```
<h2>Largest Volcanoes on Earth:</h2>
<ol>
    <li>Tamu Massif</li>
    <li>Kilimanjaro</li>
    <li>Mauna Loa</li>
</ol>
```

Mỗi phần tử đều có tag <li> riêng.

Cả danh sách được nằm trong tag <ol>.

Trang web sẽ trông như thế này:



KIỂM TRA KIẾN THỨC

1. Code sau gặp lỗi ở đâu?

```
<p>This is a paragraph</b>
```

2. Code sau gặp lỗi ở đâu?

```
<p>I like to <b>Move it, Move it</p></b>
```

3. Xếp hạng element tiêu đề từ lớn nhất đến nhỏ nhất.

4. Bạn dùng tag gì để khiến văn bản in đậm?

5. Bạn dùng tag gì đến khiến văn bản in nghiêng?

6. Sự khác biệt giữa hai element danh sách <ul> và ?

7. Những phần tử riêng biệt trên danh sách được xác định bằng tag nào?

8. Element <p> dùng như thế nào?

9. Nếu một element không có nội dung hoặc tag đóng, nó được gọi là _____.

10. Element <br> có nhiệm vụ gì?

11. Element nào sau đây là element rỗng?

A.

B.

C. <p>

D. <h3>

12. Element <hr> có nhiệm vụ gì?

ĐÁP ÁN

1. Tag mở và tag đóng phải có cùng tên.

2. Code không được lồng ghép đúng cách. Vị trí của tag và cần được đổi cho nhau.

3. <h1>, <h2>, <h3>, <h4>, <h5>, <h6>

4.

5. <i>

6. dùng cho danh sách không có thứ tự (gạch đầu dòng), trong khi <ol> dùng cho danh sách có thứ tự (đánh số).

7. Danh sách phần tử hoặc <li>.

8. Element dùng để biến văn bản thành một đoạn.

9. Element rỗng.

10. Element <br> tạo tách dòng hoặc dòng trống giữa hai element.

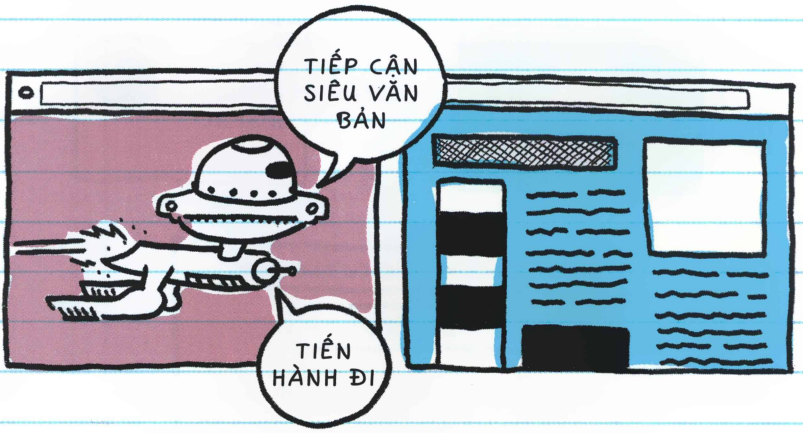
11. **A**

12. Element <hr> vẽ đường kẻ ngang giữa hai element trên trang web.

MỘT LẦN NỮA, TÔI LẠI CÔ ĐƠN.
hr

Chương 37

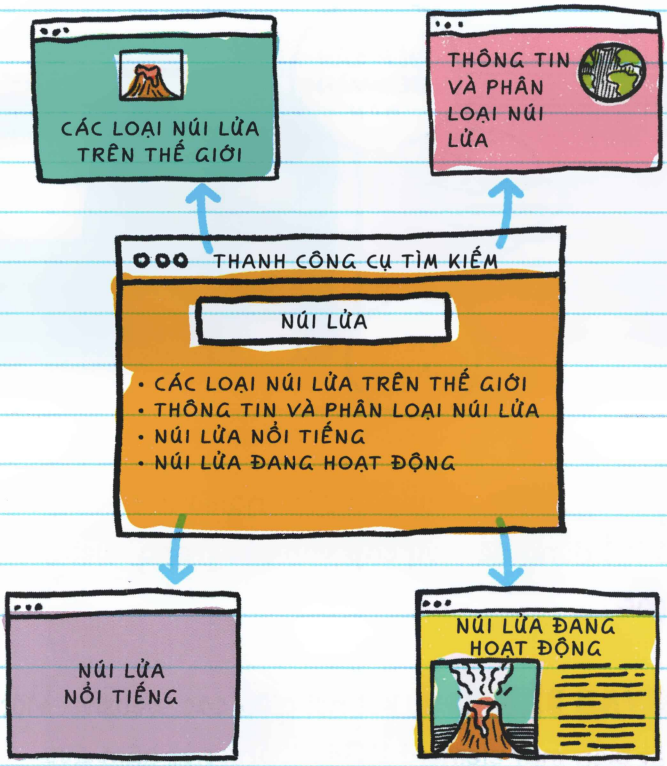
ELEMENT LIÊN KẾT



Trong HTML, phần "siêu văn bản" nghĩa là những liên kết có thể dùng để truy cập vào các trang, website và file khác.

SIÊU VĂN BẢN là sự kết nối giữa các website. Một liên kết biến một element HTML thành nút bấm đưa người dùng tới website mới.

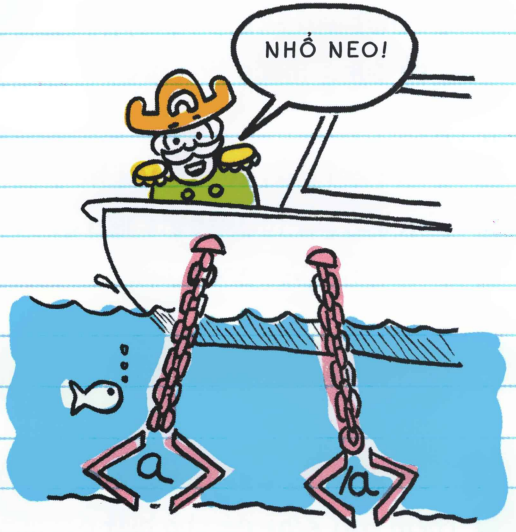
Đường liên kết là thứ khiến website hữu dụng. Người dùng có thể liên tục nhảy sang thông tin liên quan hoặc trang web khác. Ví dụ, khi bạn tìm từ "volcanoes" (núi lửa) trong thanh công cụ tìm kiếm, bạn sẽ nhận được một danh sách website có nhắc đến núi lửa. Bạn có thể nhấn vào bất kỳ kết quả tìm kiếm nào và nhảy tới website, vì công cụ tìm kiếm đã thêm đường liên kết cho mỗi trang được liệt kê.



Liên kết cũng bổ trợ cho **NAVIGATION MENU** trên website. Navigation menu là tổ hợp liên kết, mỗi liên kết dẫn tới một địa điểm cụ thể trên website. Liên kết cho ta ghé thăm từng trang trên website chỉ với một cái nhấn chuột.

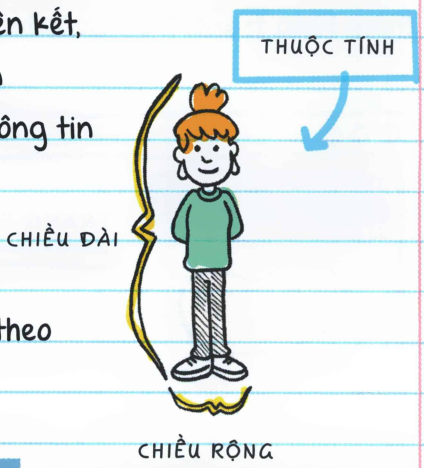
ELEMENT MỎ NEO <a> dùng để thêm liên kết vào website.

Bên cạnh việc đặt tên liên kết vào <a> và , bạn cần phải thêm địa chỉ web cho website nơi người được chuyển đến.



THUỘC TÍNH thêm thông tin bổ trợ về element. Ví dụ, bạn có thể dùng thuộc tính để thêm địa chỉ website vào liên kết, chiều dài và chiều rộng bạn muốn cho ảnh, và những thông tin phụ trợ khác.

THUỘC TÍNH
Thông tin thêm nằm trong element.



Thuộc tính được bao gồm bên trong tag mở và tuân theo khuôn mẫu sau:

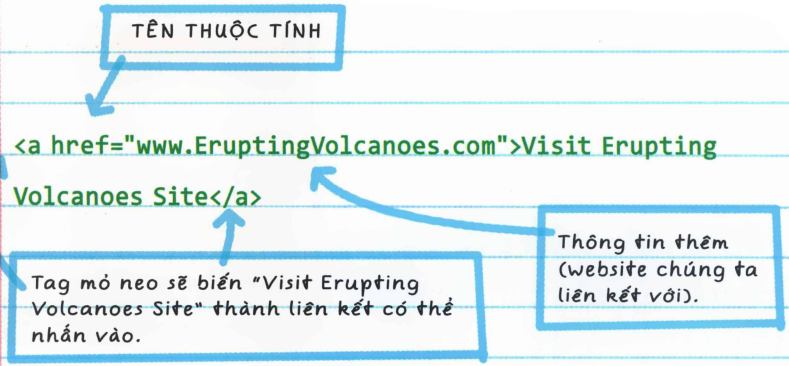
tên thuộc tính="information"

Website bạn muốn chuyển đến.

Để tạo một liên kết tới website khác, sử dụng thuộc tính **href**.

Dùng element mỏ neo <a> với thuộc tính herf để thêm liên kết tới website liên quan tới núi lửa "www.EruptingVolcanoes.com":

1. Đặt tên website bạn muốn ghé thăm bằng cách sử dụng .
2. Thêm địa chỉ web đóng vai trò liên kết.
3. Thêm tag để cho biết nơi liên kết kết thúc.



ELEMENT HÌNH ẢNH

Hình ảnh giúp website thú vị và hấp dẫn hơn.

ELEMENT HÌNH ẢNH <img> dùng để thêm hình ảnh vào website. Element <img> là element rỗng, vậy nên nó không có tag đóng.

THUỘC TÍNH SRC (nguồn) dùng để chỉ định tên file và nơi chứa file hình ảnh, như vậy trình duyệt có thể tìm thấy file và hiển thị nó trên website. File hình ảnh cần được lưu với file website hoặc được tham chiếu từ website khác.

Ví dụ, bạn có thể dùng địa chỉ web của hình ảnh núi lửa từ website khác để thêm nó vào trang web núi lửa. Khi xem website, nhấn chuột phải vào hình ảnh trên cửa sổ (nhấn command đối với Mac) và chọn "copy image address" (sao chép địa chỉ hình ảnh) trong menu. Hành động này sẽ sao chép địa chỉ lưu hình ảnh. Bạn có thể dùng địa chỉ này làm giá trị cho thuộc tính src của bạn.

> Đừng lo lắng nếu hình ảnh quá lớn hoặc quá nhỏ – nó luôn có thể được thay đổi kích thước bằng cách sử dụng CSS sau này.

Địa chỉ hình ảnh được sao chép từ website khác sẽ có định dạng như sau:

"www.eruptingvolcanoes.com/images/volcano1.jpg"

A B C

Hình ảnh lưu cùng website thường được giữ trong các file. Địa chỉ của chúng được xác định trong địa chỉ web. Phần đầu tiên sẽ là website (A). Sau đó là gạch chéo và tên của thư mục chứa hình ảnh (B). Sau đó là một gạch chéo nữa và tên của hình ảnh với file mở rộng (C).

Để liên kết tới hình ảnh, đầu tiên bạn cần thêm element <img> sau đó dán địa chỉ web của hình ảnh vào phần giá trị thuộc tính src.

Gõ phần này vào.

Dán địa chỉ hình ảnh mà bạn đã sao chép vào đây.

```
<img src="www.eruptingvolcanoes.com/images/volcano1.jpg">
```

Chỉ liên kết tới hình ảnh bạn sở hữu hoặc được cho phép sử dụng với sự ghi nhận đúng cách.

KIỂM TRA KIẾN THỨC

1. Element <hr> và <img> có điểm chung gì?

2. Liên kết là gì?

3. Tại sao liên kết hữu dụng đối với website?

4. Phần nào của element dưới đây là thuộc tính?
Click Here

5. Nhìn vào những dòng code dưới đây. Bạn dùng element nào để thêm liên kết vào website?
   A. <link src="www.website.com">click here</link>
   B. <link href="www.website.com">click here</link>
   C. click here
   D. click here

6. Loại thuộc tính nào có trong element dưới đây?
All The Links

7. Bạn dùng element nào để thêm ảnh vào trang?

8. Bạn cần loại thuộc tính nào để thêm tên file ảnh vào element hình ảnh?

9. Code sau gặp lỗi gì?

```
<a href="">click here</a>
```

10. Khoanh tròn code không cần thiết trong dòng dưới đây:

```
<img src="catapalooza.jpg"></img>
```

ĐÁP ÁN

1. <hr> và <img> đều là element rỗng.

2. Liên kết là sự kết nối với website khác hoặc nội dung khác trên internet.

3. Liên kết cho phép bạn nhảy sang trang, thông tin, video, hình ảnh khác... một cách dễ dàng và ngay lập tức.

4. Click Here

5. D

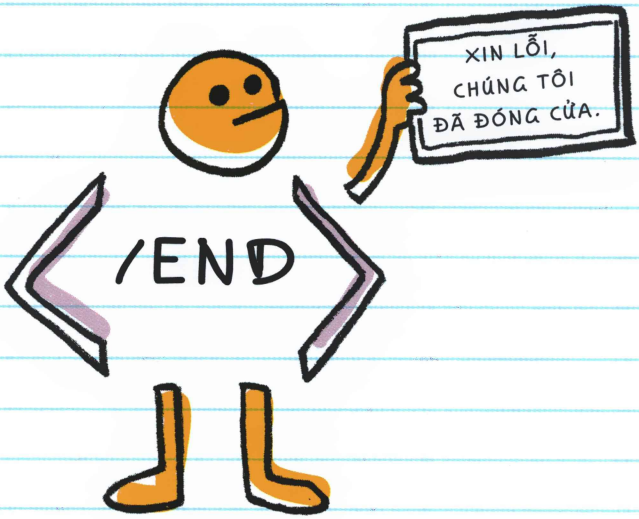
6. href

7. <img>

8. src

9. Code thiếu địa chỉ web giữa dấu ngoặc kép của thuộc tính herf.

10. <img src="catapalooza.jpg"></img>
Vì element <img> là element rỗng, tag đóng là không cần thiết.



TÔI CẢM THẤY THẬT KẾT NỐI VỚI BẠN.
CLICK HERE
BABYSLOTH.COM

Chương 38

TẠO PHONG CÁCH VỚI CSS

CASCADING STYLE SHEETS (CSS)

Tự bản thân HTML có thể tạo nên những website hữu dụng, nhưng nó không hiệu quả trong việc tạo nên những website đẹp. **CASCADING STYLE SHEETS** (CSS) là ngôn ngữ dùng để tạo phong cách cho file HTML. CSS mô tả màu sắc, kích cỡ, bố cục, hình nền, phông chữ, và các tính năng thiết kế khác cho website. Nó cũng cho phép bạn tạo phong cách khác nhau cho trang web trên thiết bị khác nhau (ví dụ như thiết bị cầm tay, máy tính xách tay, và máy tính để bàn) trong khi vẫn giữ nguyên nội dung HTML.

Có nhiều cách để thêm CSS vào website:

- Bạn có thể thêm CSS trực tiếp vào file HTML ở phần đầu hoặc thêm dưới dạng thuộc tính vào trong mỗi element HTML.

- Bạn có thể tạo một file CSS riêng biệt để chứa thông tin. Việc này giúp code của bạn dễ đọc và dễ sắp xếp hơn.

Đây là cách tốt hơn!

TẠO FILE CSS

Để tạo một file CSS mới, sử dụng cùng chương trình mà bạn dùng để tạo file HTML (Textedit hoặc Notepad). Bạn có thể đặt tên file CSS tùy ý, nhưng nhiều lập trình viên lưu file CSS của website là **style.css**. Để đơn giản, lưu file CSS của bạn trong cùng thư mục với file HTML.

Tạo một phong cách CSS

PHONG CÁCH được tạo nên bởi sự kết hợp **SELECTOR** với **DECLARATIONS**.

PHONG CÁCH
Thay đổi cụ thể đối với vẻ ngoài của nội dung HTML.

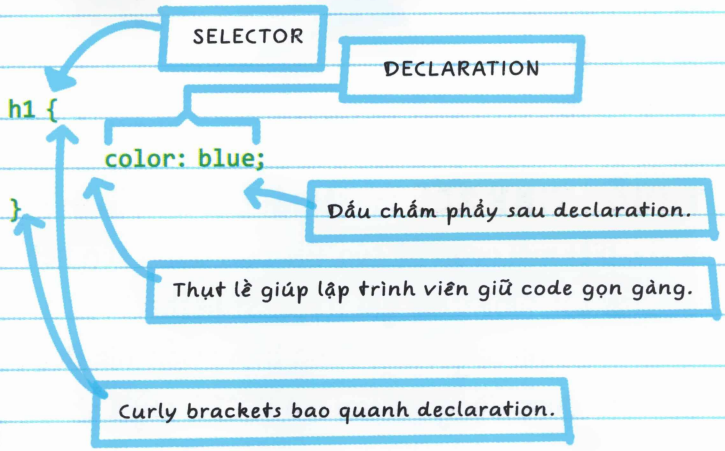
Selector gán phong cách cho một element cụ thể trong file HTML. Selector là tên của element không có ngoặc nhọn <>.

Declaration là thay đổi trong diện mạo. Ví dụ, bạn biến chữ thành màu tím hoặc tạo viền quanh ảnh.

Selector = element được tạo phong cách.
Declaration = Phong cách mới trông như thế nào.

Đặt (những) declaration vào trong curly brackets. Thụt lề đối với mỗi declaration để giữ code gọn gàng. Tương tự như HTML, CSS không bị ảnh hưởng bởi khoảng cách. Nếu bạn xóa mọi khoảng cách và thụt lề, code vẫn chạy bình thường.

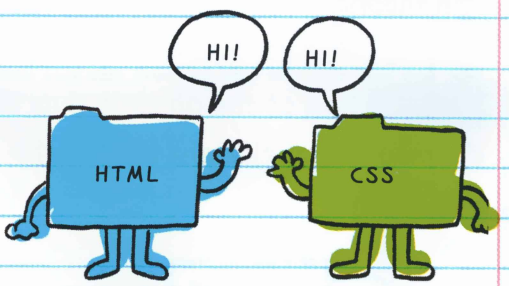
VÍ DỤ, bạn có thể gõ code dưới đây vào file .css để đổi màu tất cả element tiêu đề <h1> trong website thành màu xanh:



Phong cách cùng declaration được định dạng cách dòng và có curly bracket ở dòng cuối, đây là một gợi ý để giúp lập trình viên đọc code dễ hơn. Lập trình viên có thể viết code trong một dòng như thế này **h1{color:blue;}** và máy tính vẫn đọc giống hệt.

Liên kết CSS và file HTML

Trước khi phong cách mới trong file CSS có thể áp dụng cho website, file cần được liên kết. Bạn có thể áp dụng phong cách trong file CSS cho website HTML bằng cách thêm element vào phần đầu của HTML. Element liên kết cần 3 thuộc tính:



1. **THUỘC TÍNH REL** là viết tắt của "mối quan hệ" và cho biết mối quan hệ của file bạn đang liên kết vào element <link>. Khi giá trị là "stylesheet", bạn đang cho trình duyệt biết rằng bạn muốn file được nhắc đến được dùng với vai trò là CSS style sheet cho website.

một mẫu hiển thị phong cách và cách sắp xếp bố cục.

2. **THUỘC TÍNH TYPE** cho biết kiểu file - ví dụ file văn bản/CSS.

3. **THUỘC TÍNH HERF** xác định tên và địa chỉ của file CSS. Nếu file CSS ở trong cùng địa chỉ với file HTML, thì bạn chỉ cần thêm tên file CSS.

Dưới đây là định dạng để liên kết "style.css" với file HTML bằng element liên kết:

```
<head>

    <link rel="stylesheet" type="text/css" href="style.css">

</head>
```

Thuộc tính Rel.

Thuộc tính Herf: Tên của file CSS.

Thuộc tính Type: Chỉ định đây là file CSS.

Đôi khi lập trình viên lưu file CSS ở thư mục riêng (thường có tên là "css") trên máy chủ website của họ. Nếu bạn làm vậy, thêm tên thư mục và dấu gạch chéo vào giá trị herf như sau:

Tên thư mục.

```
<head>
    <link rel="stylesheet" type="text/css" href="css/style
    .css">
</head>
```

Element <link> cùng với tất cả thuộc tính của nó, báo trình duyệt web tìm file "style.css" và áp dụng các phong cách trong file đó vào thư mục HTML.

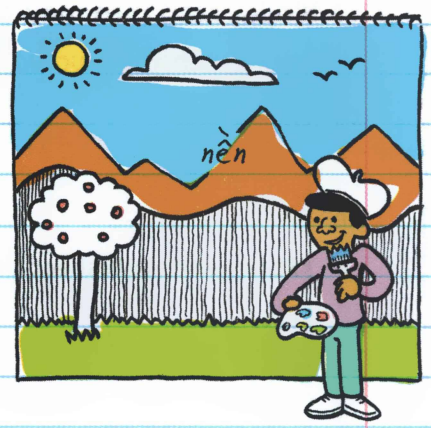
Sau khi liên kết file CSS với fie HTML, mọi thay đổi với CSS sẽ thay đổi cách website hiển thị khi trình duyệt được tải lại.

THUỘC TÍNH MÀU SẮC CỦA CSS

Có ba thuộc tính mà bạn có thể sử dụng trong một phong cách để thay đổi màu sắc của trang web.

Màu sắc thay đổi màu sắc của văn bản trong element.

Nền thay đổi màu sắc nền của khoảng không gian mà element chiếm. Ví dụ, đặt màu nền của element <body> thành màu xám sẽ biến cả trang thành màu xám, nhưng nếu thay đổi màu nền của element <p> thành màu hồng sẽ chỉ biến nền của đoạn thành màu hồng.



Màu viền thay đổi màu của viền nếu element có viền. Nếu nó không có viền, phong cách này sẽ không tạo ra thay đổi gì cả.

Có vài cách khác nhau để chỉ định màu sắc bằng CSS:

Tên màu sắc

HEX

RGB

Tên màu sắc

TÊN MÀU SẮC là tổ hợp 147 màu đã có tên cụ thể và có thể dùng trong CSS.

Sử dụng tên màu sắc tiện lợi vì bạn không phải ghi nhớ hoặc tìm code màu. Nhưng nó có thể gây hạn chế vì chỉ có 147 màu.

VÍ DỤ, bạn có thể dùng màu "red" (đỏ) làm giá trị cho thuộc tính màu để biểu thị màu đỏ thuần.

Định dạng để cập màu sắc như sau: **thuộc tính: giá trị;**

```
color: red;
```

THUỘC TÍNH

GIÁ TRỊ (TÊN MÀU: RED)

Ví dụ về tên màu sắc:

VÍ DỤ, bạn có thể biến văn bản trong element tiêu đề <h1> thành màu đỏ bằng cách sử dụng giá trị "tomato" cho thuộc tính màu sắc:

```
h1 {
        color: tomato;
}
```

Hexadecimal (HEX)

HEXADECIMAL, hoặc HEX, là dạng code dùng để biểu thị hơn 16 triệu màu. Code dùng 6 ký tự và kết hợp số 0-9 và chữ cái A-F.

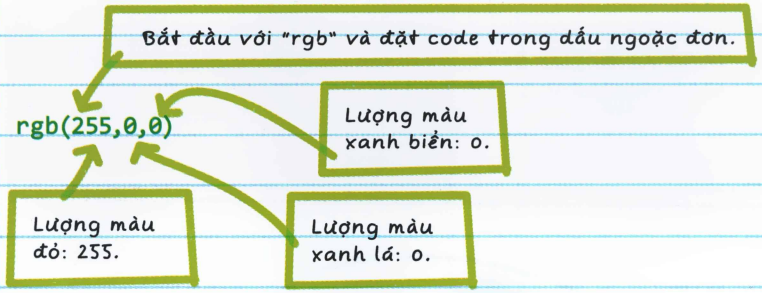
Trong CSS, mọi code màu HEX đều bắt đầu với "#". Ví dụ, giá trị HEX cho màu đỏ thuần là: #FF0000. Có nhiều sắc thái màu đỏ, mỗi sắc thái có giá trị khác nhau. Ví dụ, #700000 có sắc thái tối hơn nhiều so với đỏ thuần.

RGB

RGB là viết tắt của "red, green, blue" (đỏ, xanh lá, xanh biển) và là một dạng code khác để biểu thị những màu sắc tương tự như HEX. Code RGB dùng 3 giá trị (số 0-255) để cho biết có bao nhiêu màu đỏ, xanh lá và xanh biển trong màu sắc đó. Số 0 nghĩa là không có màu, và số 255 là lượng màu cao nhất có thể.

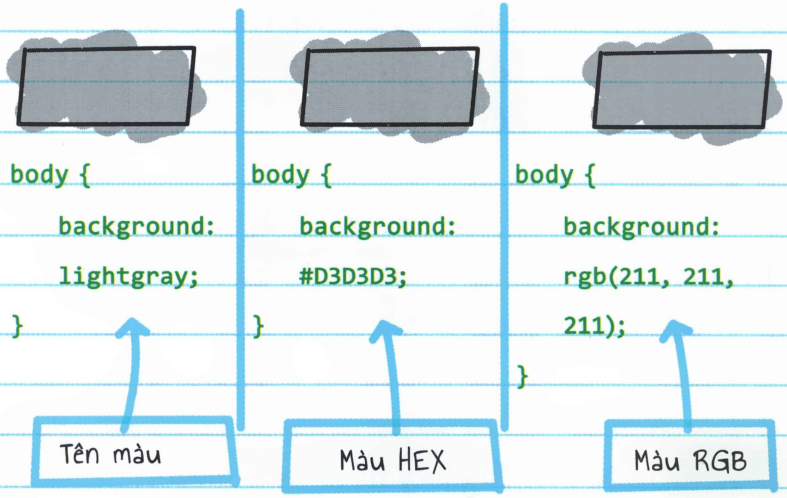
Giá trị đầu tiên của code biểu thị lượng màu đỏ, giá trị thứ hai là màu xanh lá, và giá trị thứ ba biểu thị màu xanh biển (đỏ, xanh lá, xanh biển).

VÍ DỤ, code dưới đây biểu thị màu đỏ thuần.

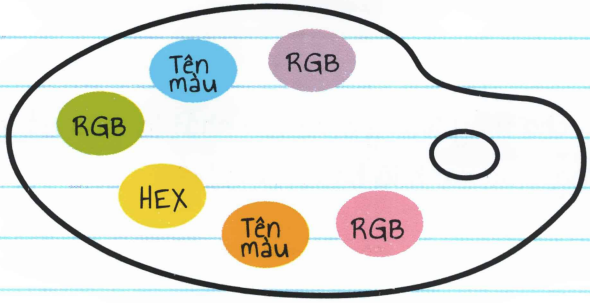


Giá trị cho thấy lượng màu cao nhất là màu đỏ, không có lượng màu xanh lá hoặc xanh biển.

Dưới đây là ví dụ về cách đặt màu nền thành màu xám nhạt bằng cách sử dụng tên màu, HEX, và giá trị RGB (mọi phong cách này sẽ cho cùng một kết quả):



Bạn không nhất thiết chỉ dùng một cách biểu thị màu sắc cho website. Bạn có thể đổi giữa tên màu, màu HEX, và màu RGB trong cùng một file.



THUỘC TÍNH ELEMENT CỤ THỂ

Một số element, ví dụ như bảng hoặc danh sách, có thuộc tính chỉ dùng cho riêng element đó (không giống như thuộc tính màu dùng chung).

VÍ DỤ, bạn có thể thay đổi phong cách gạch đầu dòng của danh sách không theo thứ tự bằng cách sử dụng thuộc tính `list-style-type` đi kèm với một trong những giá trị cụ thể dưới đây:

Đây là mặc định

disc	●	square	■
circle	○	none	(không có gạch đầu dòng, chỉ có văn bản)

Nếu bạn muốn hiển thị danh sách không có thứ tự với square thay vì disc, bạn có thể chèn code CSS vào style sheet chọn element <ul> và dùng mô tả dưới đây:

```
ul {
        list-style-type: square;
}
```

Hầu hết lập trình viên không nhớ hết tất cả thuộc tính và giá trị liên quan đối mọi element có trong HTML. Bạn có thể tìm thuộc tính CSS và giá trị trên internet.

BÌNH LUẬN

Máy tính bỏ qua bình luận trong CSS.

Bình luận giúp phong cách của bạn được hệ thống hơn. Vậy nên khi thư mục CSS dài hơn, bạn vẫn có thể dễ dàng tìm được những đoạn code khác nhau.

Trong CSS, bình luận bắt đầu với `/*` và kết thúc với `*/`.

VÍ DỤ. bạn có thể thêm bình luận vào file CSS dán nhãn "Heading styles":

```
/* Heading styles */
h1 {
        color: tomato;
}

h2 {
        color: #D3D3D3;
}
```

bình luận

KIỂM TRA KIẾN THỨC

1. CSS là viết tắt của từ gì?

2. Bạn có thể dùng CSS với HTML code bằng hai cách nào?

3. CSS dùng để làm gì?

4. Bình luận trong CSS bắt đầu và kết thúc với _____.

5. Bạn viết code CSS nào để khiến toàn bộ nền website biến thành màu tím?

6. Code dưới đây có lỗi. Bạn cần thêm gì để sửa nó?

```
h1
    color: gray;
}
```

7. Gọi tên thuộc tính và giá trị trong declaration dưới đây.

```
color: orange;
```

_____ _____

8. Selector có nhiệm vụ gì trong một phong cách CSS?

9. Bạn cần thuộc tính gì để đổi màu dòng trong element <hr> ?

10. Bạn cần thuộc tính gì để đổi màu văn bản trong element tiêu đề <h1>?

ĐÁP ÁN

1. Cascading style sheets

2. CSS có thể được đặt trực tiếp vào file HTML, hoặc bạn có thể tạo file CSS riêng biệt rồi liên kết nó với file HTML.

3. Thay đổi về phong cách như màu sắc, kích cỡ, phông chữ, và các thiết kế khác cho website.

4. /* và */

5.
```
body {
    background: violet;
}
```

6. Cần thêm vào mở curly bracket.
```
h1 {
    color: gray;
}
```

7. color: orange;

Thuộc tính — Giá trị

8. Selector dùng để chỉ định element áp dụng phong cách. Ví dụ, phong cách có selector "p" sẽ áp dụng cho tất cả element <p>.

9. Màu viền

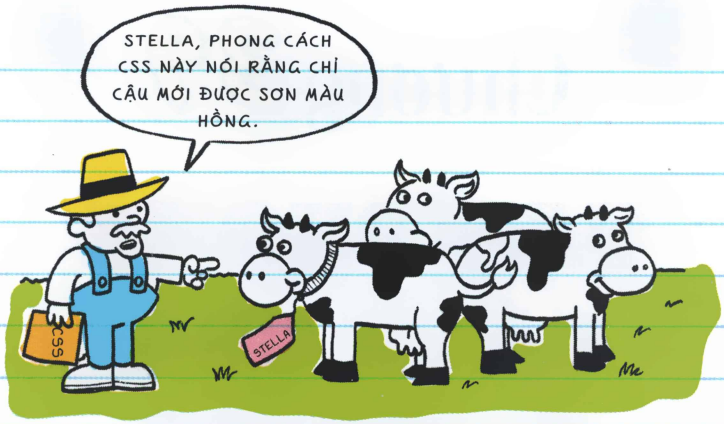
10. Màu sắc

Chương 39

TẠO PHONG CÁCH CHO ELEMENT RIÊNG BIỆT VỚI CSS

XÁC ĐỊNH ELEMENT

SELECTOR ID (IDETIFIER) giống như tag tên – chúng xác định một element cụ thể và áp dụng phong cách CSS cho nó. Tên id trong thư mục CSS cho biết, "Phong cách này chỉ áp dụng cho element có id này".



Khi dùng thuộc tính id, bạn phải làm việc cả trong CSS và thư mục HTML.

Selector Id trong CSS

Bạn có thể tạo selector id cho phong cách CSS mà áp dụng với một element cụ thể trong file HTML. Thay vì dùng tên element, ví dụ như "p", làm selector cho phong cách mới, gõ "#" và tên id. Tên id có thể là bất kỳ thứ gì bạn muốn, nhưng không được có khoảng cách. Bạn có thể tạo một selector id có tên là "introduction" và gán nó cho phong cách có cỡ phông chữ 18 **PIXELS**, làm như sau:

```
#introduction {
        font-size: 18px;
}
```

px là viết tắt của pixel và là một trong những cách đo cỡ phông chữ. Cỡ phông chữ cơ bản trên website là 16 px, cùng kích cỡ với phông 12 trong Microsoft Word.

Pixel (px) là điểm sáng nhỏ nhất trên màn hình. Nếu bạn nhìn thật gần vào màn hình, bạn có thể thấy những chấm siêu nhỏ. Mỗi chấm là một pixel. Lùi ra xa màn hình và các chấm sẽ hòa thành hình ảnh rõ nét.

Thuộc tính Id trong HTML

Thuộc tính được thêm vào tag để cho máy tính thêm thông tin về thuộc tính. Dùng thuộc tính id để xác định element bạn muốn tạo phong cách bằng code CSS mà bạn thêm vào file CSS. Ví dụ, bạn có thể chèn thuộc tính id có tên là "introduction" để xác định một element <p> trong trang web:

Thêm code này vào tag mở <p>.

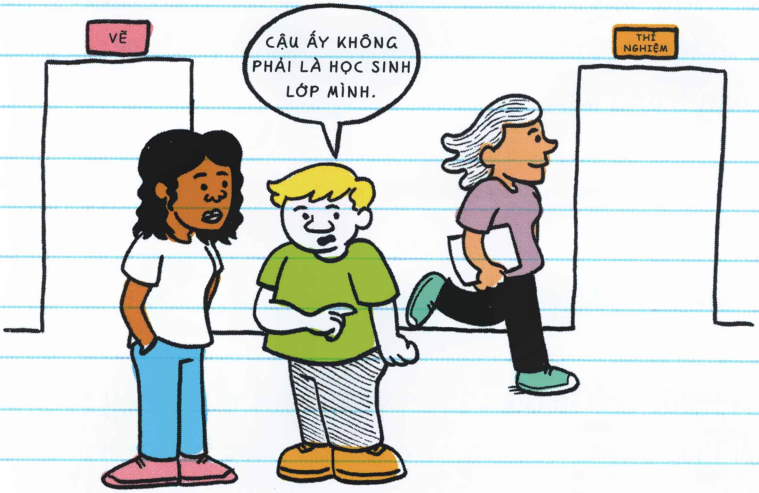
```
<p id="introduction">Oh, hello there! . . .</p>
```

Với thuộc tính id được thêm vào, element <p> này sẽ thay đổi theo phong cách #introduction được tạo ra trong CSS, và cỡ chữ cho văn bản "Oh, hello there!..." sẽ được đặt ở mức 18px khi hiển thị trên website.

SELECTOR CLASS

SELECTOR CLASS dùng để tạo phong cách cho một nhóm element. Selector class xác định nhiều element và áp dụng cùng phong cách cho tất cả chúng.

Khi sử dụng selector class, bạn phải làm việc với cả CSS và thư mục HTML. Ví dụ, bạn có thể dùng selector class để tô màu xanh cho văn bản trong 2 hoặc 3 đoạn trên website.



Selector Class trong CSS

Để tạo một selector class cho một phong cách CSS, gõ một period (.) và một tên class. Tên class có thể là bất kỳ thứ gì bạn muốn, nhưng không được có khoảng cách. Bạn có thể tạo selector class có tên là "overview" và gán cho nó phong cách là màu xanh cho văn bản:

```
.overview {
        color: blue;
}
```

Thuộc tính class trong HTML

Bạn có thể thêm thuộc tính class vào element <p> cụ thể bằng cách gõ **class=** và tên class.

VÍ DỤ, bạn có thể chèn class có tên là "overview" vào hai element <p> xác định trong trang:

Thêm code này vào tag mở <p>.

```
<p class="overview">Oh, hello there! . . .</p>
<p class="overview">On this site we . . .</p>
<p>To learn more about . . .</p>
```

Có hai đoạn sẽ hiển thị văn bản màu xanh trên website.

Các class có thể được thêm vào nhiều element, thậm chí nếu các element không cùng loại. Bạn có thể thêm class vào tiêu đề và nó sẽ biến tiêu đề thành màu xanh:

Thêm thuộc tính class vào giá trị "overview".

```
<h1 class="overview" >Introducing . . .</h1>
```

QUAN TRỌNG: Hãy cẩn thận với việc đánh vần và viết hoa. Nếu bạn đánh vần hoặc viết hoa id hoặc class trong thư mục HTML khác với cách bạn đánh vần hoặc viết hoa nó trong file CSS, phong cách sẽ không được áp dụng đúng cách. Nếu văn bản không hiển thị theo ý bạn, điều đầu tiên bạn cần làm và kiểm tra đánh và vần viết hoa.

CHIỀU RỘNG VÀ CHIỀU CAO

Thuộc tính CSS **chiều rộng** và **chiều cao** cho phép bạn chỉnh lại kích cỡ element như ảnh, danh sách, và đoạn.

Khi mô tả chiều rộng và chiều cao của hình ảnh, px cho biết vật thể rộng hoặc cao bao nhiêu pixel.

Khi bạn chỉnh lại kích cỡ hình ảnh, nếu bạn chỉ chỉnh chiều rộng, chiều cao sẽ tự động chỉnh theo để hợp với chiều rộng.

VÍ DỤ, nếu bạn có hình ảnh rộng 100 px và cao 200 px, và bạn đặt chiều rộng là 200 px, chiều cao sẽ tự động chỉnh sang 400 px vì chiều rộng của ảnh này bằng nửa chiều cao của nó.

Để đặt chiều rộng hình ảnh trên website thành 175 px, thêm code này trong file CSS:

```
img {
        width: 175px;
}
```

Bạn cũng có thể chỉnh cỡ ảnh theo phần trăm so với kích cỡ element khối hiển thị ảnh.

VÍ DỤ, bạn có thể đặt hình ảnh trên website về cỡ 50% (hoặc một nửa) cỡ element khối, sử dụng code sau:

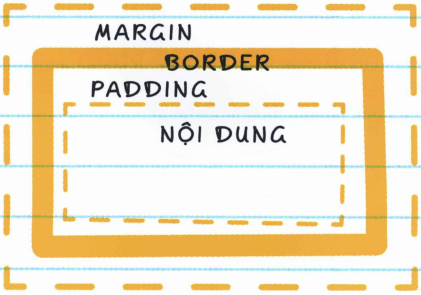
```
img {
    width: 50%;
}
```

BORDERS, MARGINS VÀ PADDING

Bạn có thể dùng CSS để thêm khoảng trống xung quanh hình ảnh.

Borders thêm khung màu xung quanh tất cả element bao gồm hình ảnh, đoạn, tiêu đề và danh sách.

Margins và **padding** thêm khoảng trống xung quanh element. Padding là khoảng trống giữa nội dung của element và border của nó. Margin là khoảng trống giữa border và phần còn lại của website.



Bạn có thể thêm margin vào element <img> bằng cách sử dụng thuộc tính margin trong CSS.

VÍ DỤ, bạn có thể cho hình ảnh một margin 10 pixel bằng cách thêm giá trị thuộc tính "margin" và đặt giá trị của nó là "10px".

Chọn tất cả hình ảnh trong thư mục HTML.

```
img {
        margin: 10px;
}
```

Thêm margin.

Một số thuộc tính, ví dụ như thuộc tính border, đặc biệt vì bạn có thể thêm nhiều hơn một giá trị vào cùng một declaration.

Ví dụ, thêm "6px solid orange" làm giá trị để tạo border dày 6 px, đặc ruột và màu cam:

```
img {
        border: 6px solid orange;
}
```

Chèn dấu chấm phẩy ở cuối mỗi declaration.

Để tạo phong cách bao gồm margin, border, padding và chiều rộng hình ảnh, bạn có thể kết hợp tất cả declaration bằng cách sử dụng một selector img duy nhất, mỗi declaration mới nằm ở một dòng:

```
img {
        margin: 10px;
        border: 6px solid orange;
        padding: 20px;
        width: 50%;
}
```

Phong cách sẽ cho tất cả element <img> một margin 10 px, border đặc, màu cam và rộng 6 px, một padding 20 px, và chiều rộng ảnh bằng 50% chiều rộng cửa sổ trình duyệt.

Declaration cũng có thể được viết là img {**margin**: 10px; **border**: 6px **solid orange**; **padding**: 20px; **width**: 50%;}. Thụt lề và ngắt dòng giúp lập trình viên đọc code dễ dàng hơn.

KIỂM TRA KIẾN THỨC

1. Sự khác biệt giữa selector id và selector class trong CSS?

2. Px là viết tắt của _____.

3. Selector id trong CSS luôn bắt đầu với biểu tượng gì?

4. Trong CSS, selector class có _____ đứng trước tên class.

5. Nếu bạn muốn thay đổi kích cỡ hình ảnh, bạn nên dùng thuộc tính CSS nào?

6. Cái nào tạo khoảng cách giữa nội dung element và border: margin hay padding?

7. Xem bảng ở trang sau để trả lời câu hỏi 7A -D.

   A. Bạn nên thêm code gì để khiến element <p> với nội dung "Good Luck!" hiển thị văn bản màu đỏ cà chua (tomato red)? Bạn nên thêm nó vào đâu và vào file nào?

FILE HTML: "INDEX.HTML"	FILE CSS: "STYLE.CSS"

<!DOCTYPE html>
 	
<html>
<head>
<title>Test Prep</title>
<link rel="stylesheet" type="text/css"
href="style.css">
</head>
<body>
 This is a test
 Things to remember:
 	
 - Don't stress out
 - Keep Calm and Test On
 - Do your best!
 	
 Good Luck!
</body>
</html>
	#redText {
	color: tomato;
	}
	li {
	font-size: 24px;
	}
	.border {
	border: 2px solid gray;
	}

B. Bạn nên dùng code nào để thêm border 2 px đặc ruột màu xám vào tiêu đề <h1> và element danh sách thứ hai? Bạn thêm nó vào đâu và vào file nào?

C. Bạn nên điều chỉnh code nào để thay đổi kích cỡ phông chữ danh sách từ 24 px sang 16 px? Bạn điều chỉnh file nào?

D. Bạn nên thêm code nào để khiến tất cả phần tử được được liệt kê biến thành màu xám? Bạn thêm code vào đâu?

ĐÁP ÁN

ĐÁP ÁN

1. Selector id dùng để chọn element duy nhất với id đi kèm, trong khi selector class dùng để chọn nhiều element trong cùng một class.

2. Pixel

3. #

4. Period

5. Chiều rộng và chiều cao

6. Padding

7. **A.** Bạn nên thêm id="redText" vào element <p> cuối cùng trong file HTML:

```
<p id="redText">Good Luck!</p>
```

B. Bạn nên thêm class="border" vào element <h1> và element <li> thứ hai, và vào file HTML:

```
<body>
<h1 class="border">This is a test</h1>
<p>Things to remember:</p>
<ul>
    <li>Don't stress out</li>
    <li class="border">Keep Calm and Test On</li>
    <li>Do your best!</li>
</ul>
<p>Good Luck!</p>
</body>
```

C. Bạn nên điều chỉnh declaration "font-size: 24px;" trong phong cách có selector "li", chuyển sang "font-size: 16px;", điều chỉnh trong file CSS như sau:

```
li {
        font-size: 16px;
}
```

D. Bạn nên thêm declaration mới vào phong cách có selector "li" trong file CSS:

```
li {
        font-size: 16px;
        color: gray;
}
```

Index

KÝ HIỆU

= toán tử gán, 151

+ toán tử phép cộng, 342–343, 357

- toán tử phép trừ, 357

* toán tử phép nhân, 357

\ toán tử phép chia, 357

== toán tử bằng, 168, 170, 376–377, 403

> toán tử lớn hơn, 168, 376–377, 403

>= toán tử lớn hơn hoặc bằng, 168, 376–377, 403

dấu thăng, 50, 316, 530, 540

< toán tử nhỏ hơn, 168, 376–377, 403

<= toán tử nhỏ hơn hoặc bằng, 168, 376–377, 403

!+ không bằng, 168, 376–377, 403

% phép toán Modulo, 357

** số mũ, 357

// phép chia làm tròn xuống, 357

CHỈ MỤC

404 Not Found code (status code), 453

A
An ninh mạng, 463–479
Ảnh hưởng của máy móc, mô tả, 9
Ảnh thu nhỏ, 208–209
Apple II, 15
Arduinos, 72
Âm thanh, nhân vật, 212

B
Babbage, Charles, 115
Bàn tính, 13
Bảng chỉnh sửa âm thanh, 212
Bảng chỉnh sửa hình ảnh, 211
Berners-Lee, Tim, 448
Biến, 148–162
  Giới thiệu, 148–149
  Gán và đặt tên, 150–153
  Cài sẵn, 250–251
  Tính toán với số, 351–352
  Thay đổi giá trị của chuỗi, 337
  Bộ đếm, 381–386
  Khai báo, 150
  Định dạng tên, 328–329
  Đặt tên, 325–327
  Đặt tên với chuỗi, 334
  Số đóng vai trò, 323, 351–364
  Tham số, 194–196
  Trong Python, 323–332, 351–364
  Trong Scratch, 247–253, 260–263
  Các dạng thông tin được lưu trữ, 154–157
  Xem thêm câu lệnh điều kiện
Biến cài sẵn, 250–251
Biến đếm, 381–386
Biến thành bình luận, 82
Biểu đồ, 61, 63
Biểu đồ cột, 62
Biểu đồ dây, 62
Biểu đồ tròn, 63
Biểu thức đại số, 155
Biểu thức/Giá trị Boolean
  Giới thiệu, 156–157, 247
  Toán tử so sánh, 168–171
  Câu lệnh điều kiện phức hợp, 407–411
  Câu lệnh điều kiện, 403
  Trong Python, 323–324, 374–377
Biểu thức, toán học, 356–357
Biểu tượng
  #, 316
  =, 151
  >>>, 311
  Dấu gạch chéo, 338
  Biểu thức Boolean, 403
  Toán tử so sánh, 168–170, 376
  Cho bình luận trong CSS, 534
  Dấu ngoặc kép, 316
  Cho bình luận trong HTML, 490
  Cho dánh sách bên trng, 372–373
  Cho biểu thức toán học, 357
  Ngoặc đơn, 227, 358–359
  Ngoặc kép, 316, 333–334, 335, 338–339
  Ngoặc kép đơn, 316, 336
  Gạch chân (_), 152
Biểu tượng gạch chân, 152
Bình luận
  Giới thiệu, 81–83
  Thuộc tính, 101–102
  Trong CSS, 534
  Trong code HTML, 490
  Trong Python, 315–316
  Trong Scratch, 205

Bit, 131
BOMBE, 469
Boolean, George, 157
Bộ nhớ, mô tả, 18
Bộ phiên dịch, 125
Bộ xử lý đồ họa (GPU), 19
Bộ xử lý sự kiện, 188
Bộ xử lý thông tin, 41
Bộ xử lý trung tâm (CPU), 18-19
Byte, 131-132

C
C++, dùng cho, 122-123
Các dạng số, 352-355
Cách tiếp cận có hệ thống để khắc phục sự cố, 31-32
Cài đặt "truy cập", 30
Cải tiến giải pháp, 71
Camel case (mixed case), 329
  Giới thiệu, 481-482, 523-524
  Selector class, 543
  Thuộc tính màu, 528-532
  Màu sắc, 50
  Bình luận, 534
  Thuộc tính element cụ thể, 533-534
  Selector id, 540-541
  Kết nối file HTML, 526-528
  Tạo file, 524-528
  Tạo phong cách cho element riêng biệt, 539-553
Cascading style sheets (CSS), 523-538
  Giới thiệu, 481-482, 523-524
  Selector class, 543
  Thuộc tính màu, 528-532
  Màu sắc, 50
  Bình luận, 534
  Thuộc tính element cụ thể, 533-534
  Selector id, 540-541
  Kết nối file HTML, 526-528
  Tạo file, 524-528
  Tạo phong cách cho element riêng biệt, 539-553
Câu hỏi đa lựa chọn, 59-60
Câu hỏi mở, 59
Câu hỏi mức độ, 60
Câu lệnh AND, 171-174, 288, 407, 409-410
Câu lệnh điều kiện, 163-178
  Giới thiệu, 163-167
  Cơ bản, 284
  Toán tử, 168-170
  Phức hợp, 171-174, 285-288, 407-411
  Câu lệnh elif, 405-406
  Câu lệnh else if , 165-166, 174
  Câu lệnh else , 165, 404-405
  Lồng ghép, 174-175, 285-288, 411-414
  Thủ tục, 302
  Trong Python, 403-422
  Trong Scratch, 283-288
  Vòng lặp điều kiện, 181, 393-397
Câu lệnh điều kiện cơ bản, 284
Câu lệnh điều kiện lồng ghép, 174-175, 285-288, 411-414
Câu lệnh điều kiện phức hợp, 171-174, 285-288, 407-411
câu lệnh elif, 405-406
câu lệnh else, 165, 404-405
câu lệnh else if, 165-166, 174
Câu lệnh if . . . then, 163-167
Câu lệnh if then, 283
Câu lệnh if then else, 283
Câu lệnh lặp, 179-186
Câu lệnh NOT, 173-174, 288, 407, 411
Câu lệnh OR, 172-173, 288, 407, 409-410
CDs (đĩa nén), 13, 18
Chạy khối, 223-224
Chế độ bitmap, 211
Chế độ vector, 211
Chiều rộng element, 544-546

Chuỗi
Mô tả, 154
Định dạng, 338–340
Trong Python, 323–324, 333–350
Chuyển đổi số, 355
Chương trình
Thuật toán, 112–114
So sánh với ứng dụng, 17
Định nghĩa, 6
Là một phần của hệ thống máy tính, 6
Xem thêm ứng dụng/phần mềm ứng dụng (app); phần mềm nhắc nhở,
Chương trình con, 280–283
Chương trình trí tuệ nhân tạo (AI), 42, 140–141
Cipher, 467–469
Cipher Caesar, 468–469
Code
Định nghĩa, 46
Cách máy tính đọc, 123–132
HTML, 481–483, 484, 488–490
Máy, 124–125
Trong Python, 314–315
Tái sử dụng, 191–196
Code giả, 133, 182
Code HTML, 481–483, 484, 488–490
Code máy, 124–125
Colossus Mark 1, 14
Cộng, chuỗi, 342–345
Cộng chuỗi, 342–345
cộng đồng, tìm sự giúp đỡ, 101–102
Công nghệ máy tính, công dụng, 3
Cửa sổ chỉnh sửa, 311, 313
Cửa sổ shell, 311–312
CSS. See cascading style sheets (CSS)

D
Danh sách
Giới thiệu, 365
Thêm, 371
Thay đổi thuộc tính, 533
Hàm, 367–372
Văn bản HTML, 503–505
Bên trong, 372–373
Element danh sách, 503–505
Element danh sách phần tử, 503–504
Vòng lặp lần, 387–388
Toán tử, 371
Có thứ tự, 505
Dấu câu, 365–366
Trong Python, 323–324, 365–374
Thay thế giá trị, 368
Trong Scratch, 251–253
Sắp xếp, 370–371
Âm thanh, 212
Nhân vật, 208–209
Biểu tượng, 365
Biểu tượng cho danh sách bên trong, 372–373
Biến, 157, 247
Bên trong danh sách, 372–374
Danh sách, 157, 247
Danh sách âm thanh, 212
Danh sách bên trong, 372–373
Danh sách có thứ tự, 505
Danh sách nhân vật, 208–209
Danh sách trang phục, 210
Dấu bằng, 151
Dấu câu
Lưu ý, 315
Cho danh sách, 365–366
Xem thêm biểu tượng, dấu câu độc lập
Dấu ngoặc kép
Cho bình luận, 316
Trong hàm print, 335
Trong chuỗi, 333–334, 338–339
Dấu ngoặc kép, 316
Dấu ngoặc kép đơn, 316, 336
Di chuyển chú rùa, 424–427
Dịch vụ hội nghị truyền hình, 458

Diễn giải, 103
Dòng mới, xuất, 340
Dữ liệu
  Giới thiệu, 43–46
  Định nghĩa, 8
  Mô tả, 8
  Mã hóa, 47–51
  Dòng chảy thông tin, 19–20
  Các dạng, 323–324
  Sử dụng, 61–63
Dữ liệu đầu ra, 44
Dữ liệu đầu vào, 44, 46

Đ
Đảo ngược từ, 153
Đặt tên biến trong Python, 325–327
Đầu não, 452
Đầu ra
  Mô tả, 20
  Dòng chảy thông tin, 20
  Trong Python, 312
  Giá trị trả về, 434–437
Đầu vào
  Mô tả, 17
  Dòng chảy thông tin, 19–20
  Người dùng, 250–251, 344–345, 432–437
Động cơ phân tích, 115–116
Đếm ngược, 386
Địa chỉ IP, 449–450, 455
Điện thoại thông minh, 13, 15
Điều kiện (Khối C), 274
Điểm mạnh, tập trung, 98
Định dạng chuỗi, 338–340
Định dạng element, 498–499
Định dạng HTML, 488–490
Định dạng tên biến, 328–329
Định dạng vòng lặp lồn, 381
Định nghĩa hàm, 430
Định vị tài nguyên thống nhất (URL), 454
Đọc màn hình, 31
Đồ họa chú rùa, 423–429
Đường kẻ ngang, 499–501

E
Element
  Thông tin, 484–486
  Mỏ neo, 513–515
  <body>, 486–487
  In cascading style sheets (CSS), 533–534, 539–553
  Rỗng, 502
  Định dạng, 498–499
  <head>, 486–487
  Tiêu đề, 497–498
  Chiều cao, 544–546
  Văn bản HTML, 497–509
  <html>, 486
  Xác định, 539–541
  Hình ảnh, 515–517
  Liên kết, 511–521
  Danh sách, 503–505
  Danh sách phần tử, 503–504
  Đoạn, 498–499
  Danh sách không có thứ tự, 503–504
  Uebsite, 486–487
  Của websites, 486–487
  Chiều rộng, 544–546
Element <body> , 486–487
Element danh sách, 503–505
Element danh sách không có thứ tự, 503–504
Element danh sách phần tử, 503–504
Element đoạn, 498–499
Element <head> , 486–487
Element hình ảnh, 515–517
Element <html>, 486
Element liên kết, 511–521
Element rỗng, 502
Element tiêu đề, 497–498

Element văn bản HTML, 497-509
ENIAC, 14-15
Enigma, 469
Ethernet, 445-446

F
Facebook, ngôn ngữ lập trình, 122
File HTML, 452, 526-528
File, lưu, 313
File README, 83-84, 101-102

G
Gán giá trị, 324-325
Ghi chú, 81-88
Ghi nhận, 101-103
Giá trị
  Gán, 324-325
  Trong biểu thức, 356
  Số, 155-156
  Thay thế giá trị trong danh sách, 368
  Trả về, 196, 434-437
  Trả về, 196
  Biến, 148-151
Giá trị chuỗi, 263
giá trị hàng, mã nhị phân, 127-128
Giá trị số, 155-156
giá trị tham số, 314
Giá trị trả về, 196, 434-437
Giải mã hoặc giải máy cipher, 468
Giải mã thông tin, 47
Giao diện dòng lệnh (CLI), 30
Giao diện đồ họa người dùng (GUI), 30
Giao diện người dùng (UI), 29-31
Giao thức điều khiển truyền vận (TCP), 456-457
Giao thức IP-internet, 449
Giao thức mạng, 445-446
Giao thức truyền siêu văn bản bảo mật (HTTPS), 453
Giao thức truyền siêu văn bản (HTTP), 448, 453
Gigabyte, 132
Gọi hàm, 315, 430-431
Gọi thủ tục, 193-194, 300

H
Hacker, 469-470
Hacker mũ đen, 470
hacker mũ trắng, 469-470
Hàm
  Append, 369, 372
  Thân hàm, 431
  Gọi hàm, 315, 430-431
  Viết hoa, 347
  Chuyển đổi, 433-435
  Định nghĩa hàm, 430
  Định nghĩa, 192
  Float, 355
  Input, 344-345
  Insert, 369-370, 372
  Int, 354
  Len, 371, 372
  Danh sách, 367-372
  Lower, 346-347
  Có sẵn, 103
  Xuất, 314-315, 334-337, 367-368
  Thủ tục, 423
  Trong Python, 311, 314-315, 423-442
  Range, 382-388
  Remove, 370, 372
  Reverse, 370-371, 372
  Sort, 370, 372
  Chuỗi, 345-347
  Swapcase, 347
  Rùa vẽ, 428-429
  Đồ họa chú rùa, 423-429
  Upper, 347
Hàm append, 369, 372
Hàm cài sẵn, 103
Hàm chuyển đổi, 433-435

Hàm float, 355
Hàm input, 344–345
Hàm int, 354
Hàm insert , 369–370, 372
Hàm len , 371, 372
Hàm lower, 346–347
Hàm print, 314–315, 334–337, 367–368
Hàm range, 382–388
Hàm remove, 370, 372
Hàm reverse, 370–371, 372
Hàm sort , 370, 372
Hàm string, 345–347
Hàm swapcase, 347
Hàm upper, 347
Hàm viết hoa, 347
Hành động bên trong, 188
Hành động từ bên ngoài, 188
Hạt mưa, 303–305
Hệ cơ số 2, 126
Hệ cơ số 10, 126
Hệ điều hành, 22
Hệ điều hành Windows, 22
Hệ thống lưới , 214–215
Hệ thống mã hóa dữ liệu, 47–48
Hệ thống máy tính
  Định nghĩa, 2
  Mô tả, 6
Hệ thống máy tính, thiết kế, 68–72
Hiệu quả, định nghĩa, 57
Hiệu quả, thuật toán, 111–112
Hình ảnh
  Viền, margins, and padding for, 546–548
  Chỉnh kích cỡ, 544–546
Hình dung, 141, 143
Hình nền, 528
Hoa tiêu, 100
Hoạt hóa một chú mèo, 225–232
Hollerith Tabulating Machine, 61
Hopper, Grace, 125
hợp tác, 97–108

I

IDLE (Chương trình môi trường phát triển tích hợp), 311
Index, 366
Internet, 444–462
  Giới thiệu, 447–449
  Mô tả, 8–9
  Gửi thông tin, 449–450
  Lướt web, 451–458
  So sánh với web, 451
iOS, 22

J

JavaScript
  Sự tương tác, 482
  Công dụng, 122
Julius Caesar, 468

K

Key, 467
Kế hoạch, một phần của quá trình thiết kế kỹ thuật, 70
Khả năng sử dụng, 90
Khả năng tiếp cận, 90
Khai báo , 524–526
Khai báo biến, 150
Khai báo thủ tục, 191–193
Khảo sát, 58–60
Khắc phục sự cố
  Định nghĩa, 31
  Chiến thuật, 32–33
  Xem thêm gỡ lỗi chương trình
Khoa học máy tính
  Khái niệm về các lĩnh vực, 6–9
  Mô tả, 2–12
Khối
  Giới thiệu, 200–207
  Hoạt hóa chú mèo, 225–232
  Boolean, 254, 256–259, 279, 284, 287–288
  Phát, 276–277
  Khối C , 274, 278–283
  Khối mũ, 274

Bản sao, 293
Điều khiển, 273-274, 291-293
Sự kiện, 224, 275-277, 289-290
Bản mở rộng, 217-218
Lướt đi, 230-232
Đi tới, 231
Mũ, 224-225
Hình ảnh, 237-239
Tự tạo, 299-302
Toán, 254-255
Chuyển động, 215, 234-236
Khối của tôi, 306
Lồng ghép, 258, 278
Toán tử, 254-259, 263-265
Bút vẽ, 229, 242-243
Thủ tục, 299-302
Khối xếp chồng gọi thủ tục, 300
Lặp lại, 273
Báo cáo, 247-251, 254, 263-265
Chạy, 223-224
Cảm biến, 266-268
Âm thanh, 240-241
Xếp chồng, 224
Quay , 228-229
Các dạng, 224-225
Biến, 247-253, 260-263
Khối âm thanh, 240-241
Khối bản sao, 293
Khối báo cáo, 247-251, 254, 263-265
Khối Boolean, 254, 256-259, 279, 284, 287-288
Khối bút vẽ, 229, 242-243
Khối C, 274, 278-283
Khối cảm ứng, 266-268
Khối chuyển động, 215, 234-236
Khối chuyển tới, 231
Khối của tôi, 306
Khối điều khiển, 273-274, 291-293
Khối hình ảnh, 237-239
Khối lặp lại, 273
Khối lượng công việc, bình đẳng, 99
Khối lướt đi, 230-232
Khối mũ, 224-225
Khối nắp, 274
Khối nối, 227
Khối phát tin, 276-277
Khối quay, 228-229
Khối sự kiện, 224, 275-277, 289-290
Khối thủ tục, 299-302
Khối toán, 254-255
Khu vực ba lô, 206-207
Khu vực kịch bản, 204
Khối xếp chồng, 224
Khối xếp chồng gọi thủ tục, 300
Kịch bản, 223-246
Giới thiệu, 223-224
Hoạt hóa chú mèo, 225-232
Chiến thuật sửa lỗi, 233
Mô tả, 203-204
Tái sử dụng, 299-308
Lưu trữ, 206-207
Bước đi, 226-230
Xem thêm khối
Kịch bản đi, 226-230
Kiểm soát truy cập, 465
Kiểm tra mật khẩu, 394-395, 397
Kilobyte, 132
Kính viễn vọng không gian Hubble, 132
Ký hiệu màu hệ hex, 48, 49-50, 51, 530, 532
Ký hiệu màu RGB, 48, 49, 51, 131, 531-532
Kỷ Nguyên thông tin, 40-41
Kỹ thuật
Định nghĩa, 68
Hàng ngày, 72
Kỹ thuật xã hội, 474-475
Kỳ vọng, rõ ràng, 98

L
Lặp lại vòng lặp, 278-279, 281-283
Lặp lại vòng lặp đến khi, 278-279
Lập trình
Phòng ngừa, 73-76
Mô tả, 6-7
Thời kỳ đầu, 115-116

Theo cặp, 100
Xem thêm Python; Scratch
Lập trình đôi, 100
Lập trình phòng ngừa, 73-76
lập trình viên điện thoại, 21
Lập trình viên trò chơi, 21
Lập trình viên web, 21
Lidar, 42
Lovelace, Ada, 115-117
Lồng ghép khối, 258, 278
Lưu kịch bản, 206-207
Lưu trữ thông tin, 40-55

M
Mã hóa dữ liệu, 47-51
Mã hóa màu sắc, 48-51
Mã morse , 47
Mã nhị phân
Giới thiệu, 123-125
Mô tả, 48
Sử dụng để biểu diễn chữ cái, 130
Sử dụng để biểu diễn số, 126-129
Mã Tiêu Chuẩn Của Mỹ Để Trao Đổi Thông Tin (ASCII), 130
Mã trạng thái (code 404 Not Found), 453
MAC (Kiểm soát truy cập phương tiện), 449
Mạch điện, 123
Makey Makeys, 72
Malware, 472
Màn hình cảm ứng, 20
Mảng, 157, 247
Mạng cục bộ (LANs), 445
Mạng diện rộng (WANs), 445
Mạng lưới máy tính, 444-447
Mạng lưới, mô tả, 8-9
Mạng Toàn Cầu
Nguồn gốc, 448-449
Lướt, 451-458
Sử dụng internet để lướt, 451-458
Xem thêm trang web; website
Margins, 546-548
Màu sắc
Mã hóa, 48-51
Thể hiện, 51
Màu viền, 529
Máy chủ, 452
Máy chủ hệ thống tên miền (DNS), 455-456
Máy quét, 48-49
Máy tính
Mô tả, 13-27
Tương tác người - máy tính, 29-37
Sự ảnh hưởng, 9
Là công cụ xử lý thông tin, 41
Tương tác, 29-37
Các phần của, 16-23
Đọc code, 123-132
Qua các thời kỳ, 14-15
Máy tính, định nghĩa, 2
Mật khẩu, tấn công mạng, 470-471
Mật mã học, 466-469
Megabyte, 132
Menu điều hướng, 513
Menu khối, 202-203
Mèo, hoạt hóa, 225-232
Mèo Scratch, 225-232
Micro:bits, 72
Mixed case (camel case), 329
Modules, 423-424
Mốc thời gian, hợp lý, 98

N
Nexus, 448
Ngăn nắp, trong Python, 315-317
Ngắt dòng, 499-501
ngoặc đơn, 227, 358-359
Ngôn ngữ
Back-end, 122-123
Front-end, 122
JavaScript, 122, 482

Tự nhiên, 140
PHP, 122, 482
Xem thêm ngôn ngữ đánh dấu siêu văn bản (HTML); ngôn ngữ lập trình
Ngôn ngữ back-end , 122-123
Ngôn ngữ đánh dấu siêu văn bản (HTML)
Trang web cơ bản, 481-483
Selector class, 543-544
Định nghĩa, 448, 481
Mô tả, 452
Phần mở rộng, 482
Định dạng, 488-490
Thuộc tính id, 541
Trình duyệt web, 484
Ngôn ngữ front-end, 122
Ngôn ngữ lập trình, 121-138
Giới thiệu, 121
Tạo thuật toán, 133-135
Mô tả, 113
Cách máy tính đọc code, 123-132
Sử dụng, 122-123
Ngôn ngữ tự nhiên, 140
Ngựa Trojan, 472
Người dùng khiếm thị, 30-31
Người dùng khiếm thính, 30-31
Người sáng tạo, 4
Người theo chủ nghĩa hack, 470
Người tiêu thụ, 4
nhà cung cấp dịch vụ internet, 450
Nhà khoa học máy tính, thành quả, 4
Nhắc nhở, trong Python, 311
Nhận biết khuôn mẫu, 141, 142
Nhân vật, 200, 208-216, 225-232
Nhóm, làm việc cùng, 97-99
Nội dung
Phù hợp với lứa tuổi, 90
Của website, 484
Nội dung phù hợp với lứa tuổi, 90
Ô
Ô tô tự lái, 42
Ổ USB, 18
P
Packets, 449, 456
Padding, 546-548
Pascal case, 329
PEMDAS (Please Excuse My Dear Aunt Sally), 358
Phản hồi của máy tính, 58
Phản hồi, tích hợp, 89-96
Phần cứng
Tìm hiểu sâu hơn, 17-20
Mô tả, 16
Là một phần hệ thống máy tính, 23
Phần mềm
Tìm hiểu sâu hơn, 21-23
Mô tả, 16
Là một phần của hệ thống máy tính, 23
Hệ thống, 22
Chiến thuật khắc phục sự cố, 33
Xem thêm ứng dụng/ phần mềm ứng dụng (app) ; ứng dụng
Phần mềm hệ thống, 22
Phần mở rộng
.html, 482-483
.py, 313
Phần mở rộng, 217
Phần mở rộng .html, 482-483
Phần mở rộng .py , 313
Phân nhỏ, 141-142
Phân tích, mô tả , 8
Phép tính, thứ tự, 358-359
Phiên dịch, định nghĩa, 124
Phong cách, 524-526
Phỏng vấn, 58
Phông nền, 216, 276
PHP

Tính toán
Với biến số, 351-352
Toán tử, 254-259
Tọa độ x và y, 214-215
Toán tử
Trong biểu thức, 356
Danh sách, 371
Logic, 171-174, 258, 407-411
Trong Scratch, 254-259, 263-265
Chuỗi, 341-347
Toán tử chuỗi, 341-347
Toán tử gán, 151
Toán tử logic, 171-174, 258, 407-411
Toán tử so sánh, 168-170, 256-259, 376-377
Trang phục, 209-210, 237-239
Trang web
Tạo web cơ bản, 492-496
Ngôn ngữ lập trình, 121
Trích dẫn, 103
Trình duyệt web, 483, 484
Trò chơi 3 chữ cái, 411-414
Truth tables, 410-411
Truy cập, thẩm quyền, 463-466
Trường hợp sử dụng, 74-75
Trường hợp thử nghiệm, 74, 75-76
Turing, Alan, 469
Tư duy máy tính, 139-146
Từ khóa trong Python, 326
Tự tạo khối của riêng bạn, 299-302
Tương tác người - máy, 29-37
Twitter, ngôn ngữ lập trình, 122

U
Ứng dụng bản đồ, 41
Ứng dụng điện thoại, ngôn ngữ lập trình, 121
Ứng dụng mạng xã hội, 41-42, 474-475
Ứng dụng, phần mềm ứng dụng (app), 17, 21. Tương tự chương trình; phần mềm

V
Vai trò, xác định rõ, 99
Văn bản đơn thuần, cho file README, 83
Văn bản hiển thị, nhiều dòng, 336
Vấn đề di chuyển của người bán hàng, 112
Vấn đề, xác định, 69-70
Viền, 546-548
viết code màu, trong Python, 317
viết code, thời kỳ đầu, 115-116
Viết hoa, 315
viết_thường_and_gạch_chân, 328
Vi-rút, 464, 473
Vòng lặp
Cơ bản, 278-280
Vòng lặp vô tận, 278, 281
Vô tận, 395-397
Vòng lặp lồng, 180, 381-392
Lồng ghép, 182-183, 280-283, 397
Trong Python, 381-401
Lặp lại vòng lặp, 278-279, 281-283
Lặp lại vòng lặp đến khi, 278-279
Trong Scratch, 274, 278-283
Vòng lặp điều kiện, 181, 393-397
Vòng lặp cơ bản, 278-280
vòng lặp điều kiện, 181, 393-397
Vòng lặp lồng, 180, 381-392
Vòng lặp lồng ghép, 182-183, 280-283, 397
Vòng lặp vô tận, 278, 281
Vòng lặp vô tận, 395-397

W
Website
Truy cập, 451-453
Tạo khối, 481-482
Nội dung, 484
Định nghĩa, 121
Element, 486-487
Tìm kiếm, 454-456

Ngôn ngữ lập trình,122
Công cụ để viết, 482-483
Xem, 483
Web. Xem Mạng Toàn Cầu
Wi-fi, 446
Wireless, 445-446
Worms, 473

X
Xác định element, 539-541
Xác thực, 464-465
Xác thực hai yếu tố, 465
Xây dựng các bước, là một phần
của quá trình thiết kế kỹ thuật, 71
Xp (điểm kinh nghiệm), 154
Xuất biểu thức toán học, 359

Y
YouTube, ngôn ngữ lập trình, 122

SỔ TAY KHOA HỌC MÁY TÍNH VÀ MÃ HÓA

NHÀ XUẤT BẢN LAO ĐỘNG
175 Giảng Võ - Ba Đình - Hà nội
Điện thoại: 024 38515380 Fax: 024 38515381
Email: info@nxblaodong.com.vn
Website: www.nxblaodong.com.vn
Chi nhánh phía Nam:
Số 85 Cách Mạng Tháng Tám, Quận 1, Thành phố Hồ Chí Minh
Điện thoại: 02838390970 Fax: 02839257205

Chịu trách nhiệm xuất bản:
Giám đốc - Tổng biên tập
MAI THỊ THANH HẰNG

Biên tập NXB: Lê Thị Hằng
Trình bày: Hoàng Hoài Hương
Sửa bản in: Á Châu Book
Bìa: Á Châu Book

LIÊN KẾT XUẤT BẢN VÀ PHÁT HÀNH:
Công ty Cổ Phần Đầu Tư và Phát Triển Giáo Dục Quốc Tế Á Châu
Số 8 Lô 2, Dự án nhà ở Phùng Khoang, Phường Trung Văn,
Quận Nam Từ Liêm, Hà Nội
Điện thoại: 024 8582 5555 Hotline: 0916 640 166
Website: Http://achaubooks.vn Hoặc http://hocgioitoan.com.vn
Email: info@achaubooks.vn
Facebook: fb.com/hocgioitoan.com.vn

In 5.000 cuốn, khổ 14x20.5 cm, tại Công ty cổ phần In Sao Việt,
số 9/40 Ngụy Như Kon Tum, Nhân Chính, Thanh Xuân, TP Hà Nội.
Số xác nhận ĐKXB: 3775-2021/CXBIPH/14-224/LĐ.
Số QĐXB của NXB: 1684/QĐ-NXBLĐ.
Mã số sách tiêu chuẩn quốc tế (ISBN): 978-604-343-279-4.
In xong và nộp lưu chiểu năm 2022.