

# **Bài 1**

# **Nhập môn lập trình căn bản**

Môn học: PF-JAVA

# Mục tiêu

- Trình bày được nội dung, yêu cầu, lịch trình và kết quả của môn học PF
- Trình bày được ý nghĩa của lập trình và ngôn ngữ lập trình
- Trình bày được khái niệm thuật toán
- Sử dụng được pseudo-code và Flowchart để mô tả thuật toán
- Giải quyết được các bài toán cơ bản trong lập trình sử dụng code.org

**Nội dung, yêu cầu, lịch trình và kết quả của môn học PF**

# Giới thiệu về môn học PF (1)

- **Mục đích:** *Khoá học trang bị cho học viên những kiến thức và kỹ năng nền tảng về lập trình và tư duy giải quyết vấn đề. Kết thúc khoá học này, học viên phát triển được các ứng dụng phần mềm cơ bản theo mô hình Lập trình Hướng Đối tượng và ứng dụng được các cấu trúc dữ liệu và giải thuật vào trong giải pháp của mình.*
- **Thời gian:** 30 bài
- **Đánh giá:**
  - Thi thực hành và lý thuyết cuối module
  - Bảng đánh giá kỹ năng theo chuẩn đầu ra
- **Yêu cầu:**
  - Phần mềm IntelliJ

# Giới thiệu về môn học PF (2)

- **Tài liệu học tập:**

- CodeGymX: [PF-JAVA] Programming Fundamentals
- Source code mẫu trên kênh Github của CodeGym gồm các source code ứng với các bài tập ứng với từng module PF.
- Ứng dụng CodeGym Bob gồm các bài luyện tập, bài học, bài kiểm tra.

- **Tài liệu tham khảo:**

- Các tài liệu tham chiếu bên ngoài
- Introduction to Java Programming – Y. Daniel Liang
- Khoá học Java căn bản trên [Codecademy](https://www.codecademy.com/learn/java)

# Lập trình và ngôn ngữ lập trình

# Lập trình và Ngôn ngữ lập trình

- **Lập trình** là quá trình tạo ra tập các chỉ dẫn (instruction) để ra lệnh cho máy tính hoàn thành một công việc (task) nào đó
- Lập trình bao gồm rất nhiều hoạt động: Tìm hiểu yêu cầu, phân tích, thiết kế, viết code, kiểm thử, triển khai, bảo trì, mở rộng...
- Ngôn ngữ lập trình là phương tiện để lập trình viên viết ra các chỉ dẫn cho máy tính

# Các loại ngôn ngữ lập trình

- Có nhiều loại ngôn ngữ lập trình khác nhau, phục vụ cho các mục đích khác nhau
- Chẳng hạn, chúng ta có các ngôn ngữ lập trình web, ngôn ngữ lập trình desktop, ngôn ngữ lập trình mobile...
- Học lập trình có nghĩa là:
  - Học tư duy giải quyết vấn đề
  - Học một (hoặc một số) ngôn ngữ lập trình
- Học một ngôn ngữ lập trình có nghĩa là:
  - Học tư duy của ngôn ngữ đó
  - Học cú pháp của ngôn ngữ đó



# Khái niệm thuật toán

# Thuật toán (Algorithm)

- **Thuật toán**, còn gọi là **giải thuật**, là một tập hợp hữu hạn các chỉ thị hay cách thức được định nghĩa rõ ràng cho việc hoàn tất một số sự việc từ một trạng thái ban đầu cho trước.
- Ví dụ: Giả sử có hai bình A và B đựng hai loại chất lỏng khác nhau, chẳng hạn bình A đựng rượu, bình B đựng nước mắm. Giải thuật để hoán đổi (swap) chất lỏng đựng trong hai bình đó là:
  - Yêu cầu phải có thêm một bình thứ ba gọi là bình C.
  - Bước 1: Đổ rượu từ bình A sang bình C.
  - Bước 2: Đổ nước mắm từ bình B sang bình A.
  - Bước 3: Đổ rượu từ bình C sang bình B.

# Các cách biểu diễn giải thuật

- Mã giả (Pseudo-code)
- Lưu đồ (Flowchart)
- Ngôn ngữ lập trình

# Biểu diễn bằng mã giả

- Liệt kê tuần tự các bước bằng ngôn ngữ tự nhiên để biểu diễn thuật toán
- Ưu điểm
  - Đơn giản, không cần kiến thức về cách biểu diễn (lưu đồ, ngôn ngữ lập trình)
- Nhược điểm
  - Dài dòng, không cấu trúc
  - Đôi lúc khó hiểu, không diễn đạt được thuật toán

# Biểu diễn bằng mã giả - Ví dụ


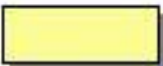




- Ví dụ: Giải thuật giải phương trình bậc nhất dạng  $ax+b=0$  như sau:
  - Bước 1: Nhận giá trị của các tham số  $a, b$
  - Bước 2: Xét giá trị của  $a$  xem có bằng 0 hay không? Nếu  $a=0$  thì làm bước 3, nếu  $a$  khác không thì làm bước 4.
  - Bước 3: ( $a$  bằng 0) Nếu  $b$  bằng 0 thì ta kết luận phương trình vô số nghiệm, nếu  $b$  khác 0 thì ta kết luận phương trình vô nghiệm.
  - Bước 4: ( $a$  khác 0) Ta kết luận phương trình có nghiệm  $x=-b/a$

# Biểu diễn bằng mã giả – Ví dụ

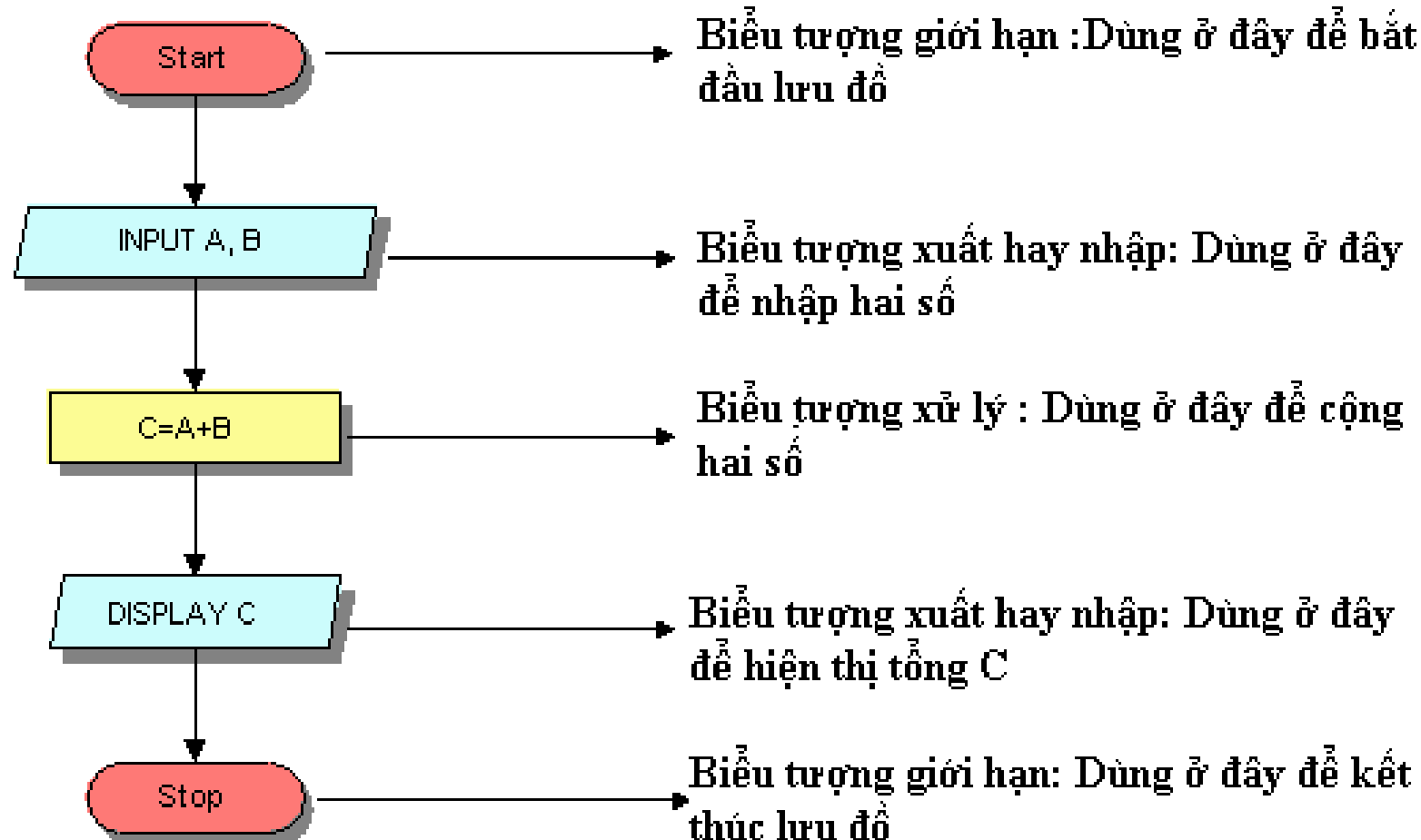
```
BEGIN
  INPUT a, b
  IF a=0
    IF b = 0
      DISPLAY "Infinitive solutions"
    ELSE
      DISPLAY "There are no solutions"
    END IF
  ELSE
    DISPLAY "x = -b/a"
  END IF
END
```

# Biểu diễn bằng lưu đồ (Flowchart)

- Lưu đồ mô tả giải thuật bằng các sơ đồ hình khối. Mỗi khối qui định một hành động.

Biểu Tượng	Mô Tả
	Bắt đầu hay kết thúc chương trình
	Những bước tính toán
	Các lệnh xuất hay nhập
	Quyết định và rẽ nhánh
	Bộ nối hai phần trong chương trình (đầu nối)
	Dòng chảy

# Lưu đồ Tính tổng hai số



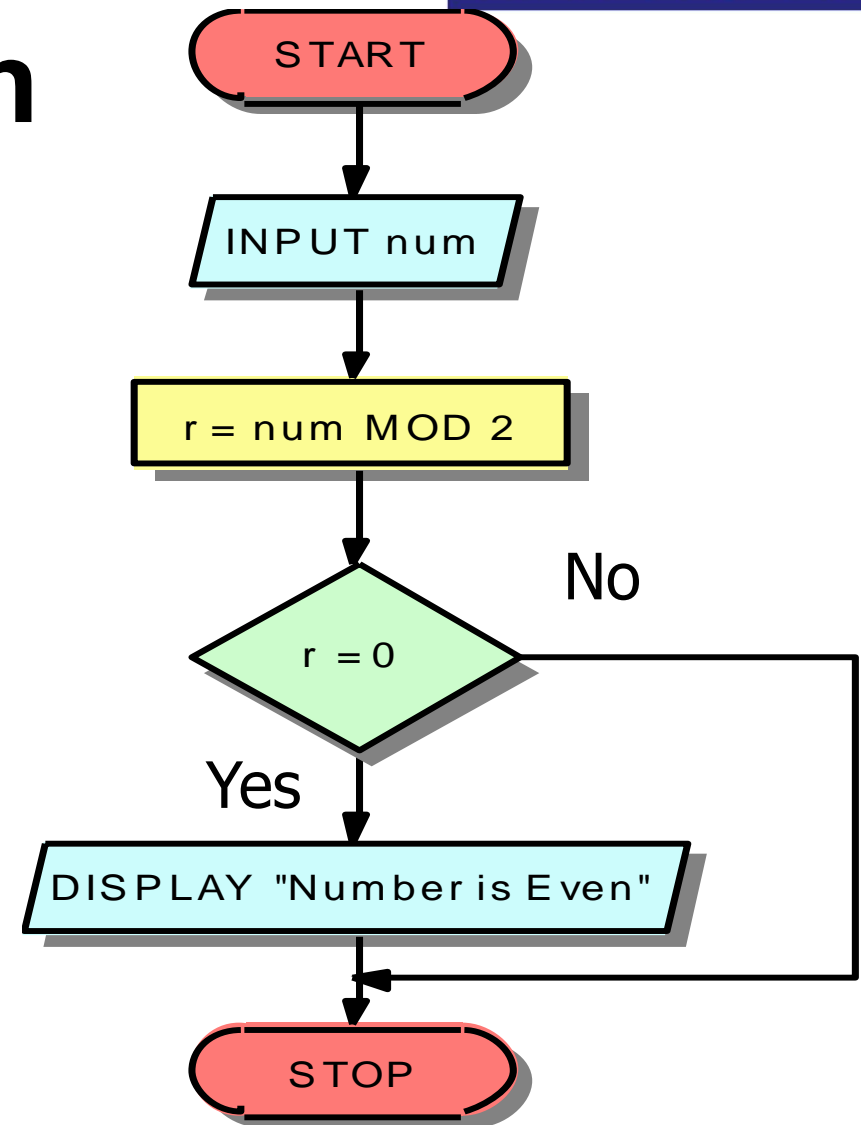


# Cấu trúc lựa chọn (selection)

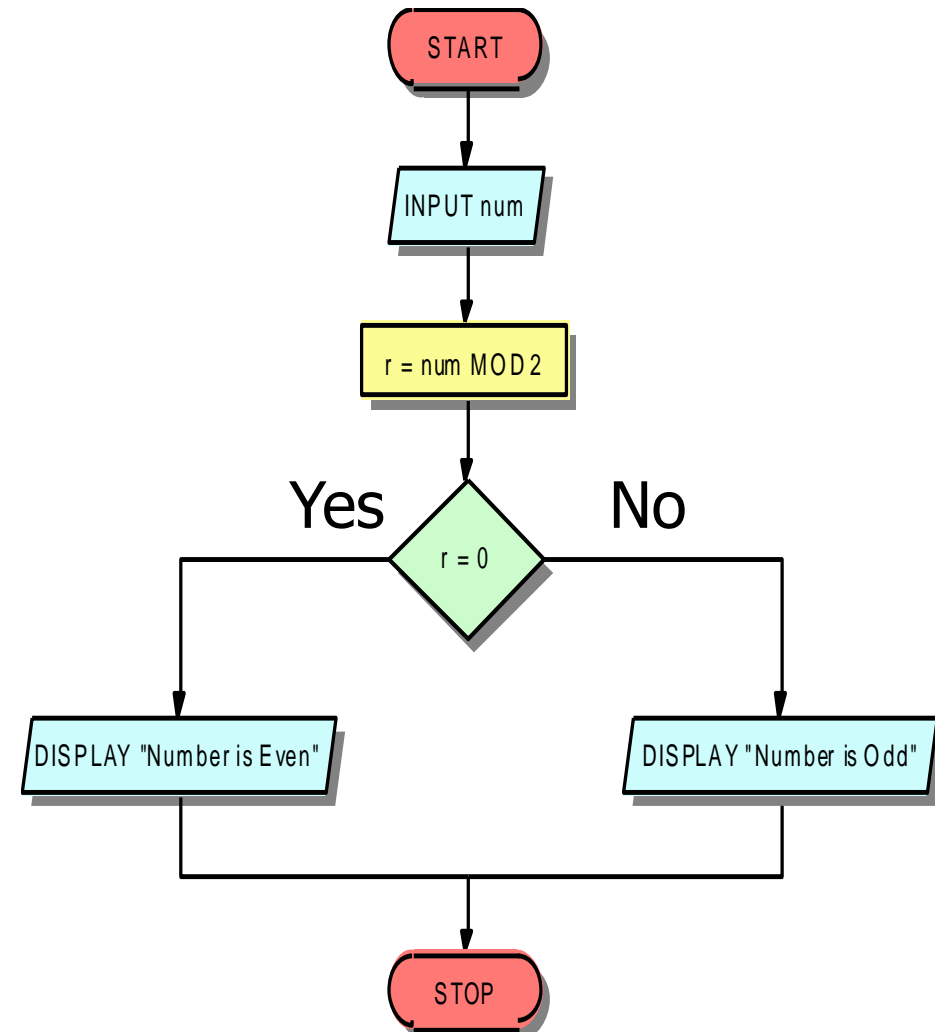
- Lựa chọn một công việc để thực hiện căn cứ vào một điều kiện nào đó.
- Có một số dạng cơ bản như sau:
  - Cấu trúc 1: Nếu < điều kiện > (đúng) thì thực hiện < công việc >
  - Cấu trúc 2: Nếu < điều kiện > (đúng) thì thực hiện < công việc 1 >, ngược lại (điều kiện sai) thì thực hiện < công việc 2 >
  - Cấu trúc 3: Trường hợp < i > thực hiện < công việc i >

# Lưu đồ kiểm tra số chẵn

```
BEGIN  
INPUT num  
r = num MOD 2  
IF r=0  
    Display "Number is even"  
END IF  
END
```



```
BEGIN
INPUT num
r=num MOD 2
IF r=0
    DISPLAY “Even Number”
ELSE
    DISPLAY “Odd Number”
END IF
END
```

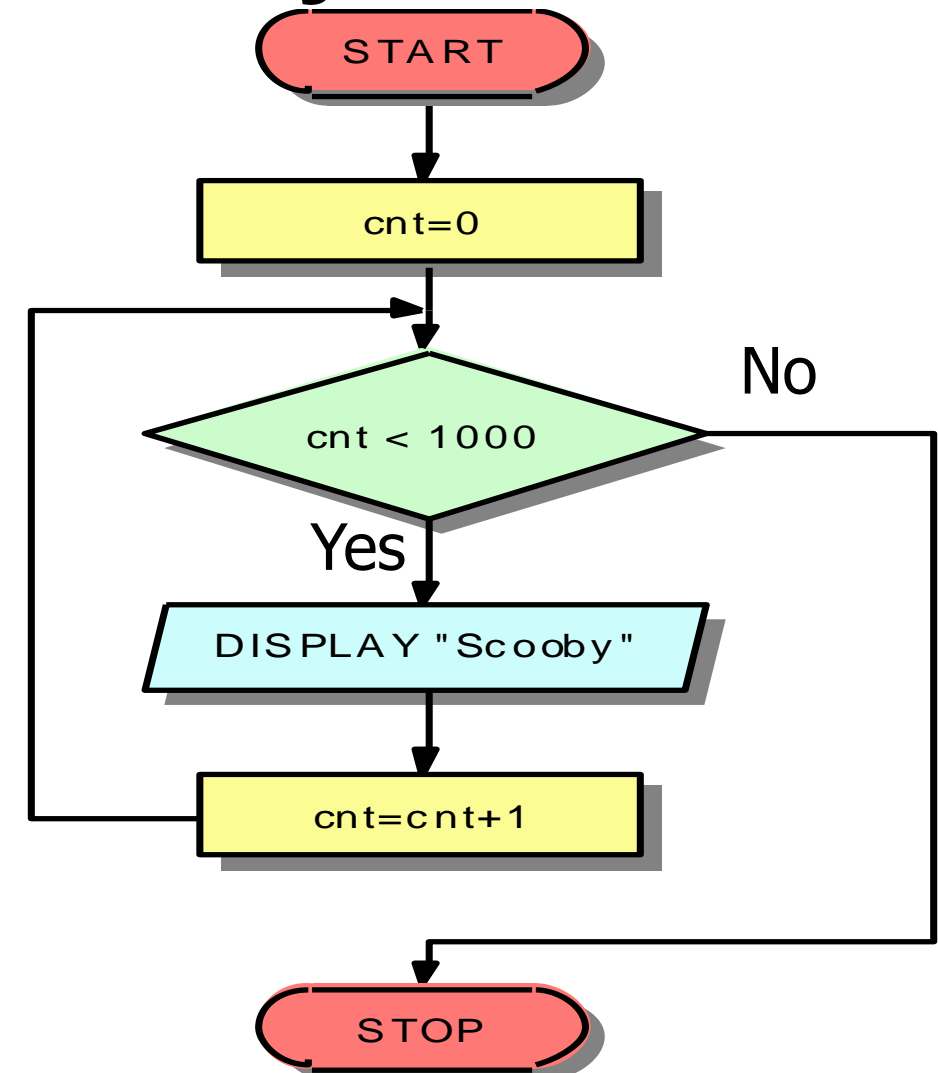


# Cấu trúc lặp (loop)

- Thực hiện lặp lại một công việc không hoặc nhiều lần căn cứ vào một điều kiện nào đó.
- Có hai dạng như sau:
  - Lặp xác định: là loại lặp mà khi viết chương trình, người lập trình đã xác định được công việc sẽ lặp bao nhiêu lần.
  - Lặp không xác định: là loại lặp mà khi viết chương trình người lập trình chưa xác định được công việc sẽ lặp bao nhiêu lần. Số lần lặp sẽ được xác định khi chương trình thực thi.

# Lưu đồ hiển thị 1000 từ Scooby

```
BEGIN  
cnt=0  
WHILE (cnt < 1000)  
DO  
    DISPLAY "Scooby"  
    cnt=cnt+1  
END DO  
END
```




**Giải quyết các bài toán cơ bản trong  
lập trình sử dụng `code.org`**

# Code.org

- Bước 1: Truy cập vào <https://code.org/student/elementary>

## Computer Science Fundamentals for Elementary Schools

For pre-readers in elementary school classrooms



**Course A**

An introduction to computer science for pre-readers.

Ages: 4-7



**Course B**

An introduction to computer science for pre-readers. (Similar to Course A, but with more variety for older students.)

Ages: 5-8

For older students in elementary school classrooms



**Course C**

Learn the basics of computer science and create your own art, stories, and games.

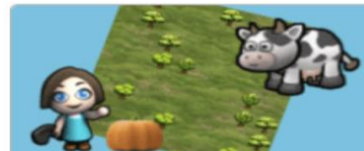
Ages: 6-10



**Course D**

Quickly cover concepts from Course C, then go further with algorithms, nested loops, conditionals, and more.

Ages: 7-11



**Course E**

Quickly cover concepts in Course C & D and then go further with functions.

Ages: 8-12



**Course F**

Learn all the concepts in Computer Science Fundamentals and create your own art, story or game.








Ages: 9-13

# Code.org

- Bước 2: Chọn lần lượt các course thực thành bắt đầu từ Course A

## Course A

Learn the basics of computer science and internet safety. At the end of the course, create your very own game or story you can share.

Continue	Get Help	 
Lesson Name	Progress	
1. Debugging: Unspotted Bugs		
2. Persistence: Stevie and the Bi...		
3. Real-life Algorithms: Plant a S...	Unplugged Activity	
4. Sequencing with Drag and Dr...		
5. Programming: Happy Maps	Unplugged Activity	
6. Programming in Maze		
7. Digital Citizenship: Going Pla...	Unplugged Activity	
8. Loops: Happy Loops	Unplugged Activity	
9. Loops in Collector		



# Code.org

- Bước 3: Chọn bài từng bài học
- Dạng Video

Lesson Name	1.ress
1. Debugging: Unspotted Bugs	1



Lesson 1: Debugging: Unspotted Bugs 1 MORE

### Video: Unspotted Bugs

Download Video

Continue

# Code.org

- Dạng bài học gồm: khái niệm cơ bản, bài tập, các hướng dẫn học bài

The screenshot shows the Code.org lesson page for 'Real-life Algorithms: Plant a Seed'. The page includes a video player with a woman planting a seed, a 'Lesson Plan' button, and a 'View Lesson Plan' button. A blue arrow points from the video player to the right, leading to a detailed lesson plan page.

**Lesson 3: Real-life Algorithms: Plant a Seed**

Unplugged | Algorithms

**Overview**

In this lesson, students will relate the concept of algorithms back to everyday, real-life activities by planting an actual seed. The goal here is to start building the skills to translate real-world situations to online scenarios and vice versa.

**Purpose**

In this lesson, students will learn that algorithms are everywhere in our daily lives. For example, it is possible to write an algorithm to plant a seed. Instead of giving vague or over-generalized instructions, students will break down a large activity into smaller and more specific commands. From these commands, students must determine a special sequence of instructions that will allow their classmate to plant a seed.

**Agenda**

- Warm Up (10 min)**
  - Vocabulary
  - What We Do Daily
- Main Activity (20 min)**
  - Real-Life Algorithms: Plant A Seed - Worksheet
- Wrap Up (10 - 20 min)**
  - Flash Chat: What did we learn?
  - Journaling
- Assessment (15 min)**
  - Real-Life Algorithms - Assessment
- Extended Learning**
  - Go Figure

**Objectives**

Students will be able to:

- Decompose large activities into a series of smaller events.
- Arrange sequential events into their logical order.

**Preparation**

- Watch the **Plant a Seed - Teacher Video**.
- Prepare supplies for planting seeds. You'll need seeds, dirt, and paper cups for each student or group.
- Print one **Real-Life Algorithms: Plant A Seed - Worksheet** for each student.
- Print one **Real-Life Algorithms - Assessment** for each student.
- Make sure each student has a **Think Spot Journal - Reflection Journal**.

**Links**

**Heads Up!** Please make a copy of any documents you plan to share with students.

**For the Teacher**

- Plant a Seed - Teacher Video**
- Real-Life Algorithms: Plant A Seed - Worksheet**
- Real-Life Algorithms - Assessment**
- Think Spot Journal - Reflection Journal**

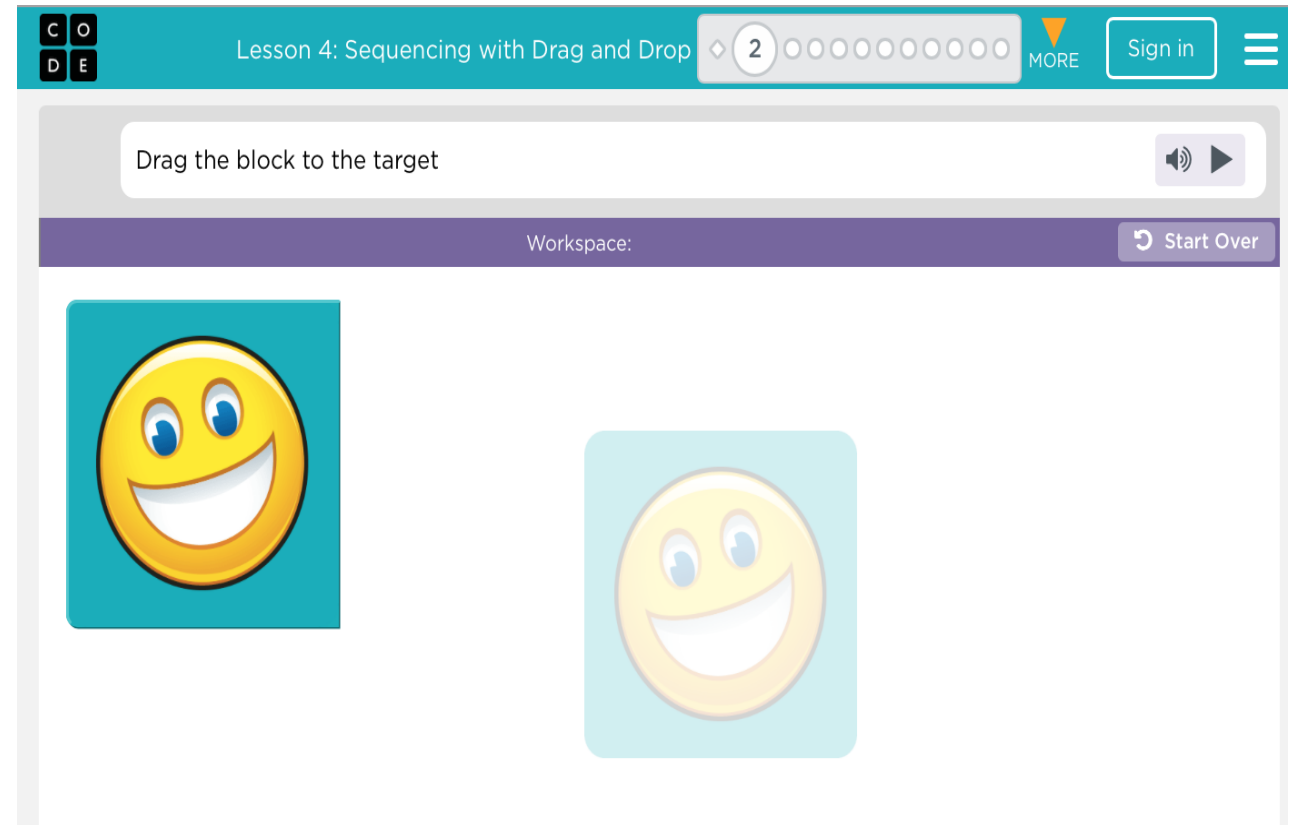
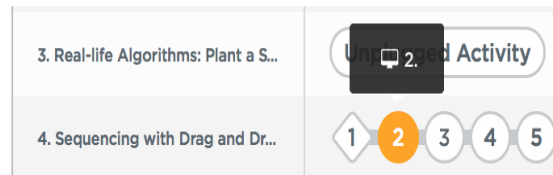
**Vocabulary**

- Algorithm** - A precise sequence of instructions for processes that can be executed by a computer

Chọn View Lesson Plan

# Code.org

- Bài thực hành



# [Bài tập] Mô tả các giải thuật cơ bản

**Bài 1:** Mô tả thuật toán nhập điểm của một sinh viên cho các môn: Vật lý, Hóa học, và Sinh học. Sau đó hiển thị điểm trung bình và tổng của những điểm này.

**Bài 2:** Mô tả thuật toán nhập một giá trị là độ 0C (Celsius) và chuyển nó sang độ 0F (Fahrenheit). [Hướng dẫn:  $C/5 = (F-32)/9$ ]

**Bài 3:** Mô tả thuật toán tính diện tích hình tròn

# [Bài tập] Bài toán có sử dụng cấu trúc điều kiện

**Bài 4:** Mô tả thuật toán nhập giá trị vào ba biến và in ra màn hình giá trị lớn nhất.

**Bài 5:** Mô tả thuật toán xếp loại sinh viên theo các qui luật dưới đây:

Nếu điểm $\geq 75$	-	Loại A
Nếu $60 \leq \text{điểm} < 75$	-	Loại B
Nếu $45 \leq \text{điểm} < 60$	-	Loại C
Nếu $35 \leq \text{điểm} < 45$	-	Loại D
Nếu điểm $< 35$	-	Loại E

# [Bài tập] Bài toán có sử dụng cấu trúc lặp

**Bài 6:** Mô tả thuật toán tìm số lớn nhất/nhỏ nhất trong một dãy số

**Bài 7:** Mô tả thuật toán game đoán số. Máy tính sẽ chọn ra một số ngẫu nhiên trong khoảng từ 0 đến 9. Người dùng sẽ đoán xem máy tính đã chọn số nào bằng cách nhập vào hộp thoại. Máy tính sẽ trả lời là đúng hay không.

# [Bài tập] Luyện tập code.org

## Course B

Learn the basics of computer science and internet safety. At the end of the course, create your very own game or story you can share.

[Try Now](#)
[Get Help](#)


Lesson Name	Progress
1. Debugging: Unspotted Bugs	1
2. Persistence: Stevie and the Bi...	1
3. Real-life Algorithms: Plant a S...	Unplugged Activity
4. Sequencing with Drag and Dr...	1 2 3 4 5 6 7 8 9 10 11 12 13
5. Digital Citizenship: My Digital...	Unplugged Activity
6. Programming: My Robotic Fri...	Unplugged Activity
7. Programming in Maze	1 2 3 4 5 6 7 8 9 10 11 12 13 14
8. Programming with Rey and B...	1 2 3 4 5 6 7 8 9 10 11
9. Loops: My Loopy Robotic Fri...	Unplugged Activity

## Course C

Create programs with loops, events, and conditionals. Translate your initials into binary, investigate different problem-solving techniques, and learn how to respond to cyberbullying. At the end of the course, create your very own game or story you can share.

[Try Now](#)
[Get Help](#)


Lesson Name	Progress
1. Persistence: Building a Found...	Unplugged Activity
2. Programming in Maze	1 2 3 4 5 6 7 8 9 10 11
3. Debugging in Maze	1 2 3 4 5 6 7 8 9 10
4. Real-life Algorithms: Paper PL...	Unplugged Activity
5. Programming in Collector	1 2 3 4 5 6 7 8 9 10 11 12 13
6. Programming in Artist	1 2 3 4 5 6 7 8 9 10
7. Loops: Getting Loopy	Unplugged Activity

# Tổng kết

- Lập trình là quá trình tạo ra các chỉ dẫn cho máy tính thực thi nhằm giải quyết một tác vụ nhất định
- Thuật toán (giải thuật) là các bước để giải quyết một vấn đề
- Mã giả (pseudo code) và lưu đồ (flowchart) là các cách thông dụng để mô tả giải thuật
- Cấu trúc lựa chọn (selection) sẽ đưa ra các hành động dựa trên việc đánh giá một điều kiện
- Cấu trúc lặp (loop) sẽ thực thi các hành động nhiều lần dựa trên việc đánh giá một điều kiện
- Code.org là một nền tảng cho phép luyện tập tư duy giải quyết vấn đề dựa trên các tình huống cụ thể



# Tài liệu

- [CodeGymX](#)
- [Code.org](#)