

# Computer Programming and Developing Applications for Education

## **Chapter 1: Computer Programming**

### **DTI3302 Computer Programming and Developing Applications for Education**

Department of Digital Technology for Education

Faculty of Education, Suan Sunandha Rajabhat University

**Content Credit By:** Asst.Prof.Nutthapat Kaewrattanapat, PhD.



**Pasawut Cheerapakorn**

Suan Sunandha Rajabhat University

## Course Description:

หลักการ ทฤษฎี ที่เกี่ยวข้องกับการเขียนโปรแกรมและพัฒนาแอปพลิเคชัน  
หลัก การพัฒนาโปรแกรมคอมพิวเตอร์ คุณสมบัติของโปรแกรมภาษาชนิด  
ต่าง ๆ หลักการเบื้องต้นเกี่ยวกับองค์ประกอบ ลักษณะคำสั่ง การเขียน  
โปรแกรม ขั้นตอนวิธี การวิเคราะห์ การออกแบบ แอปพลิเคชันเพื่อการ  
ศึกษา การประเมิน ซอฟต์แวร์ สามารถพัฒนาแอปพลิเคชันเพื่อการศึกษา

Principles, theories associated with computer programming and  
development applications, computer programming principles,  
computer language, elements of computer language, syntax,  
computer programming, algorithms, analysis and design application  
for education, software evaluation, candidate teachers able to  
developing applications for education.

Algorithm Design

Python Programming

AI for Programming

## Course Outline:

- **Chapter 1 - Computer Programming**
- Chapter 2 - Introduction to Python Programming
- Chapter 3 - Conditional Statement
- Chapter 4 - Iteration Statement
- Chapter 5 - Strings
- Chapter 6 - Lists, Tuples, Sets, Dictionaries
- Chapter 7 - Functions
- Chapter 8 - Object-Oriented Programming: OOP

# Measurement and Evaluation:

## การวัดและประเมินผล

### 1. ระหว่างการจัดการเรียนรู้

- สอบ Pre-test 0%
- การมอบหมายงาน 20%
- สอบ Post-test 15%
- การมีส่วนร่วมในชั้นเรียน 5%

### 2. การสอบกลางภาค (Midterm Examination)

- ปรนัย 35 ข้อ (35 คะแนน) อัตนัย 1 ข้อ (5 คะแนน) 20%

### 3. โครงการประจำภาคเรียน (Term Project)

- โครงการและการนำเสนอ 20%

### 4. การสอบปลายภาค (Final Examination)

- ปรนัย 35 ข้อ (35 คะแนน) อัตนัย 1 ข้อ (5 คะแนน) 20%

ร้อยละ	ระดับผลการเรียน	ความหมาย
86 – 100	A	ดีเยี่ยม
82 – 85	A-	ดีเยี่ยม
78 – 81	B+	ดีมาก
74 – 77	B	ดี
70 – 73	B-	ค่อนข้างดี
66 – 69	C+	ปานกลางค่อนข้างดี
62 – 65	C	ปานกลาง
58 – 61	C-	ปานกลางค่อนข้างอ่อน
54 – 57	D+	ค่อนข้างอ่อน
50 – 53	D	อ่อน
46 – 49	D-	อ่อนมาก
0 – 45	F	ตก

## Measurement and Evaluation:

ครั้งที่ / สัปดาห์	บทเรียน / หัวข้อ
1	แนะนำรายวิชา การวัดและการประเมินผล หัวข้อเรียนรู้ (Introduction to Course)
2	<b>บทที่ 1 การโปรแกรมคอมพิวเตอร์ (Computer Programming)</b>
3	บทที่ 2 พื้นฐานการโปรแกรมภาษาไพทอน (Introduction to Python)
4	บทที่ 3 การโปรแกรมแบบตัดสินใจ (Decision)
5	บทที่ 4 การโปรแกรมแบบทำซ้ำ (Iteration)
6	บทที่ 5 การโปรแกรมสายอักขระ (String)
7	บทที่ 6 ลิสต์ ทัวเปิล เซต และดิกชันนารี (List, Tuple, and Dictionaries)
8	บทที่ 7 ฟังก์ชัน (Function)

## Measurement and Evaluation:

ครั้งที่ / สัปดาห์	บทเรียน / หัวข้อ
9	สอบกลางภาค (Midterm Examination)
10	บทที่ 7 ฟังก์ชัน (Function) (ต่อ)
11	บทที่ 8 การโปรแกรมเชิงวัตถุ (Object-Oriented Programming: OOP)
12	บทที่ 8 การโปรแกรมเชิงวัตถุ (Object-Oriented Programming: OOP) (ต่อ)
13	การเขียนโปรแกรมด้วยปัญญาประดิษฐ์ (AI for Programming)
14	สอบปลายภาค (Final Examination)
15	นำเสนอและส่งโครงงาน (Project Pitching and Presentation)
16	

# Pre Test

**Question:**

**1. ภาษา Python เป็นภาษาคอมพิวเตอร์ลักษณะใด?**

**A**

Interpreter

**B**

Compiler

**Question:**

## 2. ข้อใดไม่เป็นองค์ประกอบของระบบคอมพิวเตอร์

**A** Hardware

**B** Software

**C** Peopleware

**D** Data/Information

**E** Communication

**F** Procedures

**Question:**

**3. "110100101010101" เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?**

**A** Low-Level Language

**C** Assembly Language

**B** High-Level Language

**D** Procedural Language

**Question:**

**4. print("Panda") เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?**

**A** Low-Level Language

**C** Assembly Language

**B** High-Level Language

**D** Procedural Language

**Question:**

**5. ข้อใดไม่ใช่คุณลักษณะของอัลกอริทึม**

**A**

ไม่คลุมเครือ  
(Unambiguous)

**C**

มีจุดสิ้นสุดการทำงาน  
(Finiteness)

**B**

มีขั้นตอนตายตัว  
(Deterministic)

**D**

มีหน่วยความจำที่เพียงพอ  
(Memory Capacity)

**Question:**

## 6. ข้อใดไม่ใช่เครื่องมือในการโปรแกรมภาษา Python

**A** Pycham

**C** Jupyter

**B** Colab

**D** Assembler

**Question:**

**7. ให้ผลลัพธ์ถูกต้องตามเงื่อนไขที่กำหนดไว้เสมอ คือ คุณลักษณะข้อใด?**

**A**

ไม่คลุมเครือ  
(Unambiguous)

**C**

มีจุดสิ้นสุดการทำงาน  
(Finiteness)

**B**

มีขั้นตอนตายตัว  
(Deterministic)

**D**

มีหน่วยความจำที่เพียงพอ  
(Memory Capacity)

**Question:**

## 8. วิธีการสอนใดเหมาะกับการเรียนรู้ด้านการโปรแกรมคอมพิวเตอร์

**A**

Passivity-based  
Learning

**C**

Experiential Learning

**B**

Cognitivism

**D**

Adaptive Learning

**Question:**

**9. การสอน Coding สิ่งแรกที่ต้องสอนให้ผู้เรียนคำนึงถึง  
ในการโปรแกรม คืออะไร?**

**A**

Input

**C**

Output

**B**

Process

**D**

Memory

**Question:**

**10. การสอน Coding สิ่งสุดท้ายที่ต้องสอนให้ผู้เรียนคำนึงถึง  
ในการโปรแกรม คืออะไร?**

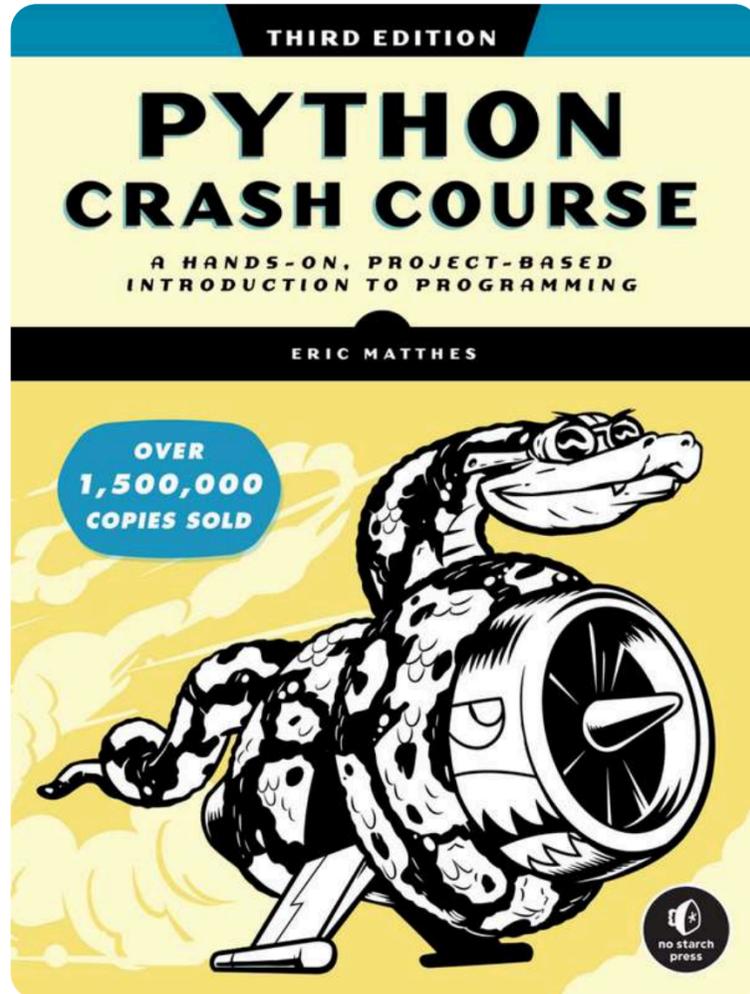
**A** Input

**C** Output

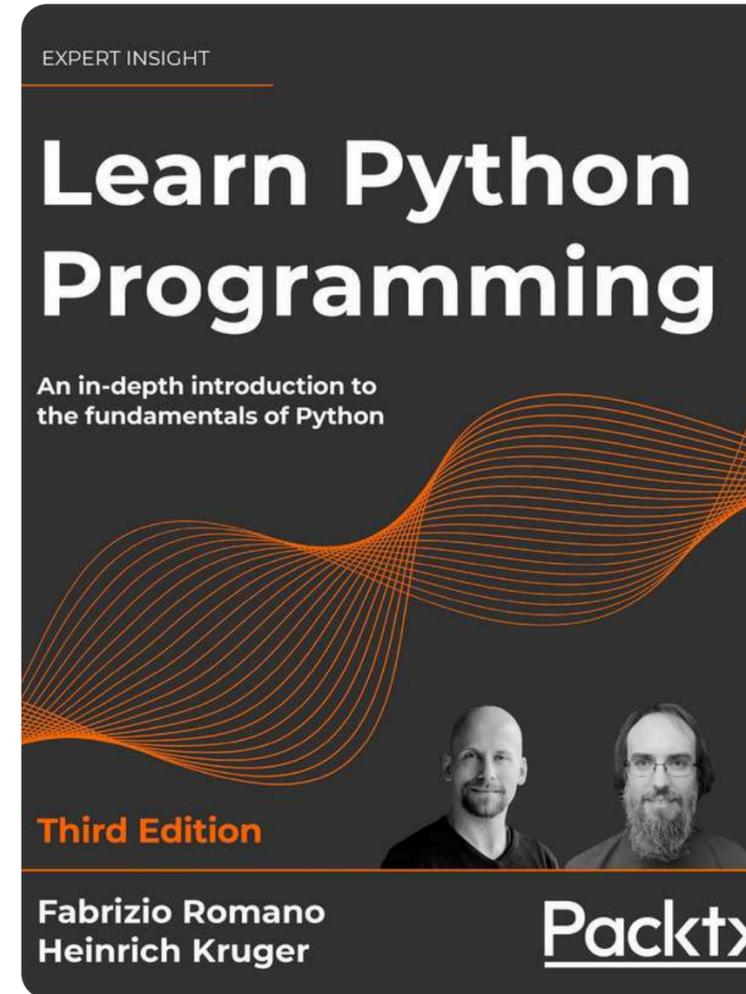
**B** Process

**D** Memory

## Learning Materials Suggestion:



Python Crash Course: A Hands-On,  
Project-Based Introduction to  
Programming [3 ed.]



Learn Python Programming  
An in-depth introduction to the  
fundamentals of Python [3 ed.]

## Learning Tools:

Python  
Fundamental

(Others eg. Pycharm, Jupiter, etc.)



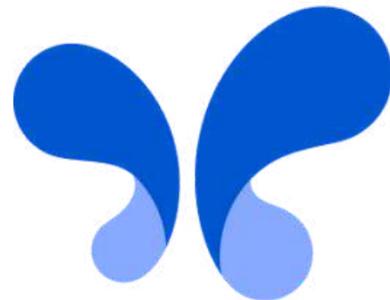
<https://colab.research.google.com/>

Python &  
Application



<https://anaconda.org/anaconda/python>

AI for  
Programming



<https://aistudio.google.com/apps>

# Learning by Doing - Experiential Learning

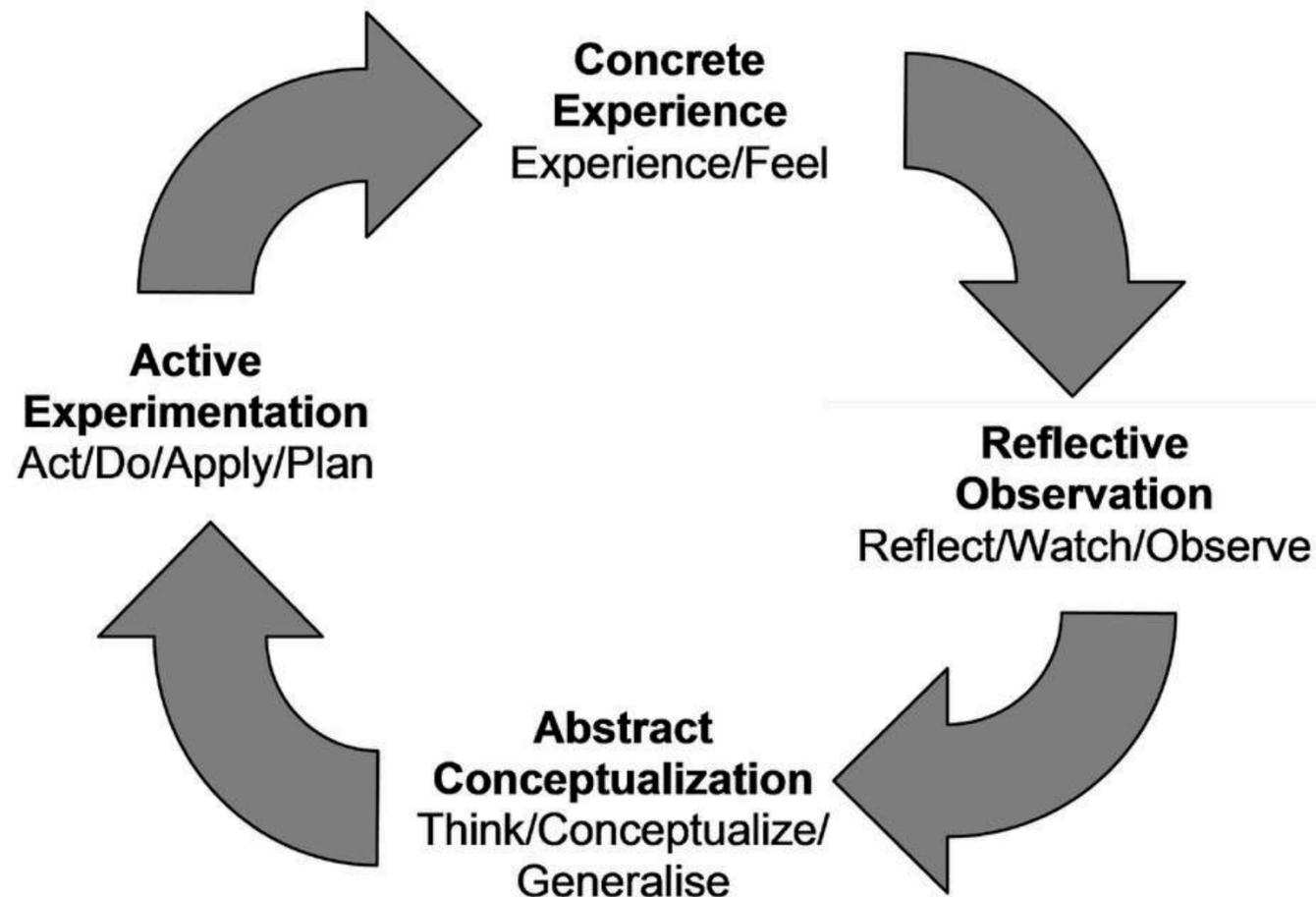
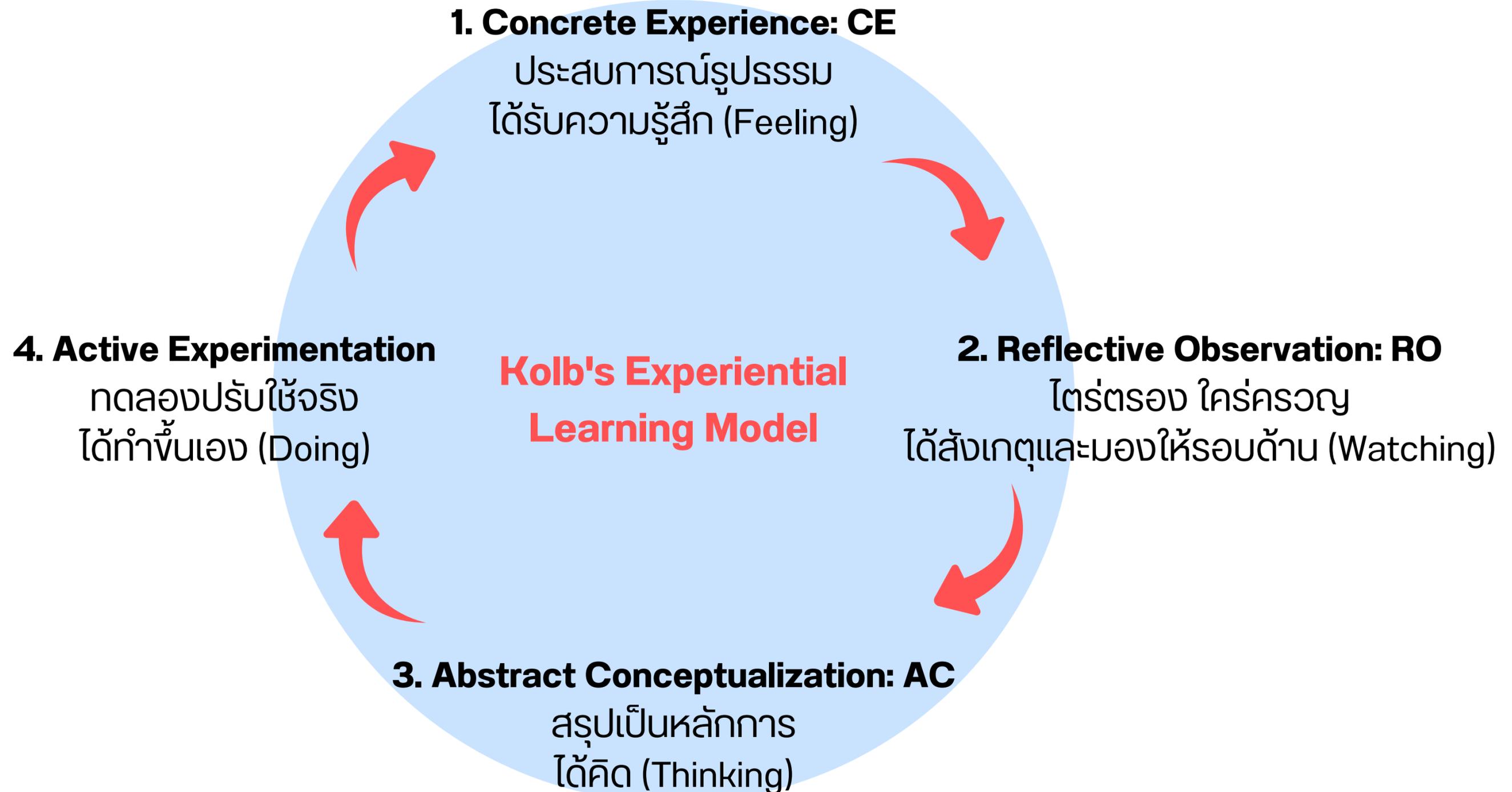


Figure 1. Model of Kolb's (1984) Experiential Learning

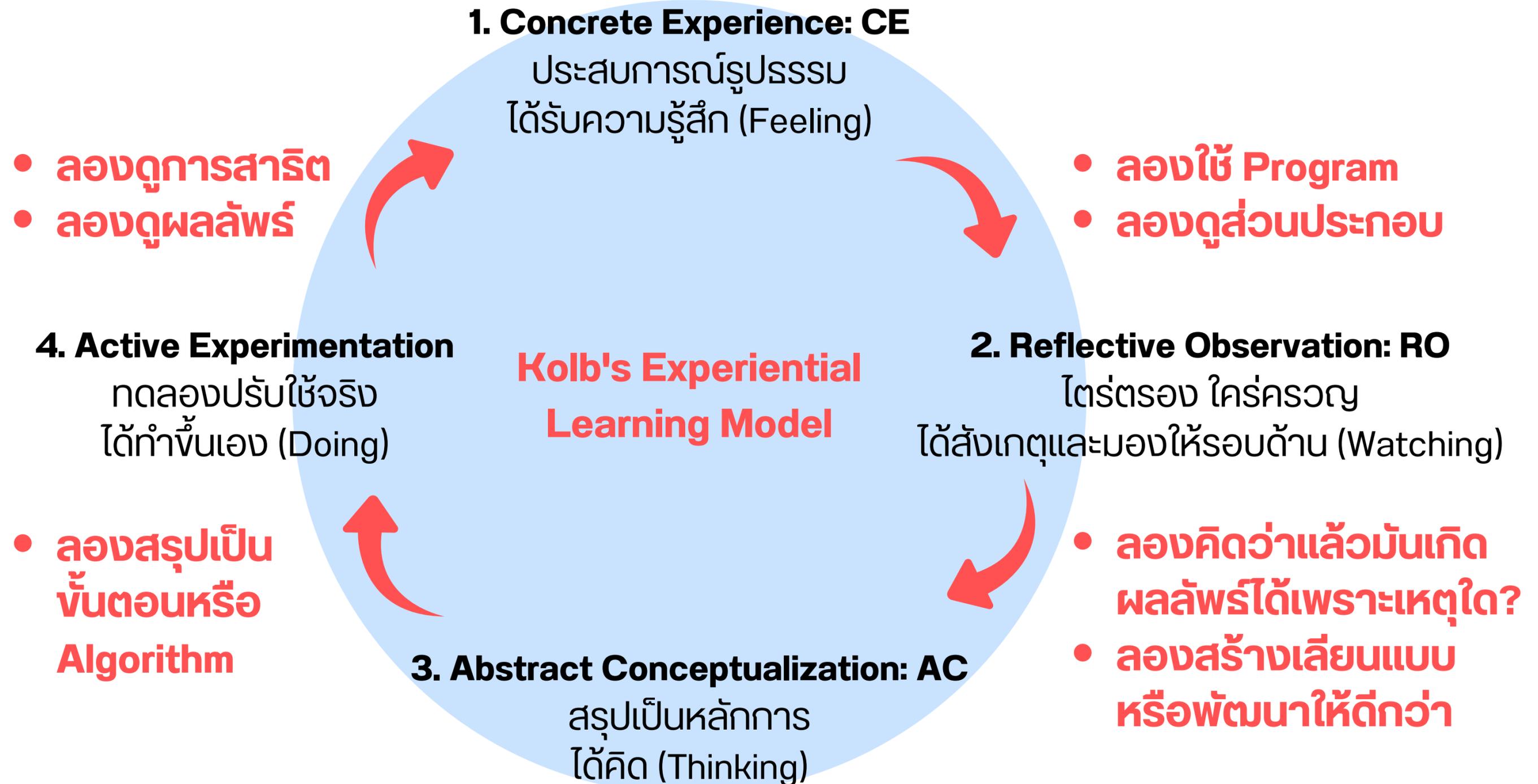
**Experiential Learning** หมายถึง การเรียนรู้ที่เกิดขึ้นจากประสบการณ์ที่ได้ทำและปฏิบัติจริง ไม่ใช่แค่การเรียนรู้จากหนังสือหรือบทเรียนที่ถูกสอนเท่านั้น เมื่อผู้เรียนได้มีโอกาสสัมผัสและทดลองทำสิ่งต่างแล้ว จะช่วยให้เกิดการเรียนรู้ที่มีความหมายและจดจำ ได้มากขึ้นเช่น

- การศึกษาด้วยการทำโครงการ (Project)
- การฝึกทักษะทางวิชาชีพ หรือ
- การเรียนรู้ผ่านการทำกิจกรรมที่มีจุดมุ่งหมายในการสร้างประสบการณ์และความรู้ใหม่ ๆ ผ่านการทดลองและปฏิบัติจริงในชีวิตประจำวันของตนเอง

# Learning by Doing - Experiential Learning



# Learning by Doing - Experiential Learning



# Experiential Learning & Computer Programming

ขั้นตอนที่  
1

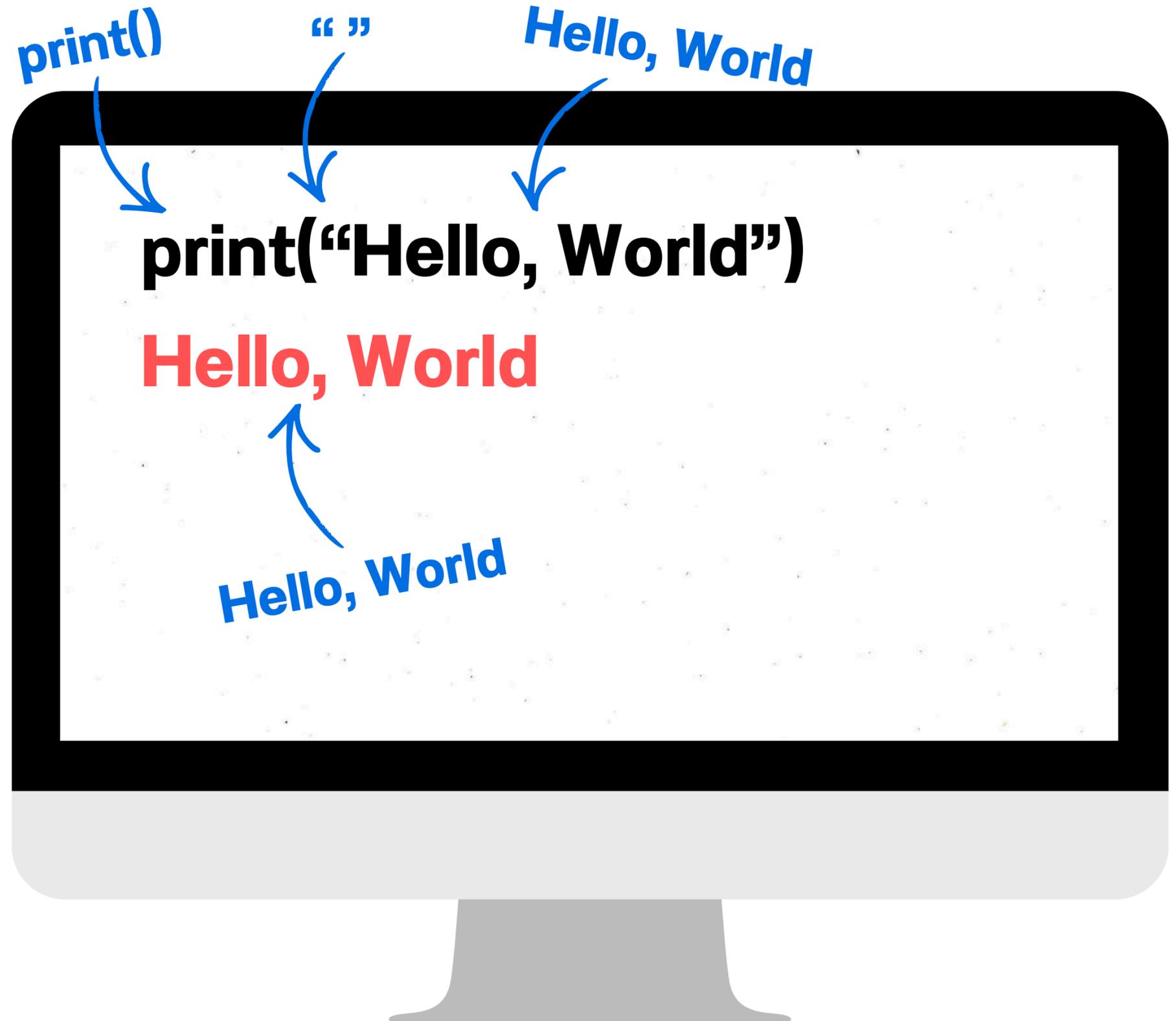
**Concrete Experience: CE**  
ประสบการณ์รูปธรรม

```
print("Hello, World")  
Hello, World
```

# Experiential Learning & Computer Programming

ขั้นตอนที่  
2

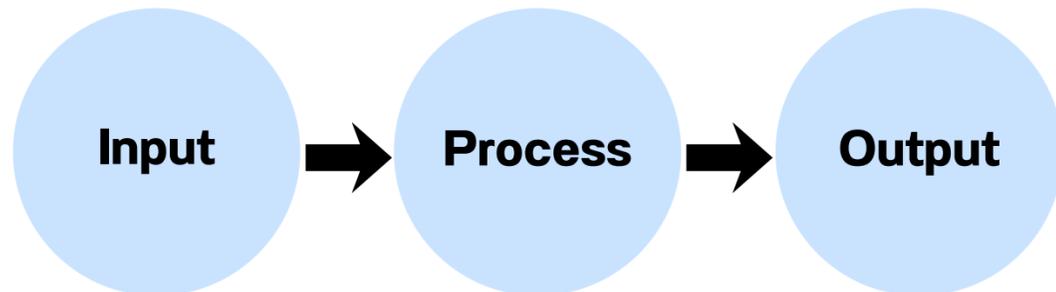
**Reflective Observation: RO**  
ไตร่ตรอง ใคร่ครวญ



# Experiential Learning & Computer Programming

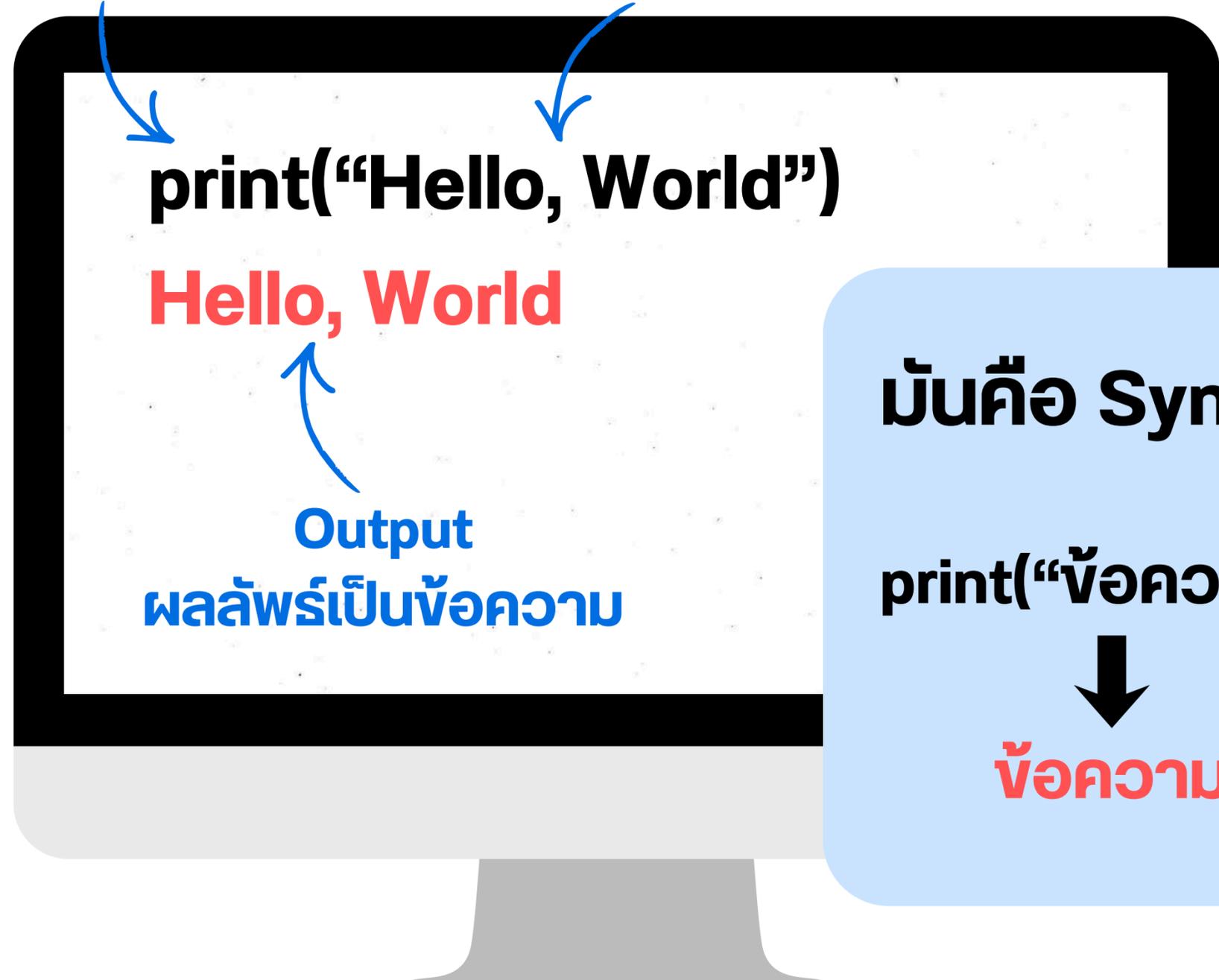
ขั้นตอนที่  
3

Abstract Conceptualization: AC  
สรุปเป็นหลักการ



Process  
คำสั่งพิมพ์

Input  
ข้อความที่อยู่ภายใต้ “ ”



มันคือ Syntax

print("ข้อความ")

↓  
ข้อความ

# Experiential Learning & Computer Programming

ขั้นตอนที่  
4

**Active Experimentation**  
ทดลองปรับใช้จริง

```
print("Hello, Pasawut")  
print("How are you?")
```

**Hello, Pasawut**  
**How are you?**

# Experiential Learning & Computer Programming



**Course  
Syllabus**



**Hello,  
World!**



**Algorithm**



**Bug!**



**Project**



**Project  
Finish**

# Chapter 1 - Computer Programming

**Computer programming** refers to the process of designing and creating sequences of instructions (code) that enable a computer to perform specific tasks or functions.

It involves writing and organizing instructions that a computer can understand and execute to achieve desired outcomes.

Programs are typically written in **programming languages such as Python, Java, C++, and others**. These languages use specific syntax and rules to communicate with computers, enabling developers to create applications, websites, software, or systems ranging from simple to complex.

**การเขียนโปรแกรมคอมพิวเตอร์** หมายถึง การออกแบบและสร้างลำดับขั้นตอนของคำสั่ง (อัลกอริทึมและโค้ด) ที่ทำให้คอมพิวเตอร์สามารถทำงานหรือ ปฏิบัติภารกิจเฉพาะตามที่กำหนดได้

**การเขียนโปรแกรมคอมพิวเตอร์** เป็นกระบวนการที่เกี่ยวข้องกับการเขียน และจัดระเบียบคำสั่งที่คอมพิวเตอร์สามารถเข้าใจและดำเนินการตามเพื่อให้ได้ผลลัพธ์ตามที่ต้องการ

โปรแกรมทั่วไปจะถูกเขียนขึ้นด้วย**ภาษาโปรแกรม เช่น Python, Java, C++ และอื่น ๆ** ซึ่งภาษาเหล่านี้ใช้ไวยากรณ์และกฎเฉพาะ เพื่อสื่อสารกับคอมพิวเตอร์ ทำให้นักพัฒนาสามารถสร้างแอปพลิเคชัน เว็บไซต์ ซอฟต์แวร์ หรือระบบต่าง ๆ ที่มีความซับซ้อนตั้งแต่ง่ายไปจนถึงยากได้

# โค้ดดัง Coding คือข้อใด?

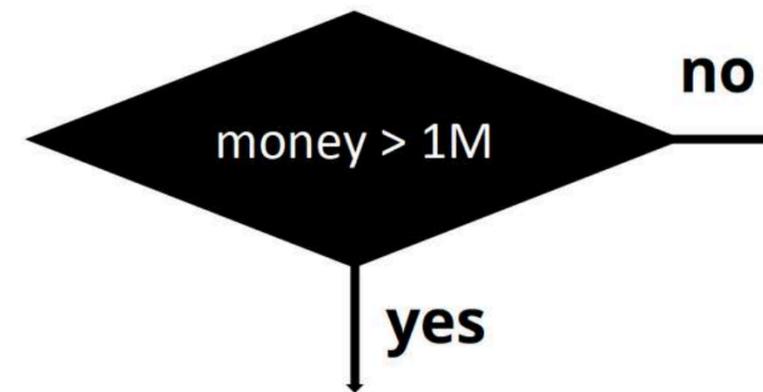
**A**

10011100110101

**B**

print("Hello World!")

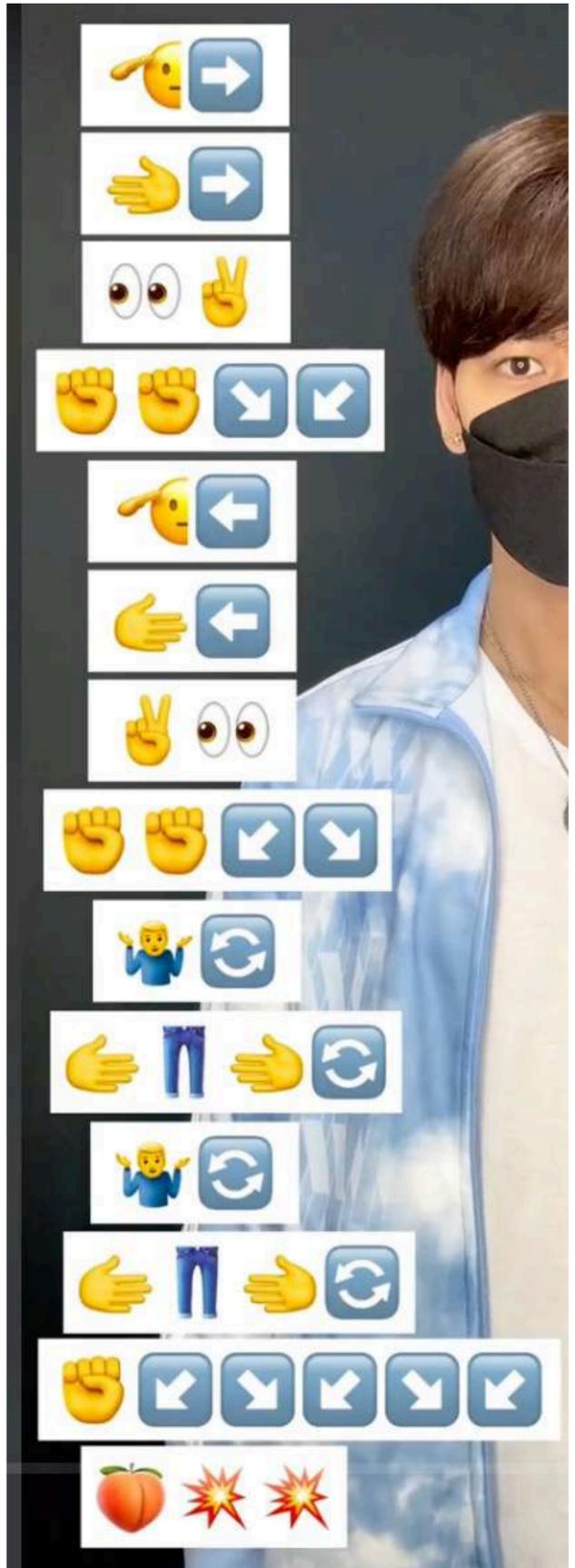
**C**



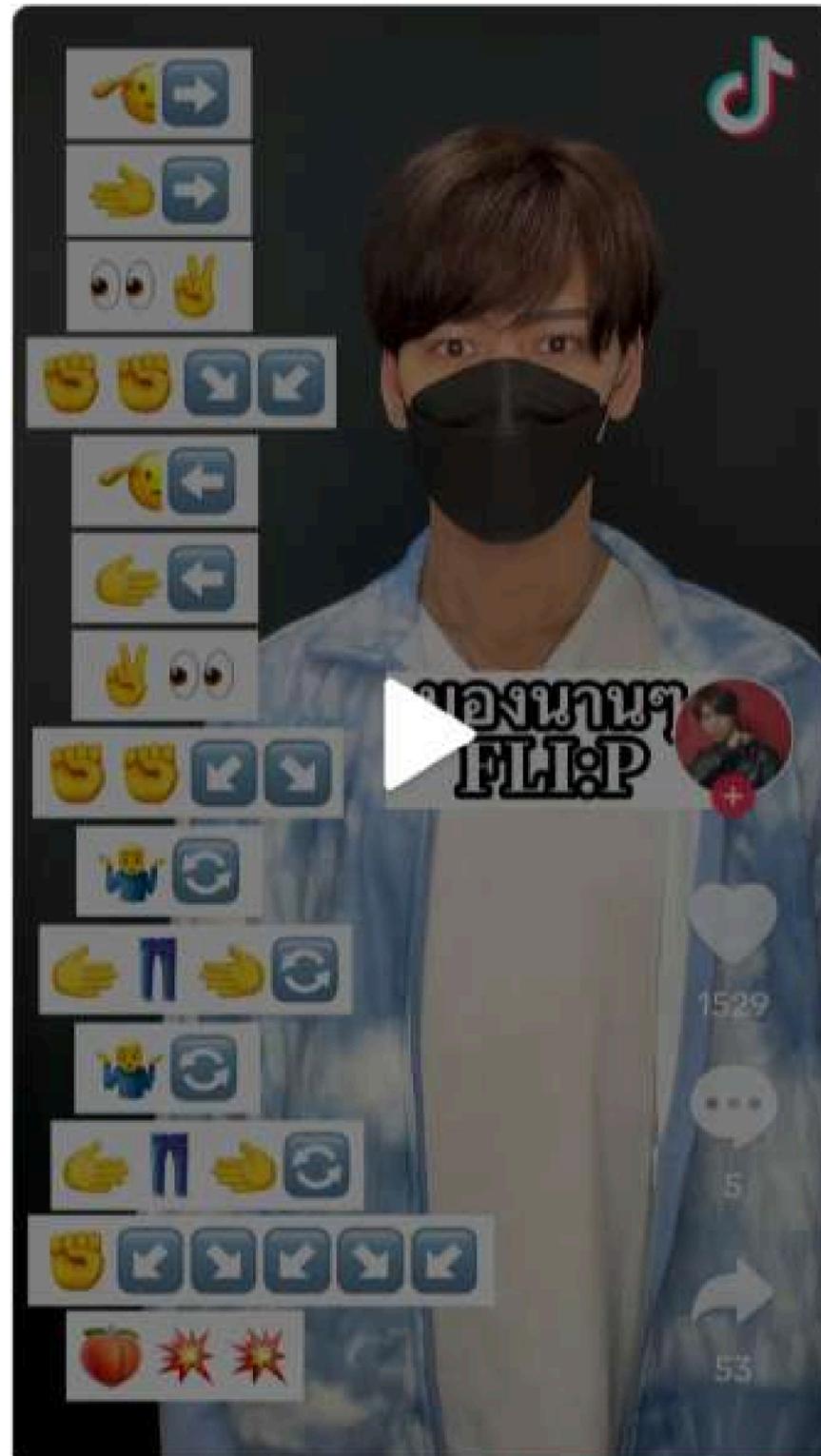
**D**



# ทำเต็มเพลงอะไร?



# ทำเต็มเพลงอะไร?



ikTok Watch more [Watch now](#)

@xxtakaharuxx

#มองนานๆ #มองนานๆchallenge

# โค้ดดิ้ง (Coding)

- Code คือ ระบบของคำ อักษร หรือ สัญลักษณ์ ในการแทนข้อความในรูปแบบหนึ่งๆ หรืออาจเป็นระบบจำนวน/ตัวเลข อักษร หรือสัญลักษณ์ที่ใช้แทนอะไรบางอย่างในลักษณะที่สั้นกระชับ หรือ อยู่ในรูปแบบที่เข้าใจได้สะดวกขึ้น
- ดังนั้น พวก รหัส, เลขฐาน, สูตร, Pseudocode, Flowchart, Diagram, Computer Programming ต่าง ๆ นับว่าเป็น Code ในลักษณะหนึ่ง เพราะอยู่ในระบบของคำ สัญลักษณ์ที่สั้นกระชับ ชัดเจน สะดวกต่อการทำความเข้าใจ และการแปลงไปสู่หน่วยอื่น ๆ ได้



dictionary.cambridge.org

Cambridge Dictionary

## code

noun • UK  /kəʊd/ US  /koʊd/

**code** noun (LANGUAGE)

- ★ **B2** [C or U] **a system of words, letters, or signs used to represent a message in secret form, or a system of numbers, letters, or signals used to represent something in a shorter or more convenient form:**  
*The message was written **in** code.*  
*She managed to **decipher/break/crack** (= succeed in understanding) the code.*  
*Each entry in this dictionary has a grammar code.*

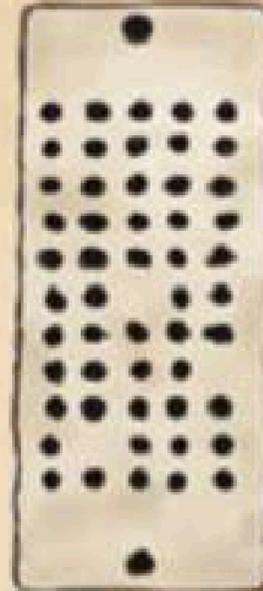
# Coding - Debug

## ADA LOVELACE FIRST COMPUTER PROGRAMMER

### The Analytical Engine

Lovelace's program turned a complex formula into simple calculations that could be encoded on punched cards and fed into Charles Babbage's Analytical Engine, a mechanical computer that he designed but never built. She published it in 1843, a century before the modern computer age.

*"I went to put in something about Bernoulli's Number, in one of my Notes, as an example of how an explicit function may be worked out by the engine, without having been worked out by human head and hands first."*



$$\frac{x}{e^x - 1} = \frac{1}{1 + \frac{x}{2} + \frac{x^2}{2 \cdot 3} + \frac{x^3}{2 \cdot 3 \cdot 4} + \text{etc.}}$$

### A Universal Computer

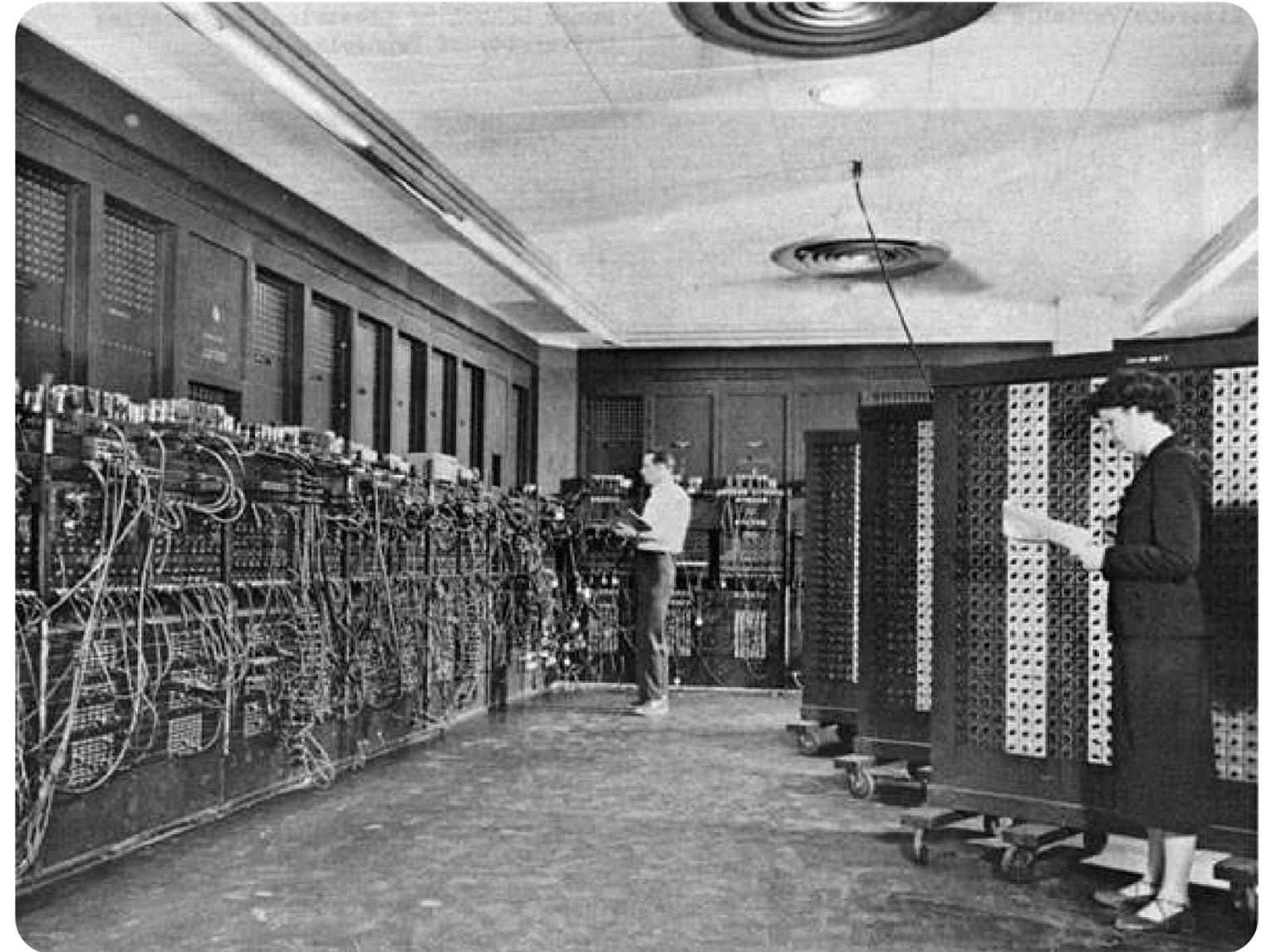
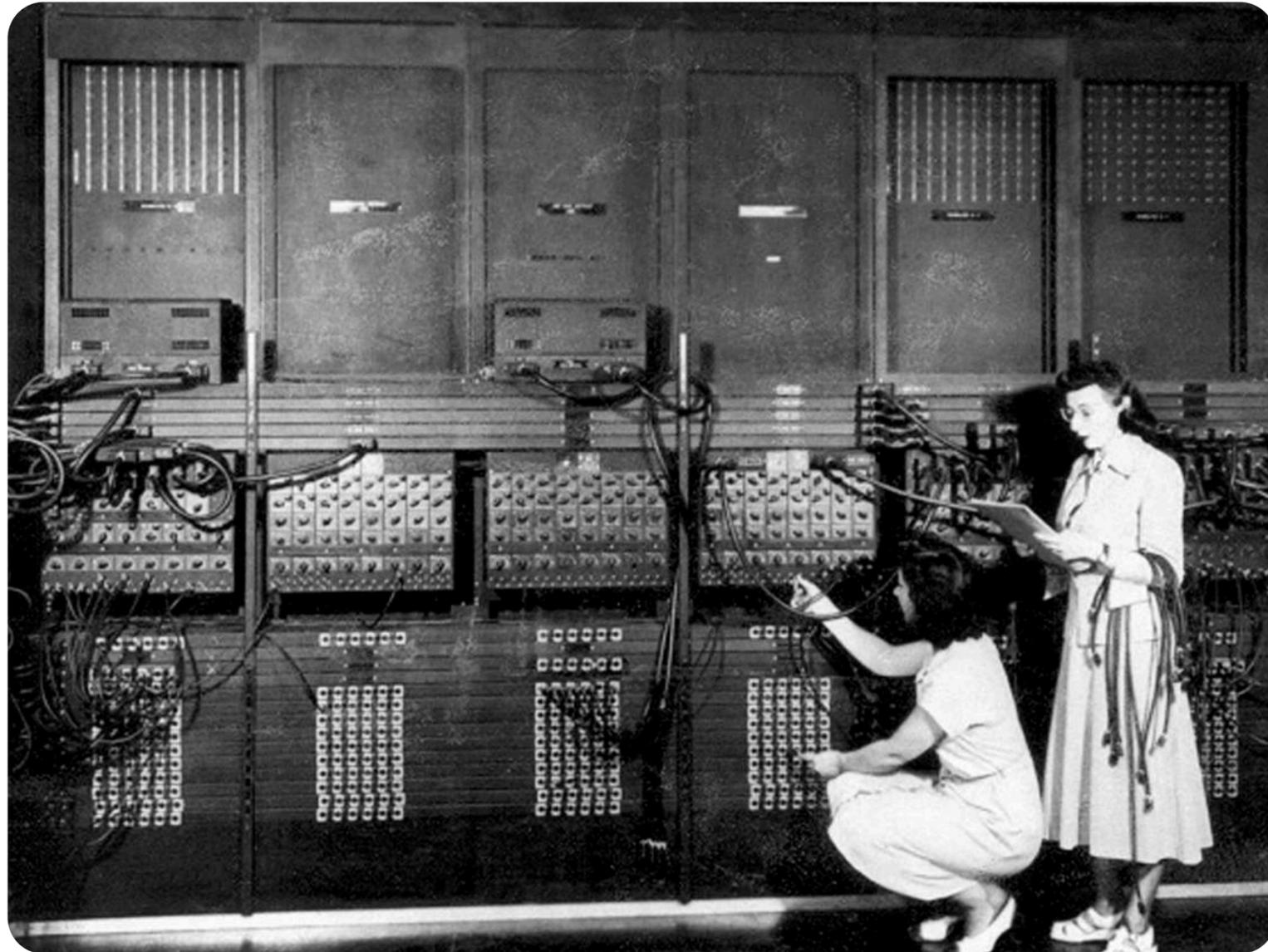
Lovelace did more than write the first computer program. She was also the first person to realize that a general purpose computer could do anything, given the right data and instructions.

*"The Analytical Engine weaves algebraic patterns just as the Jacquard loom weaves flowers and leaves."*

*"Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent."*



# Coding - Debug





**On the 9th of September of 1946,  
a Harvard technical team was asked  
to take a look at a malfunction that was  
happening with the Mark II calculator.**

# Coding - Debug

9/9

0800 Antan started  
1000 " stopped - antan ✓

1300 (032) MP-MC { 1.2700 9.037847025  
                  ~~1.98214000~~ 9.037846995 correct  
                  2.130476415 (-2) 4.615925059 (-2)  
(033) PRO 2 2.130476415  
              correct 2.130676415

Relays 6-2 in 033 failed special speed test  
in relay " 11.000 test.

1100 Started Cosine Tape (Sine check)  
1525 Started Multi-Adder Test.

1545  Relay #70 Panel F  
(moth) in relay.

First actual case of bug being found.

1630 Antan started.  
1700 closed down.

Relay 2145  
Relay 3370



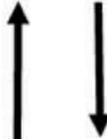
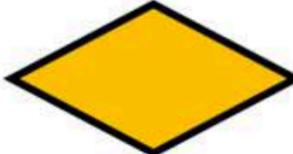
**Grace Hopper**, an American computer scientist and United States Navy rear admiral working on the Mark II calculator project at the time, added the caption “First actual case of bug being found,” and that’s the first time anyone used the word bug to describe a computer glitch. Naturally, the term debugging followed.

# Grace Brewster Hopper



# โค้ดดิ้ง (Coding)

รหัสจำลองสำหรับการเดินทาง	
รหัสจำลอง	ความหมาย
→	เดินไปข้างหน้า
↑	เดินขึ้นข้างบน
↓	เดินลงไปข้างล่าง
←	เดินไปทางซ้าย
↗	เดินเฉียงขวา
↖	เดินเฉียงซ้าย
↗	เดินเฉียงขึ้นข้างบน
↘	เดินเฉียงลงข้างล่าง

สัญลักษณ์	ความหมาย	ภาษาอังกฤษ
	หมายถึง จุดเริ่มต้น หรือ จุดสิ้นสุดการทำงาน	Begin และ End
	หมายถึง ทิศทางการไหลของงาน	Flow line
	หมายถึง การดำเนินการ หรือ การประมวลผล	Process
	หมายถึง การนำเข้าข้อมูลด้วย การป้อนค่าทางแป้นพิมพ์	Manual input
	หมายถึง การตัดสินใจตาม เงื่อนไขที่กำหนดไว้	Decision
	หมายถึง การแสดงผลบนหน้าจอ	Display

# โค้ดดิ้ง (Coding)

## รหัสจำลองของชุดคำสั่งที่ 1 : ลากเส้น

### จากจุด

(C,2) → (A,4) → (A,7) → (B,7) →  
(B,5) → (C,5) → (C,9) → (D,9) →  
(D,7) → (G,7) → (G,9) → (H,9) →  
(H,6) → (I,7) → (I,6) → (H,5) → (H,2)  
→ (C,2)

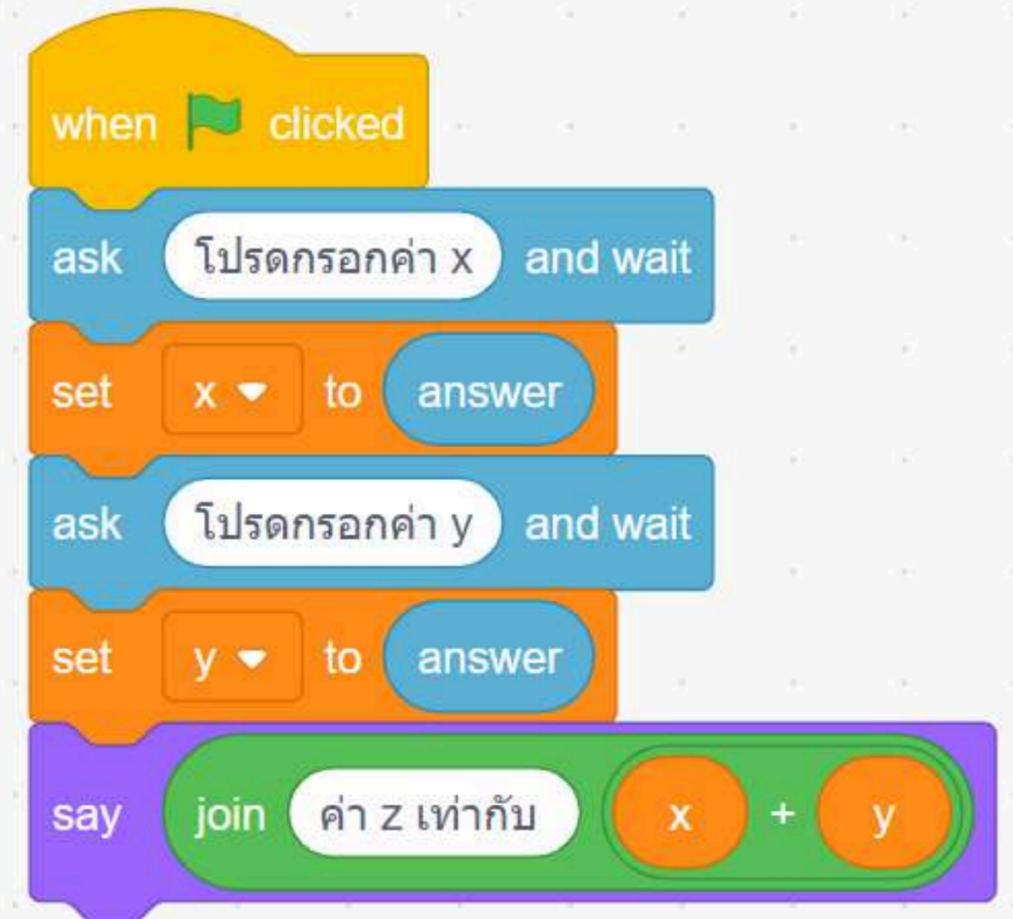
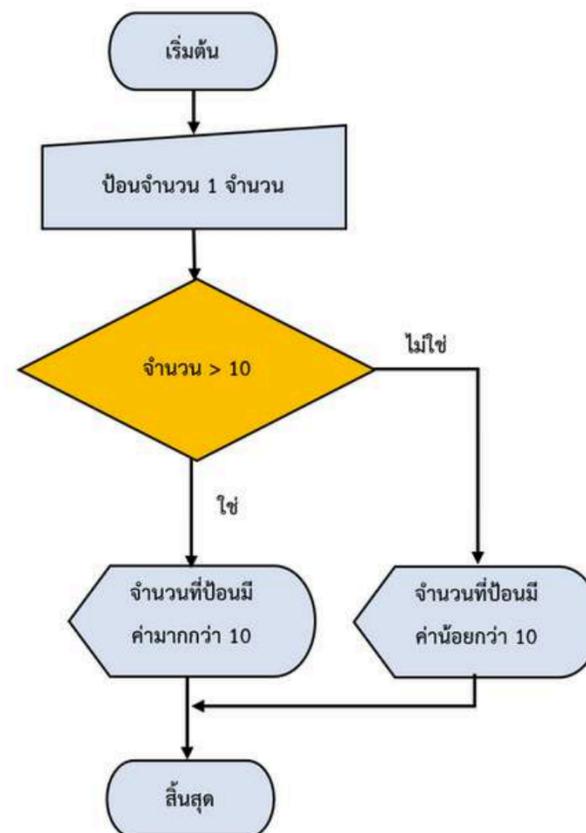
## รหัสจำลองของชุดคำสั่งที่ 2 : ลากเส้น

### จากจุด

(D,5) → (E,6) → (F,6) → (F,3) →  
(E,3) → (D,4)

แสดงผล: ..... ? .....

วัตถุประสงค์: ป้อนจำนวนด้วยแป้นพิมพ์และอยากรทราบว่าจำนวนที่ป้อนเข้ามามากกว่า 10 หรือไม่



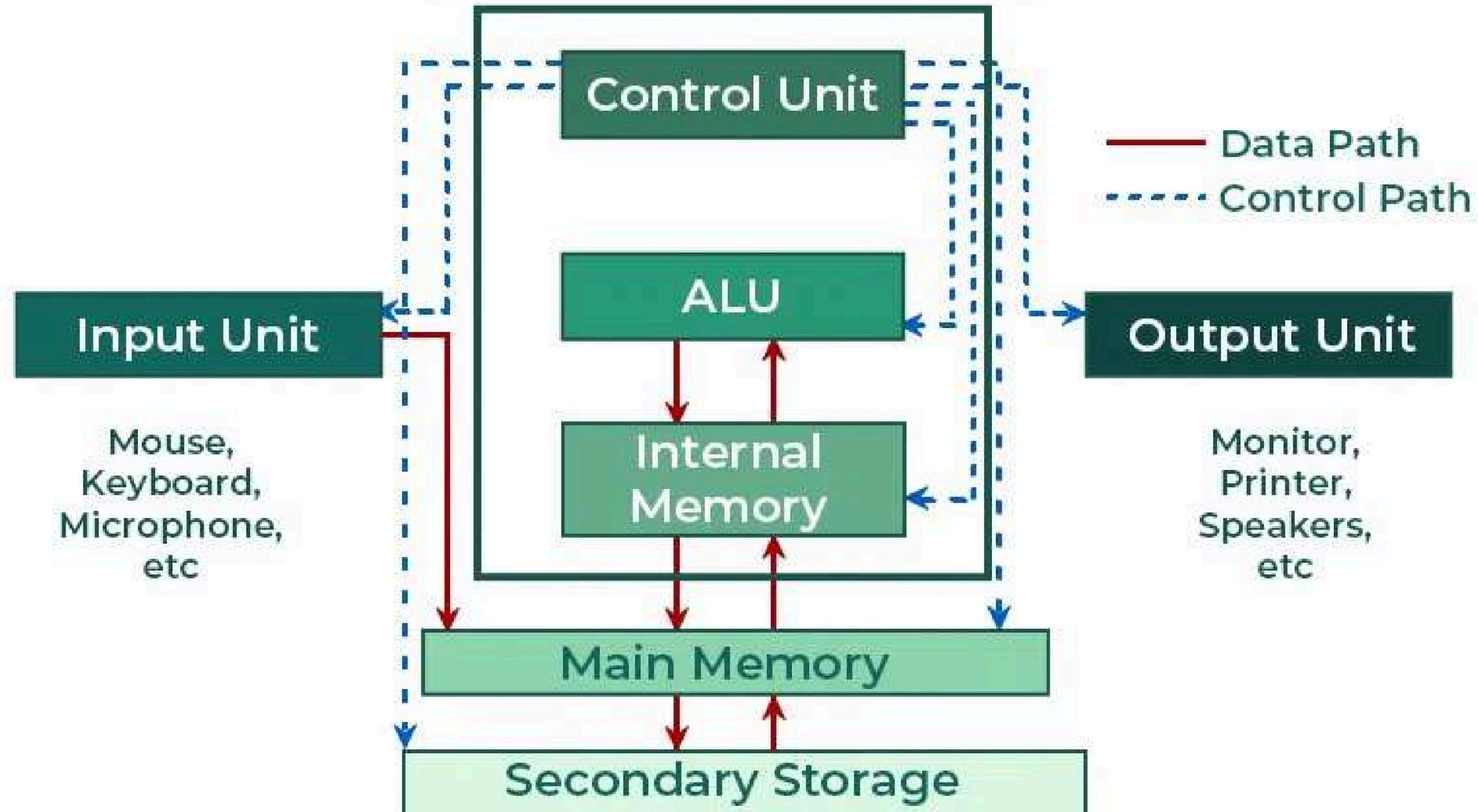
# Computer

## Computer System

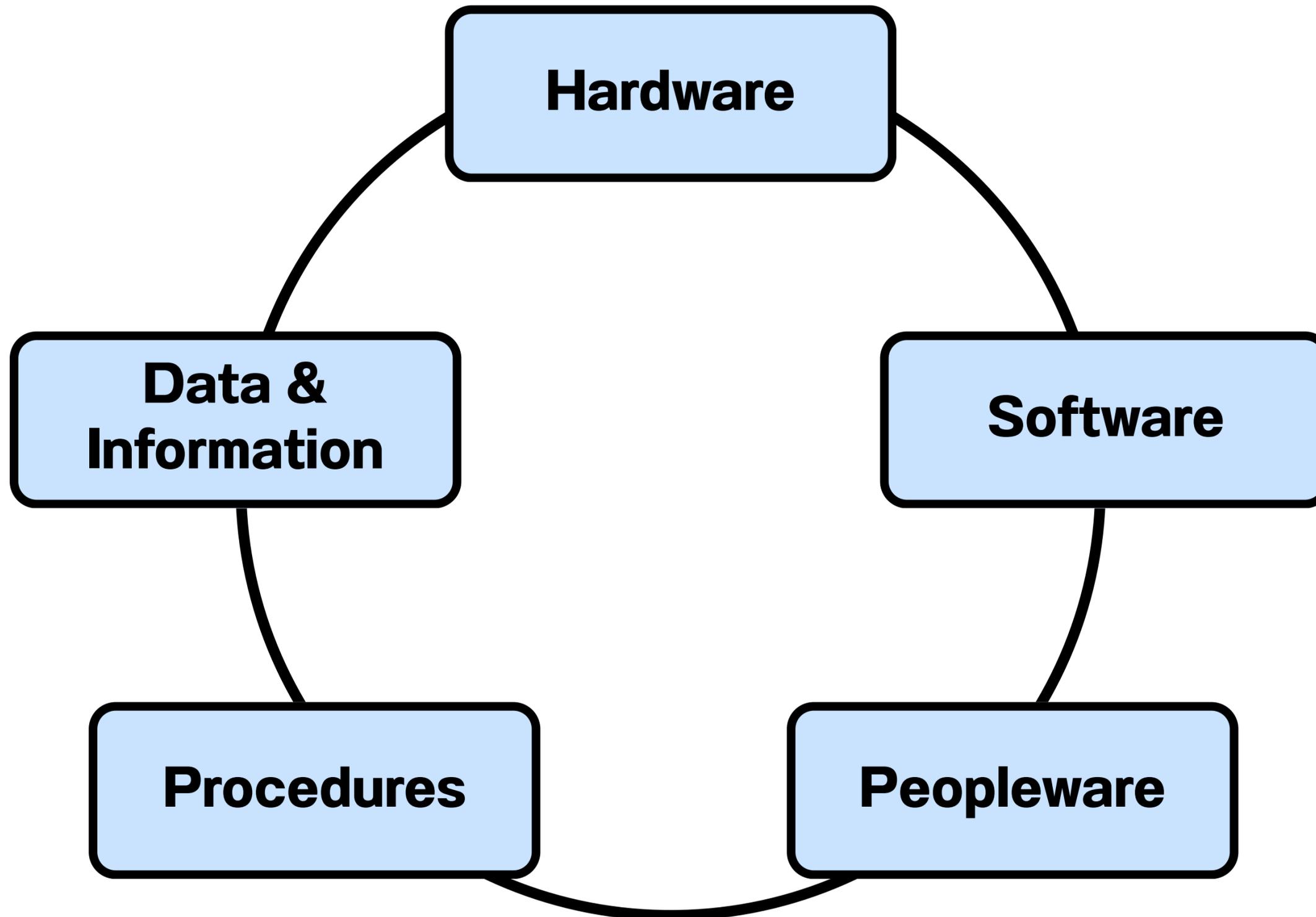


# Computer Components

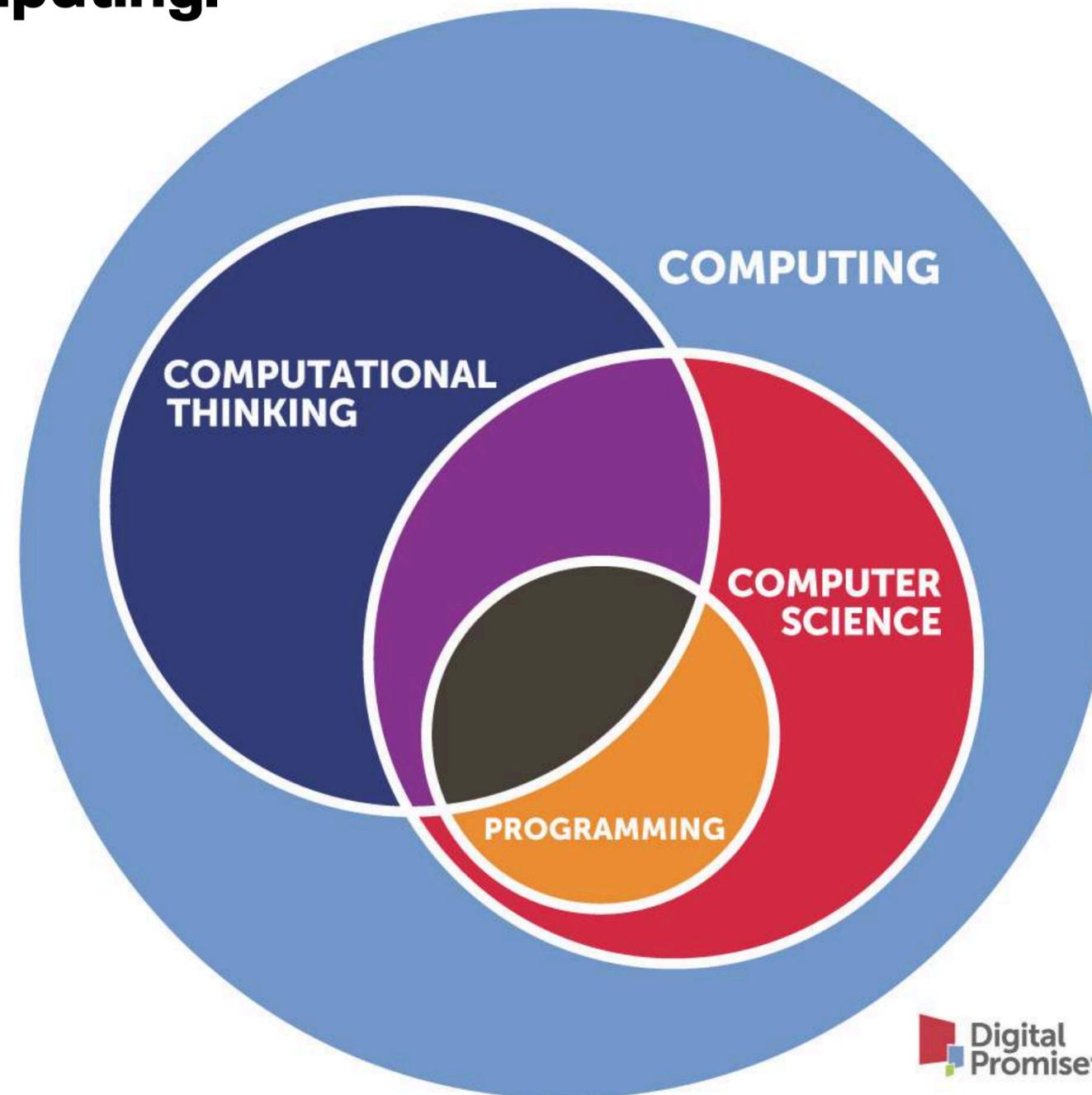
## Components of computer



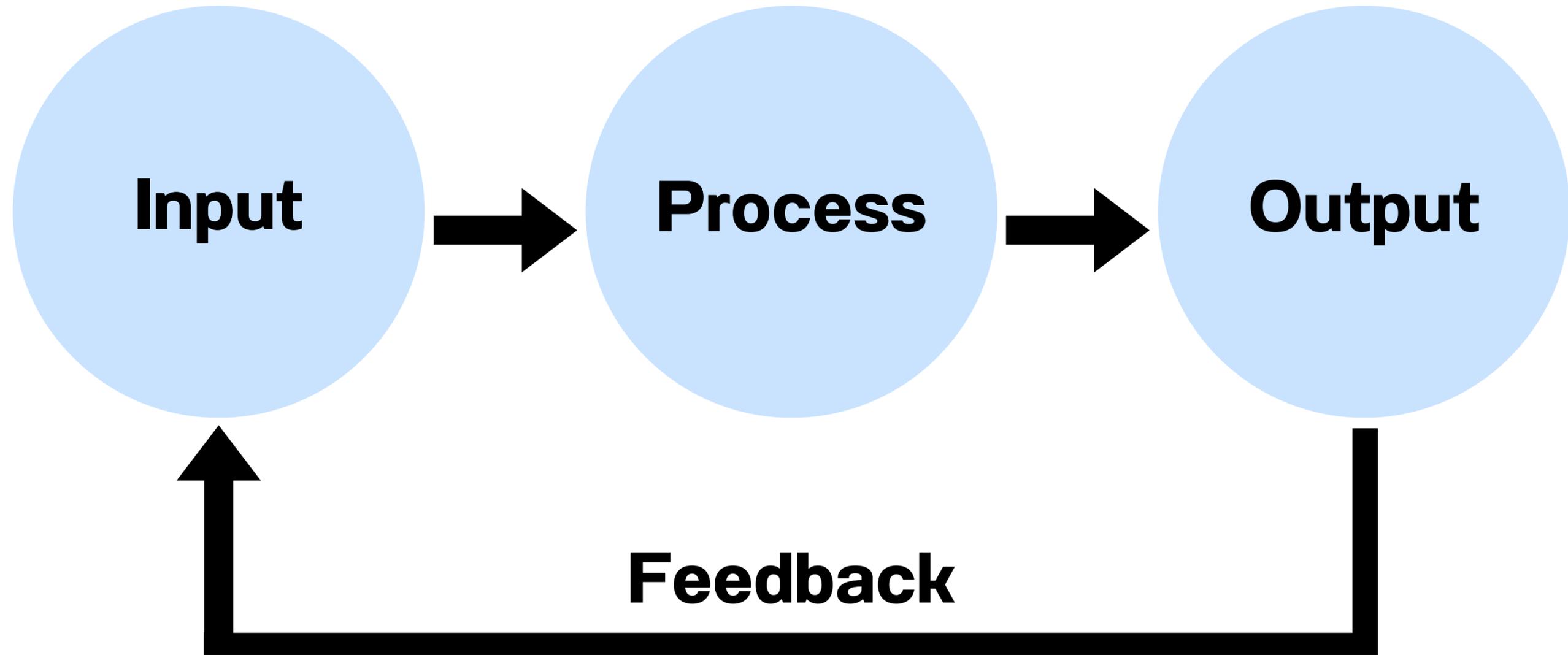
# Computer Components



**The relationship between computer science (CS), computational thinking (CT), programming and computing.**

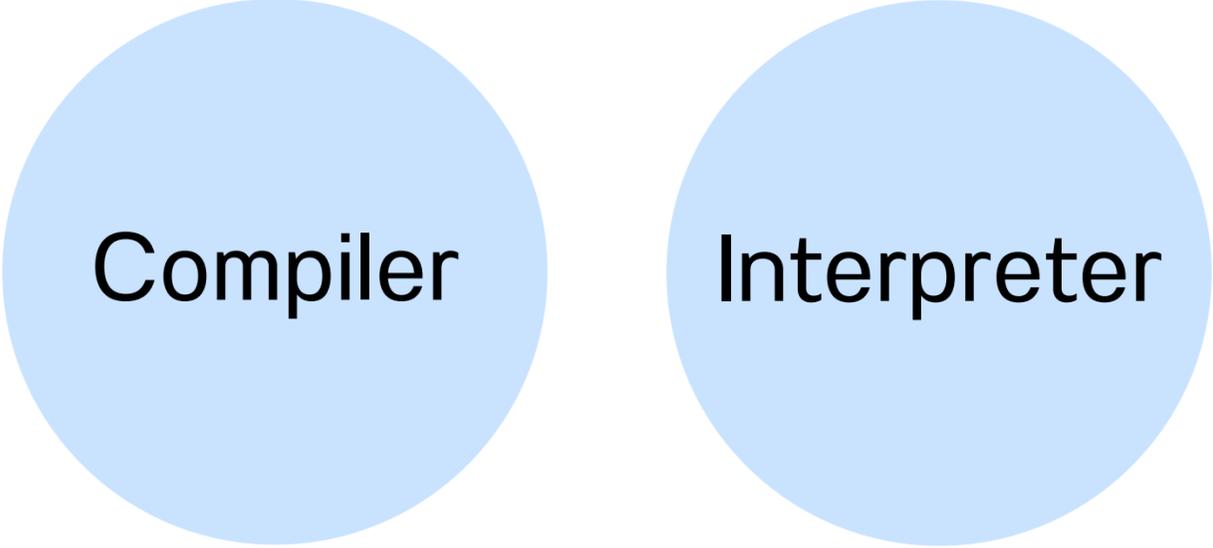


# System Theory



## Compiler & Interpreter

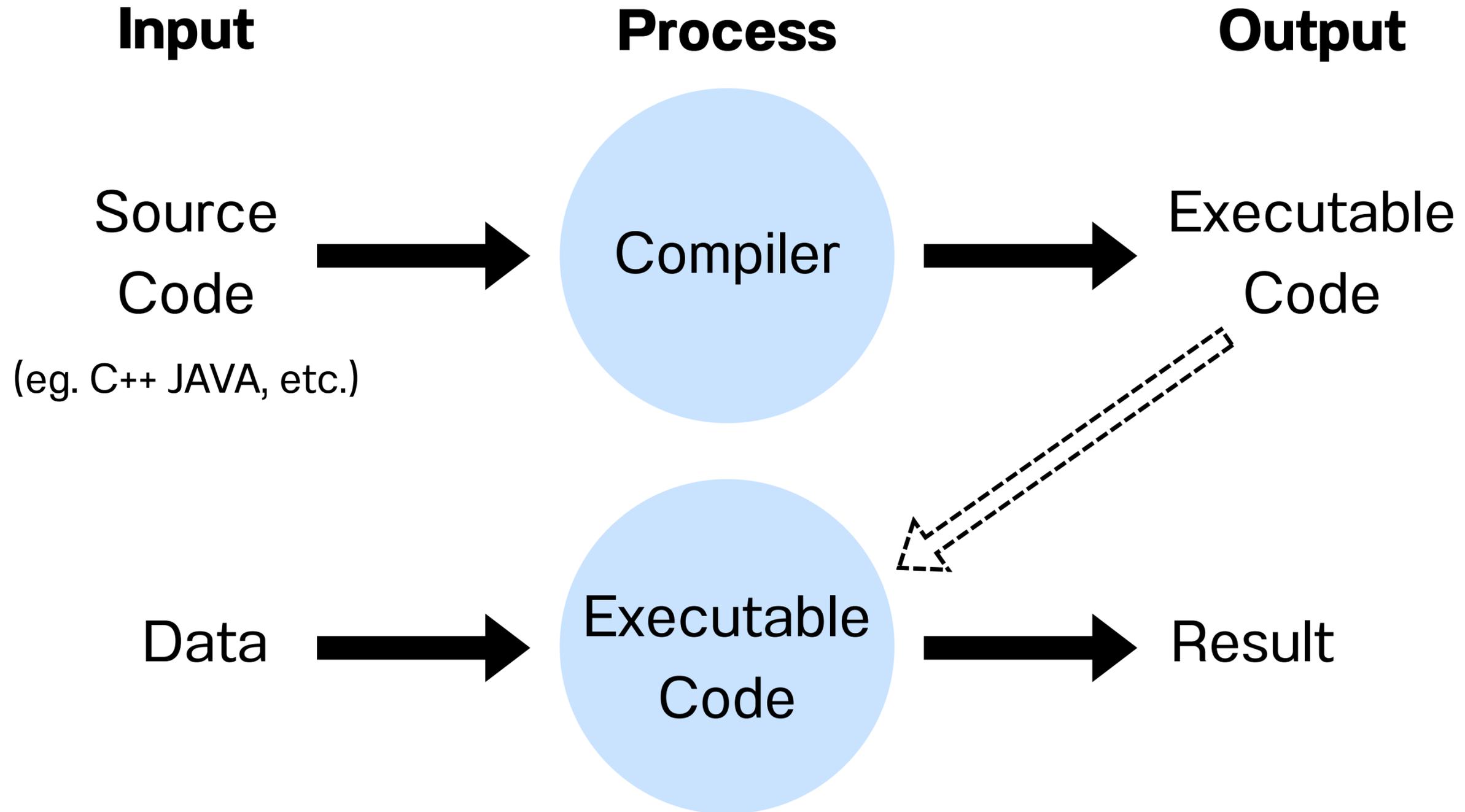
- ในภาษาคอมพิวเตอร์หลากหลายภาษา จะมีกระบวนการแปลภาษาคอมพิวเตอร์อยู่ 2 แบบ ได้แก่



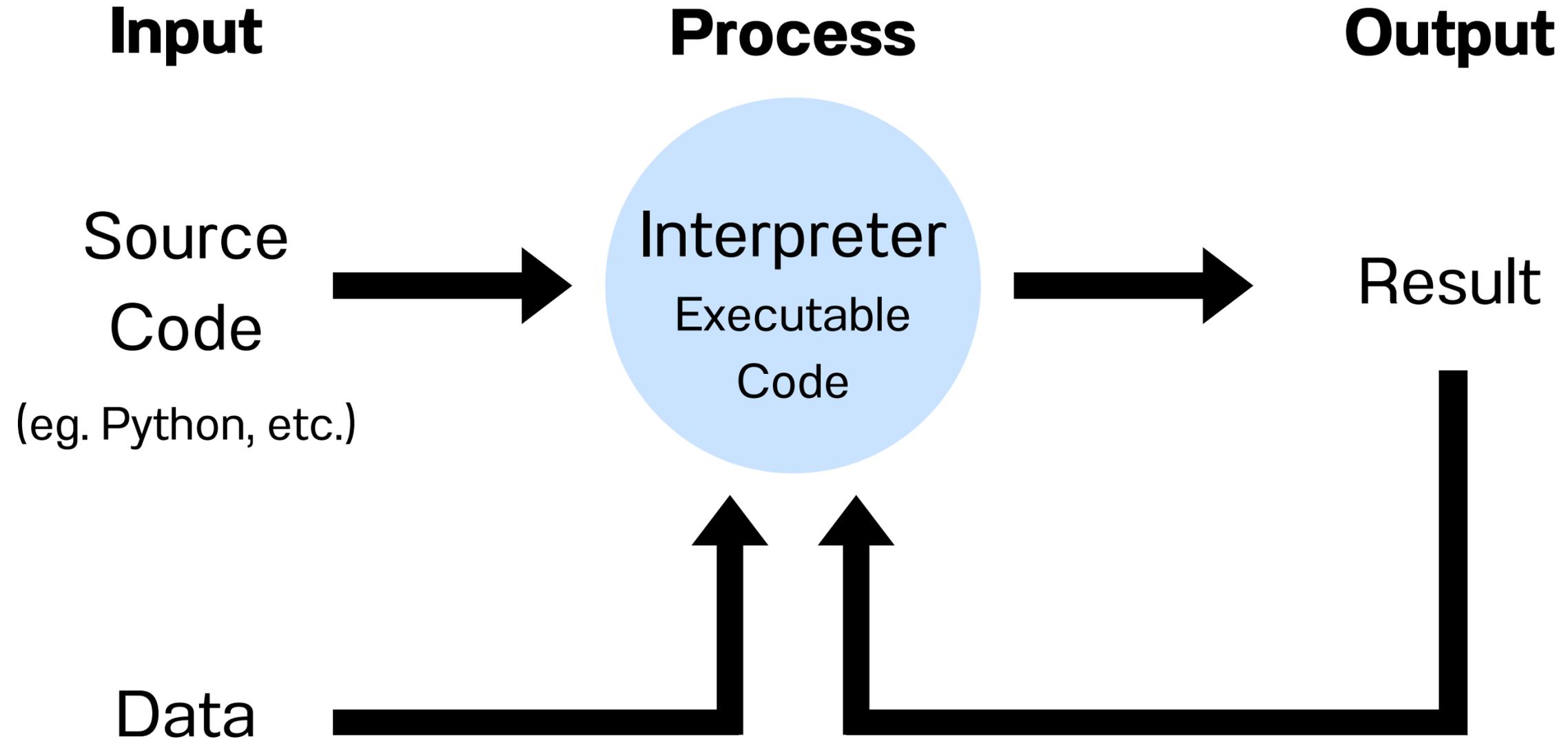
Compiler

Interpreter

# Compiler

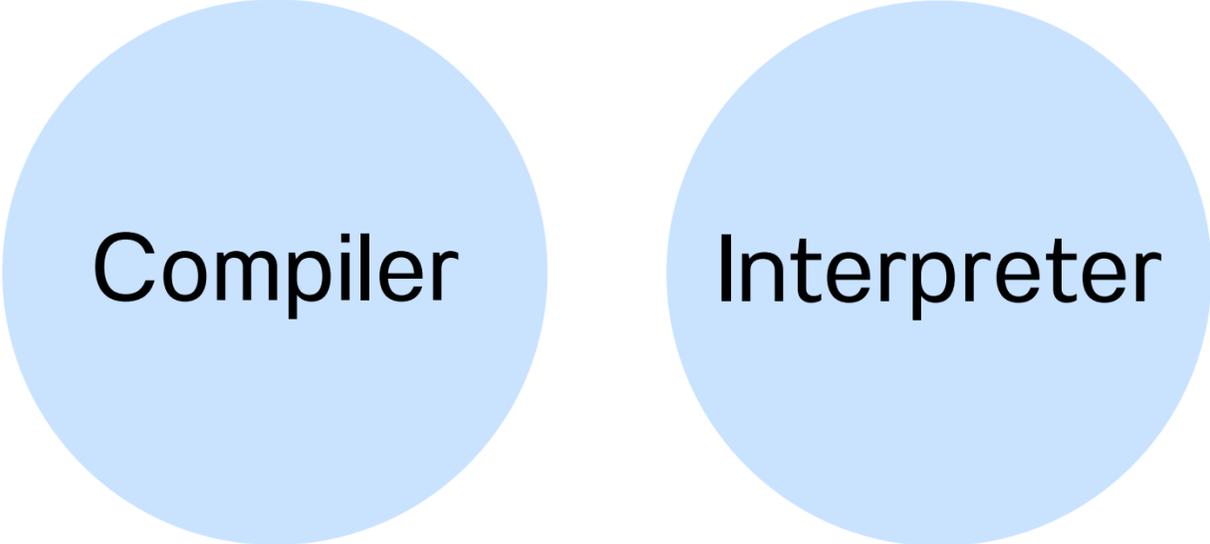


# Interpreter - Line Interpretation



## Compiler & Interpreter

- Compiler จะทำงานได้ไวกว่า Interpreter แต่หากมีโค้ดที่ Error แม้แต่จุดเดียว จะไม่สามารถทำงานได้
- Interpreter จะทำงานได้ช้ากว่า Compiler แต่จะสามารถทำงานได้ แม้จะมีโค้ดที่ Error ก็จะทำงานไปจนกว่าจะถึงจุดที่ Error



Compiler

Interpreter

# Difference Between High-Level Language & Low-Level Language

## High-Level Language

Python, C, C++, C# Language

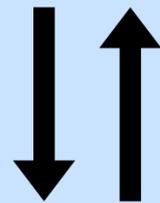
Procedural Language

Object Oriented: OO

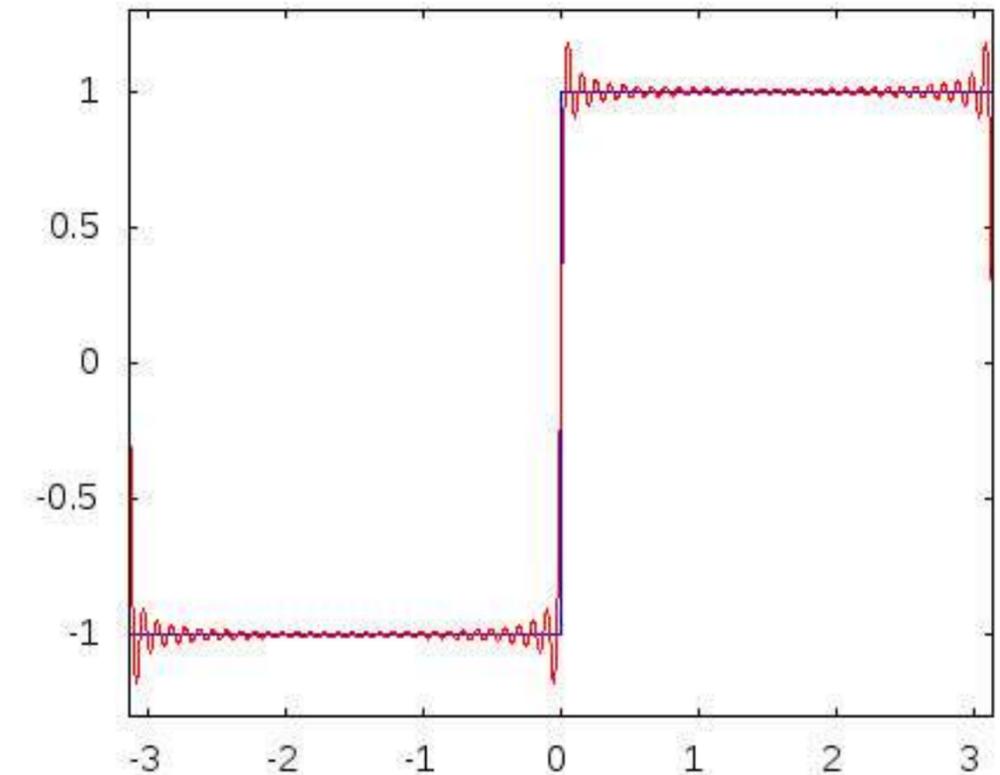
## Low-Level Language

Assembly Language

Machine Language



Hardware



# Difference Between High-Level Language & Low-Level Language

## High-Level Language

Python, C, C++, C# Language

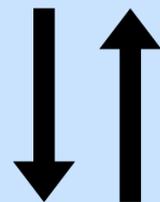
Procedural Language

Object Oriented: OO

## Low-Level Language

Assembly Language

Machine Language



Hardware

1001001001010111101010101



# Difference Between High-Level Language & Low-Level Language

## High-Level Language

Python, C, C++, C# Language

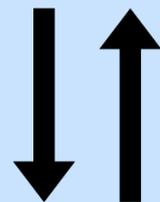
Procedural Language

Object Oriented: OO

## Low-Level Language

Assembly Language

Machine Language



Hardware

```
mov ecx, ebx  
mov esp, edx  
mov edx, r9d  
mov rax, rdx
```

# Difference Between High-Level Language & Low-Level Language

## High-Level Language

Python, C, C++, C# Language

Procedural Language

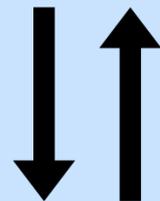
Object Oriented: OO

```
price = 25000
product = "iPad"
print("%s price:%.2f" %(product, price))
```

## Low-Level Language

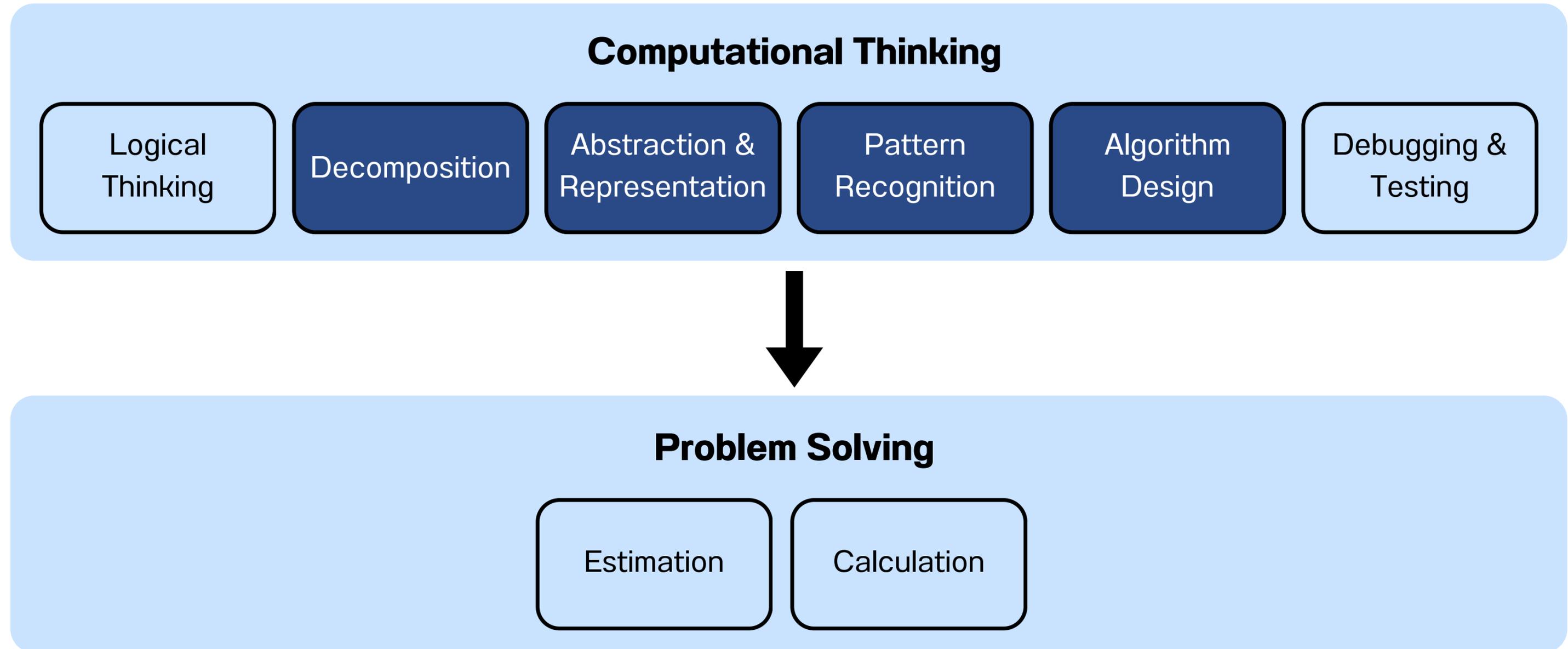
Assembly Language

Machine Language



Hardware

# Computational Thinking & Problem Solving



# Computational Thinking & Problem Solving

## Computational Thinking

Logical Thinking

Decomposition

Abstraction & Representation

Pattern Recognition

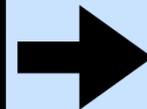
Algorithm Design

Debugging & Testing

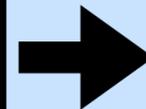


## Problem Solving

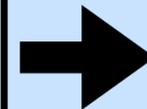
Understanding



Planning



Execution

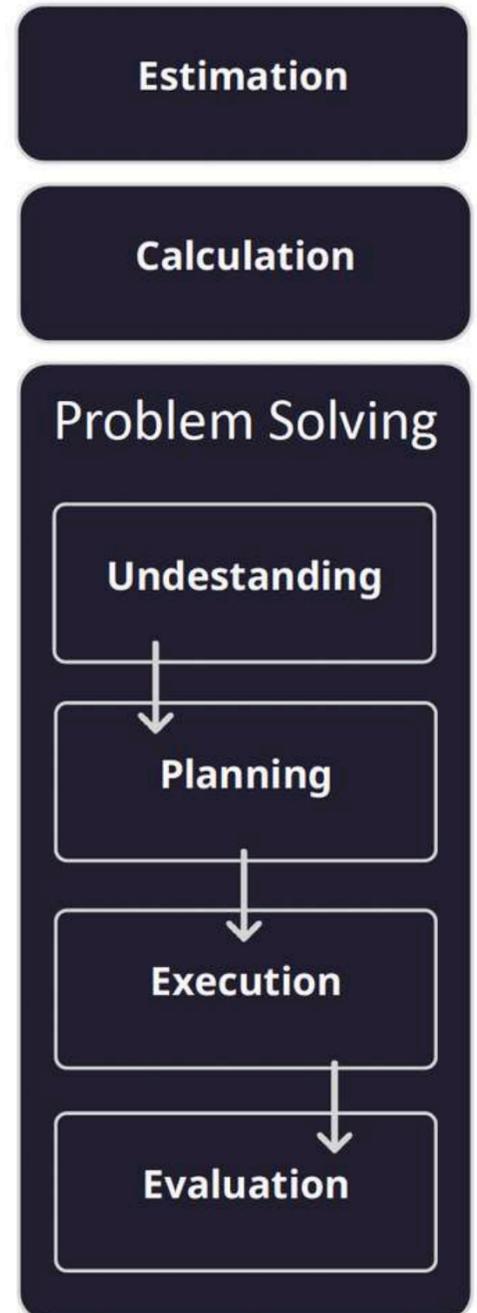
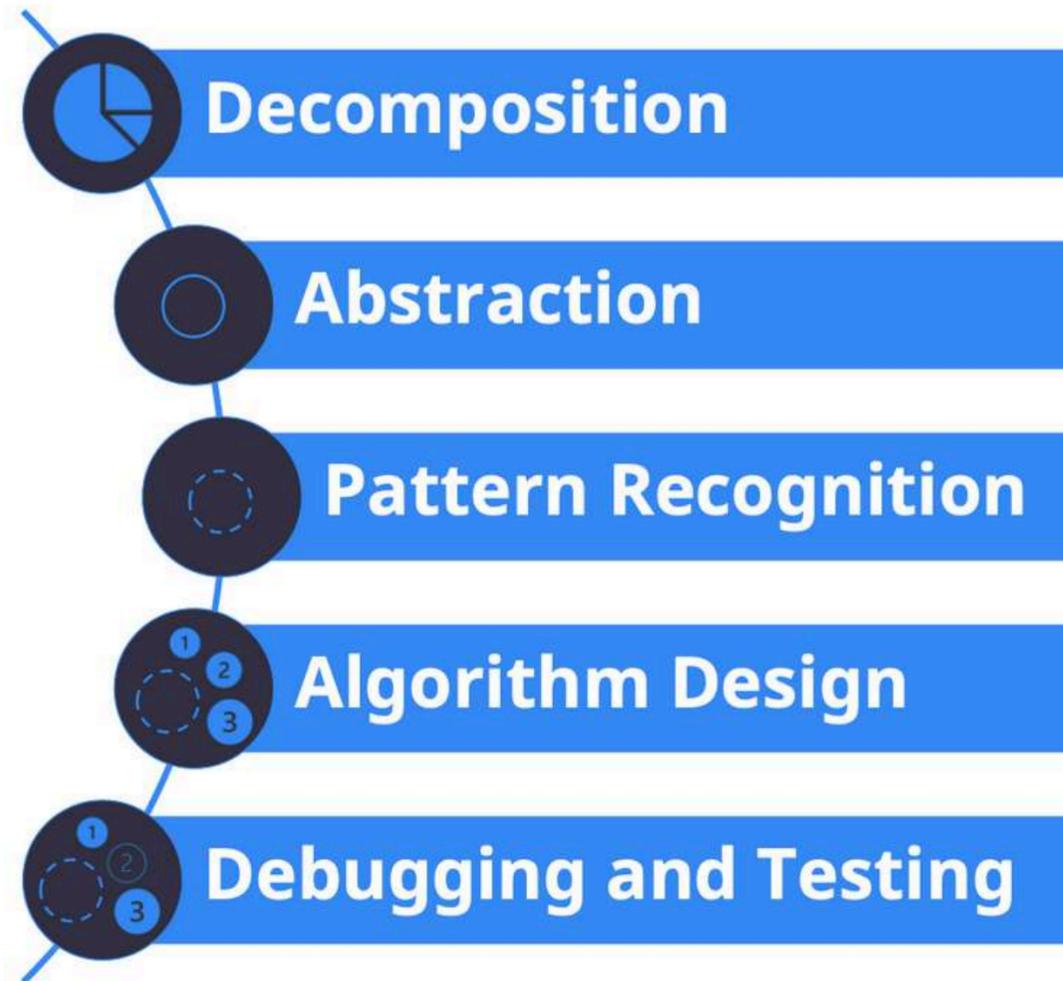


Evaluation

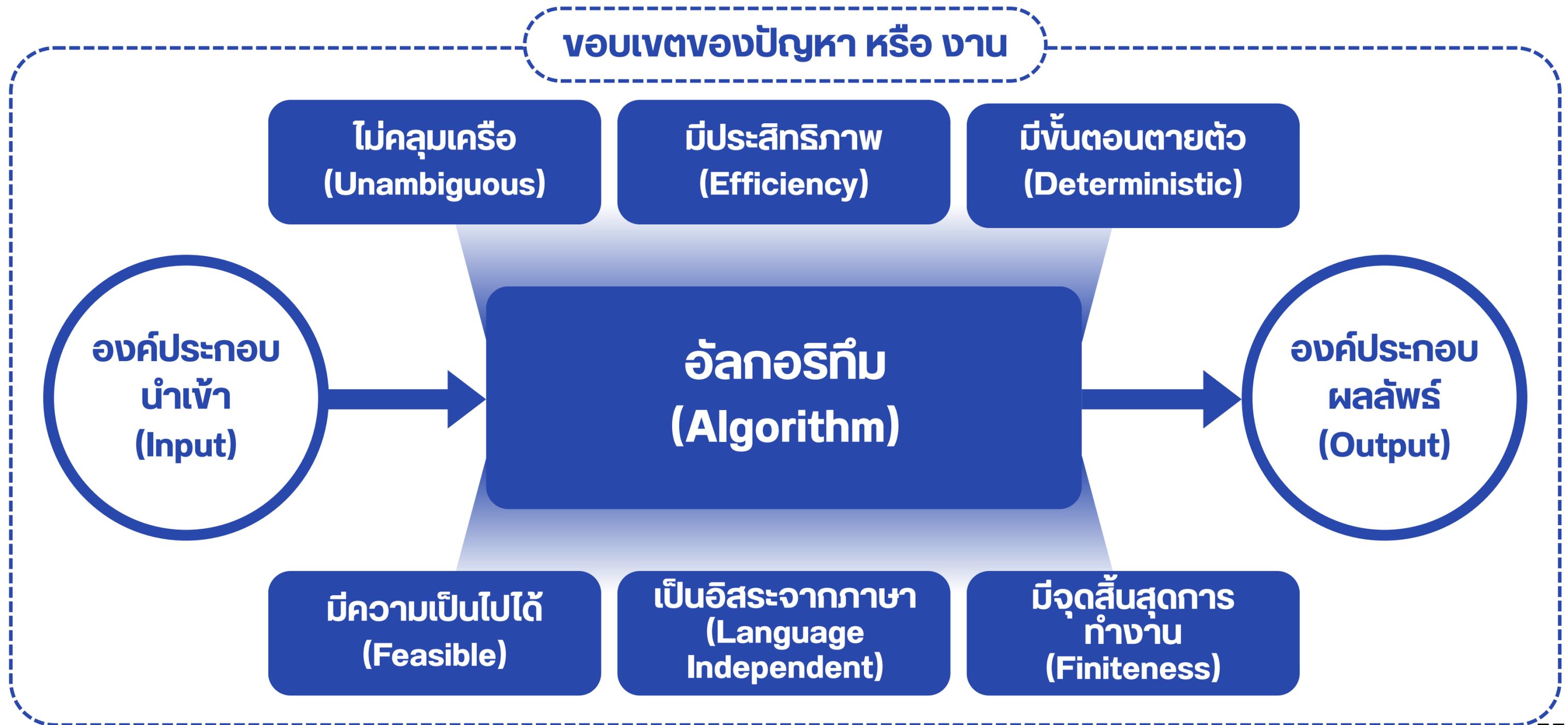
# Python & Computational Thinking

**Computational thinking is a problem solving method that involves**

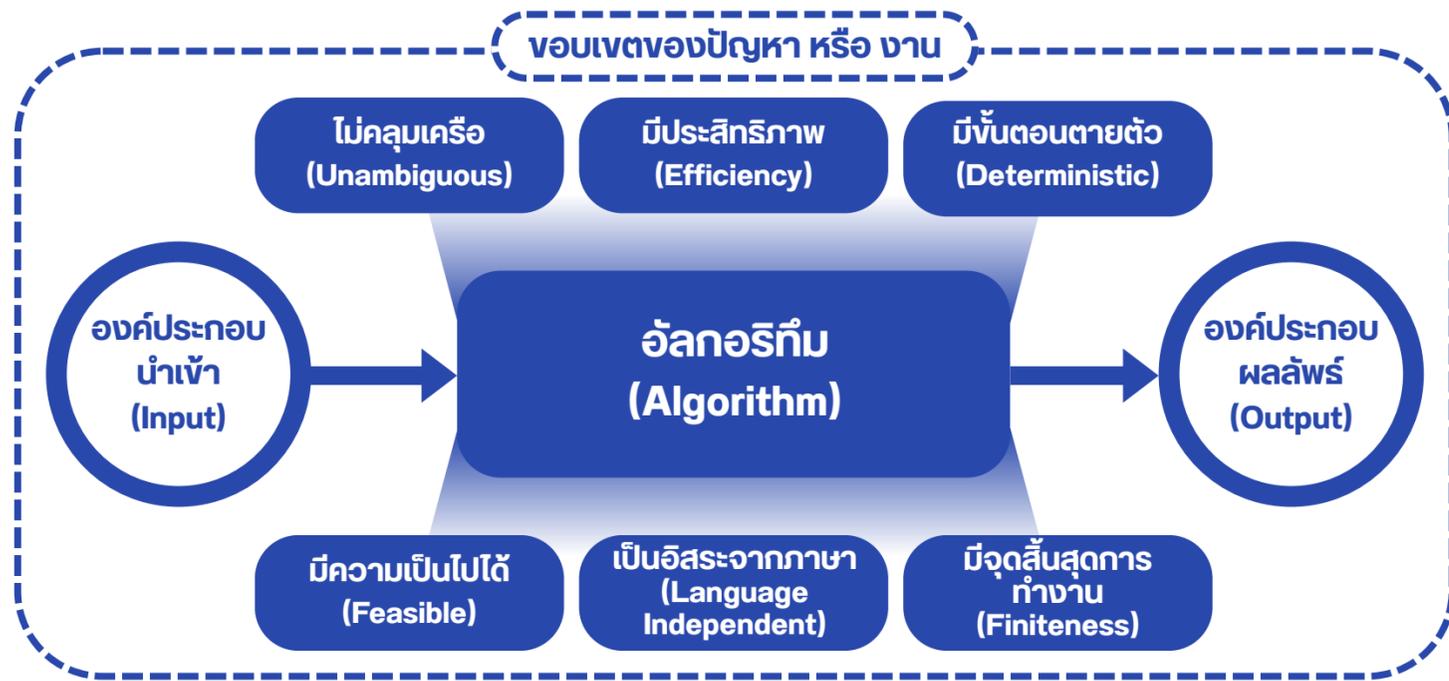
- breaking down complex problems into smaller, more manageable parts, and using logical reasoning and algorithms to solve them.
- Python is a programming language that can be used to implement computational thinking.



# คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



# คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



ลักษณะของอัลกอริทึมที่ดีที่ประกอบด้วยคุณสมบัติ ดังนี้

- 1. ไม่คลุมเครือ (Unambiguous)** อัลกอริทึมต้องไม่มีความกำกวมที่ไม่ชัดเจน และต้องมีชุดคำสั่งที่เข้าใจง่ายและไม่ทำให้เกิดความสับสนในการประมวลผล
- 2. มีประสิทธิภาพ (Efficiency)** อัลกอริทึมควรมีประสิทธิภาพในการทำงาน คือต้องให้ผลลัพธ์ที่ถูกต้อง และใช้เวลาน้อยที่สุดในการแก้ปัญหา

- 3. มีขั้นตอนตายตัว (Deterministic)** อัลกอริทึมต้องเป็น Deterministic หมายความว่า จะต้องให้ผลลัพธ์ที่เหมือนกันสำหรับข้อมูลนำเข้าที่เหมือนกันทุกครั้งที่รับ
- 4. มีความเป็นไปได้ (Feasible)** อัลกอริทึมต้องเป็นไปได้ในการนำไปใช้งานจริง สามารถทำงานบนเครื่องคอมพิวเตอร์หรือสภาพแวดล้อมที่เหมาะสม
- 5. เป็นอิสระจากภาษา (Language Independent)** อัลกอริทึมควรมองไม่เกี่ยวข้องกับภาษาเฉพาะเสียเพียงอย่างเดียว และสามารถนำไปใช้งานในหลายภาษาโปรแกรมได้
- 6. มีจุดสิ้นสุดการทำงาน (Finiteness)** อัลกอริทึมต้องสามารถสิ้นสุดการทำงานหลังจากทำขั้นตอนทุกขั้นตอนเสร็จสมบูรณ์ และไม่ต้องเข้าสู่วงจรไม่สิ้นสุด (Infinity Loop)

## อ้างอิง

- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press. ISBN: 978-0262033848.
- Sedgewick, R., & Wayne, K. (2011). Algorithms (4th ed.). Addison-Wesley Professional. ISBN: 978-0321573513.

# คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)

ขอบเขตของปัญหา หรือ งาน  
คำนวณเงินทอน

ไม่คลุมเครือ  
(Unambiguous)

มีประสิทธิภาพ  
(Efficiency)

มีขั้นตอนตายตัว  
(Deterministic)

A  
ราคาสินค้า  
150 บาท

B  
จ่ายเงิน  
500 บาท

เงินทอน = จ่ายเงิน - ราคาสินค้า

$$C = B - A$$

C  
เงินทอน  
350 บาท

มีความเป็นไปได้  
(Feasible)

เป็นอิสระจากภาษา  
(Language Independent)

มีจุดสิ้นสุดการทำงาน  
(Finiteness)

# คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)

ขอบเขตของปัญหา หรือ งาน  
คำนวณเงินทอน

ไม่คลุมเครือ  
(Unambiguous)

มีประสิทธิภาพ  
(Efficiency)

มีขั้นตอนตายตัว  
(Deterministic)

A

ราคาสินค้า  
150 บาท

B

จ่ายเงิน  
1,000 บาท

เงินทอน = จ่ายเงิน - ราคาสินค้า

$$C = B - A$$

C

เงินทอน  
850 บาท

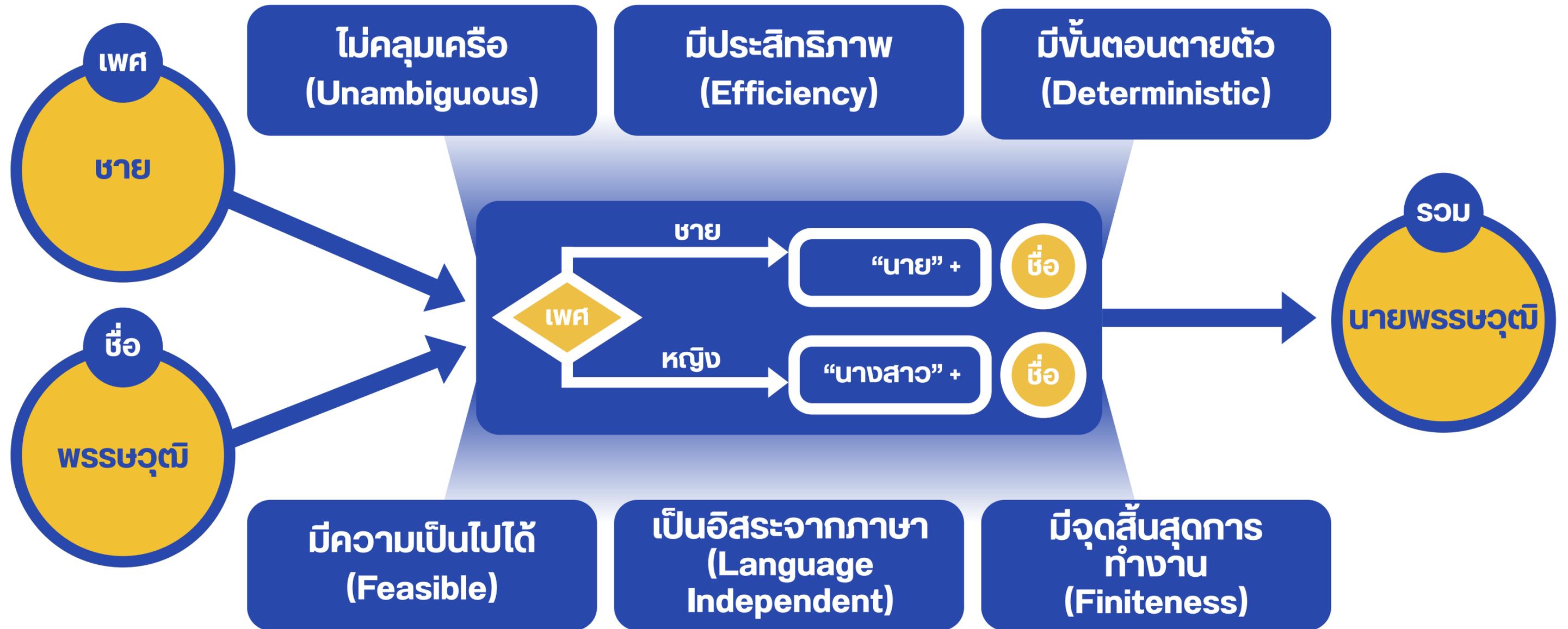
มีความเป็นไปได้  
(Feasible)

เป็นอิสระจากภาษา  
(Language Independent)

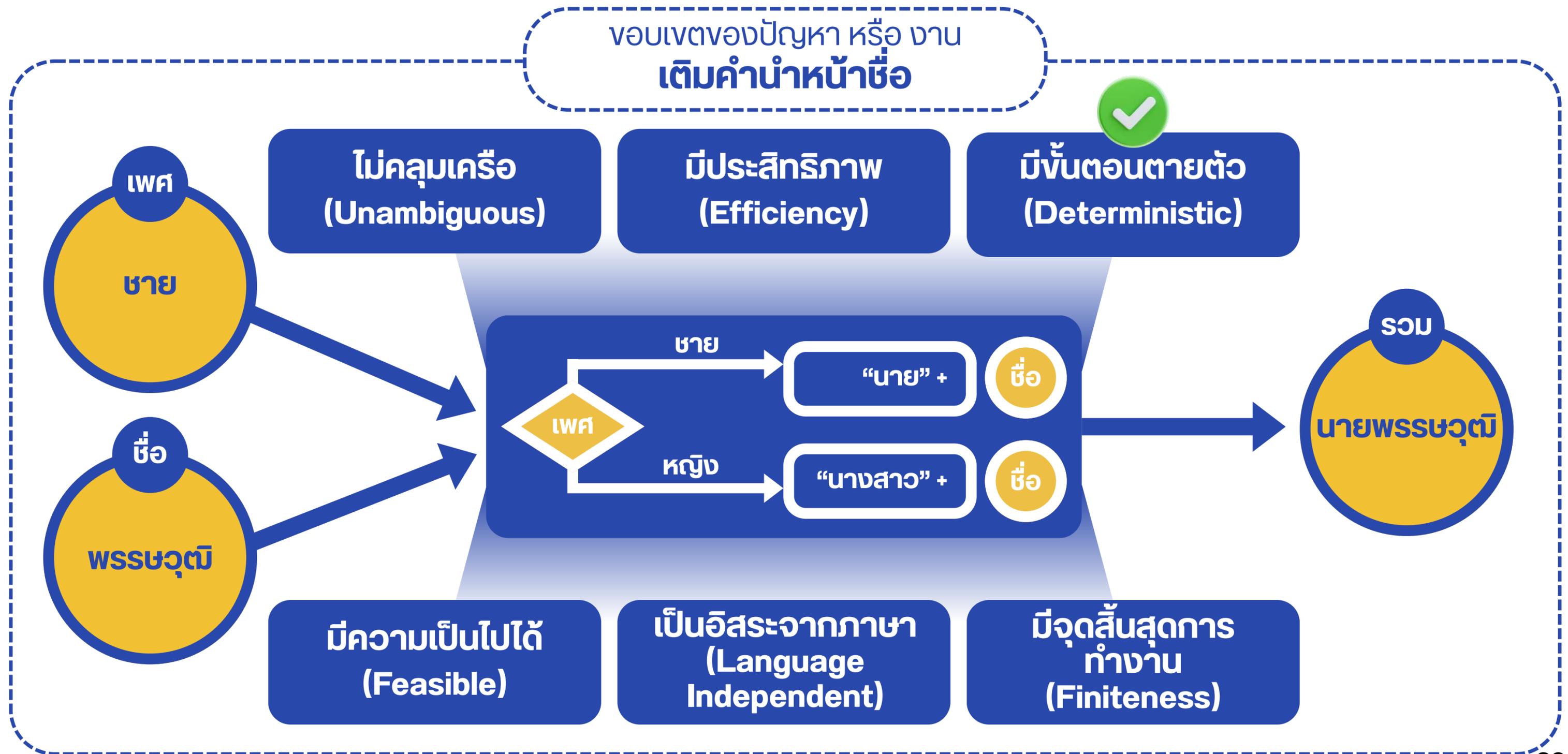
มีจุดสิ้นสุดการ  
ทำงาน  
(Finiteness)

# คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)

ขอบเขตของปัญหา หรือ งาน  
เต็มคำนำหน้าชื่อ



# ให้ผลลัพธ์ถูกต้องตามเงื่อนไขที่กำหนดไว้เสมอ คือ คุณลักษณะข้อใด



# การสอน Coding สำหรับผู้เริ่มต้น

1. Input  
การนำเข้า

3. Process  
การประมวลผล

2. Output  
ผลผลิต/ผลลัพธ์

Operation Variable ตัวแปรดำเนินการ

# ต้องการผลรวมของค่า 2 จำนวน

1. Input  
การนำเข้า

จำนวนที่ 1

จำนวนที่ 2

3. Process  
การประมวลผล

Operation Variable ตัวแปรดำเนินการ

2. Output  
ผลผลิต

# ต้องการผลรวมของค่า 2 จำนวน

1. Input  
การนำเข้า

จำนวนที่ 1

จำนวนที่ 2

3. Process  
การประมวลผล

Operation Variable ตัวแปรดำเนินการ

2. Output  
ผลลัพธ์

ผลลัพธ์

# ต้องการผลรวมของค่า 2 จำนวน

1. Input  
การนำเข้า

จำนวนที่ 1

จำนวนที่ 2

3. Process  
การประมวลผล

Operation Variable ตัวแปรดำเนินการ

1.

ผลลัพธ์



จำนวนที่ 1

+

จำนวนที่ 2

2.

แสดง

ผลลัพธ์

2. Output  
ผลลัพธ์

ผลลัพธ์

# ต้องการคำนวณส่วนลดราคาสินค้า

ราคาสินค้า 200 บาท ส่วนลด 10% ต้องจ่ายที่บาท

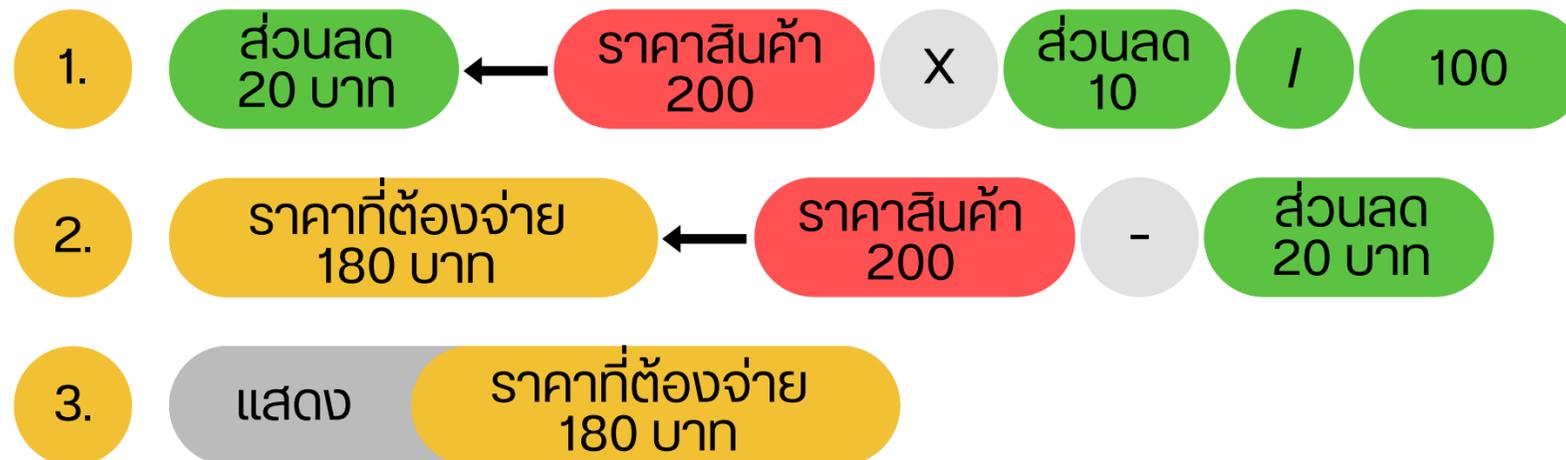
1. Input  
การนำเข้า

ราคาสินค้า

ส่วนลด

3. Process  
การประมวลผล

Operation Variable ตัวแปรดำเนินการ



ตัวแปร (Variable) เช่น ราคาสินค้า ส่วนลด ส่วน 50, 100 เป็นค่าของตัวแปร (Value)

2. Output  
ผลลัพธ์

ราคาที่ต้องจ่าย

# Python Programming Steps Using Google Colab

1. Go to **<https://colab.research.google.com/>**
2. Log in with your **SSRU gmail**
3. File -> New Notebook
4. Coding with python commands on cell
5. Run cell

# 1. Go to <https://colab.research.google.com/>

The screenshot shows the Google Colaboratory (Colab) website interface. At the top, the browser address bar displays <https://colab.research.google.com/>. The main header features the Colab logo (two overlapping orange circles), the text "Welcome To Colaboratory", and a navigation menu with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". On the right side of the header, there are links for "Share", a settings gear icon, and a blue "Sign in" button.

Below the header, the interface is divided into several sections. On the left, there is a "Table of contents" sidebar with a search icon and a list of items: "Getting started", "Data science", "Machine learning" (with a "{x}" icon), "More Resources", "Featured examples", and a "+ Section" button. The main content area has a toolbar with options: "+ Code", "+ Text", "Copy to Drive" (with a drive icon), "Connect" (with a dropdown arrow), "Editing" (with a pencil icon), and an upward arrow icon. Below the toolbar, the main content area displays a large heading "Welcome to Colab!" followed by a paragraph: "If you're already familiar with Colab, check out this video to learn about interacting with the Colab interface, the executed code history view, and the command palette." Below this text is a video player thumbnail with the title "3 Cool Google Colab Features" and a play button icon. The video thumbnail also shows a person's face and the Colab logo.

## 2. Log in with your SSRU gmail

Click Here

https://colab.research.google.com/

Welcome To Colaboratory

File Edit View Insert Runtime Tools Help

Share Settings Sign in

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Section

Welcome to Colab!

If you're already familiar with Colab, check out this video to learn about interactive code execution, the executed code history view, and the command palette.

3 Cool Google Colab Features

### 3. File -> New Notebook

The image shows a browser window at <https://colab.research.google.com/>. The page title is "Welcome To Colaboratory". The "File" menu is open, showing options: "New notebook", "Open notebook", "Upload notebook", "Rename", "Save a copy in Drive", "Save a copy as a GitHub Gist", "Save a copy in GitHub", "Save", "Revision history", "Download", and "Print". A red arrow labeled "1" points to the menu icon on the left. Another red arrow labeled "2" points to the "New notebook" option in the menu. The main content area displays "Welcome to Colab!" and a video thumbnail titled "3 Cool Google Colab Features".

## 4. Coding with python commands on cell

The image shows a screenshot of a Google Colab notebook interface. The browser address bar at the top displays `https://colab.research.google.com/`. The notebook title is `chapter1_example1.ipynb`. The menu bar includes `File`, `Edit`, `View`, `Insert`, `Runtime`, `Tools`, and `Help`, along with the status `All changes saved`. On the right side, there are buttons for `Comment`, `Share`, and a user profile icon. The main workspace shows a code cell with the following Python code:

```
1 print("Hello World!")
```

A red arrow points to the closing parenthesis of the `print` statement, with the text **Rigth Here** written below it. The interface also shows a left sidebar with icons for a menu, search, and a variable `{x}`. The top right of the code cell has a toolbar with icons for undo, redo, link, comment, settings, copy, and delete.

## 5. Run cell

The screenshot shows a Google Colab notebook interface. The browser address bar displays `https://colab.research.google.com/`. The notebook title is `chapter1_example1.ipynb`. The top navigation bar includes `File`, `Edit`, `View`, `Insert`, `Runtime`, `Tools`, and `Help`, along with `Comment`, `Share`, and a user profile icon. The `Runtime` section shows `RAM` and `Disk` usage indicators, and the `Editing` mode is active. The main workspace contains a code cell with the following content:

```
1 print("Hello World!")
```

The output of the cell is `Hello World!`. A red arrow points to the play button icon on the left side of the code cell, with the text `Click Here` written in red below it. A green checkmark icon is visible at the bottom right of the code cell, indicating successful execution.

# Activity: Type the Python command as shown in the picture and observe the results.

https://colab.research.google.com/

chapter1\_example1.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

[1] ✓ 0s 1 print("Hello World!")

{x} Hello World!

✓ 0s [▶] 1 a = "Python"  
2 b = "course"  
3 c = "is awesome"  
4 print(a,b,c)

2 RUN

1 CODE



# Post Test

**Question:**

**1. ภาษา Python เป็นภาษาคอมไพเลอร์ลักษณะใด?**

**A**

Interpreter

**B**

Compiler

**Question:**

## 2. ข้อใดไม่เป็นองค์ประกอบของระบบคอมพิวเตอร์

**A** Hardware

**B** Software

**C** Peopleware

**D** Data/Information

**E** Communication

**F** Procedures

**Question:**

**3. "110100101010101" เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?**

**A** Low-Level Language

**C** Assembly Language

**B** High-Level Language

**D** Procedural Language

**Question:**

**4. print("Panda") เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?**

**A** Low-Level Language

**C** Assembly Language

**B** High-Level Language

**D** Procedural Language

**Question:**

**5. ข้อใดไม่ใช่คุณลักษณะของอัลกอริทึม**

**A**

ไม่คลุมเครือ  
(Unambiguous)

**C**

มีจุดสิ้นสุดการทำงาน  
(Finiteness)

**B**

มีขั้นตอนตายตัว  
(Deterministic)

**D**

มีหน่วยความจำที่เพียงพอ  
(Memory Capacity)

**Question:**

## 6. ข้อใดไม่ใช่เครื่องมือในการโปรแกรมภาษา Python

**A** Pycham

**C** Jupyter

**B** Colab

**D** Assembler

**Question:**

**7. ให้ผลลัพธ์ถูกต้องตามเงื่อนไขที่กำหนดไว้เสมอ คือ คุณลักษณะข้อใด?**

**A**

ไม่คลุมเครือ  
(Unambiguous)

**C**

มีจุดสิ้นสุดการทำงาน  
(Finiteness)

**B**

มีขั้นตอนตายตัว  
(Deterministic)

**D**

มีหน่วยความจำที่เพียงพอ  
(Memory Capacity)

**Question:**

## 8. วิธีการสอนใดเหมาะกับการเรียนรู้ด้านการโปรแกรมคอมพิวเตอร์

**A**

Passivity-based  
Learning

**C**

Experiential Learning

**B**

Cognitivism

**D**

Adaptive Learning

**Question:**

**9. การสอน Coding สิ่งแรกที่ต้องสอนให้ผู้เรียนคำนึงถึง  
ในการโปรแกรม คืออะไร?**

**A**

Input

**C**

Output

**B**

Process

**D**

Memory

**Question:**

**10. การสอน Coding สิ่งสุดท้ายที่ต้องสอนให้ผู้เรียนคำนึงถึง  
ในการโปรแกรม คืออะไร?**

**A**

Input

**C**

Output

**B**

Process

**D**

Memory

Computer Programming and  
Developing Applications for Education

**Thank You**

**DTI3302 Computer Programming and Developing Applications for Education**

Department of Digital Technology for Education

Faculty of Education, Suan Sunandha Rajabhat University



**Pasawut Cheerapakorn**

Suan Sunandha Rajabhat University