Computer Programming and
Developing Applications for Education

Chapter 1: Computer Programming

Pasawut Cheerapakorn
Suan Sunandha Rajabhat University

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.

Course Description:

หลักการ ทฤษฎี ที่เกี่ยวข้องการเขียนโปรแกรมและพัฒนาแอปพลิเคชัน หลัก การพัฒนาโปรแกรมคอมพิวเตอร์ คุณสมบัติของโปรแกรมภาษาชนิดต่าง ๆ หลักการเบื้องต้นเกี่ยวกับองค์ประกอบ ลักษณะคำสั่ง การเขียนโปรแกรม ขั้นตอนวิธี การวิเคราะห์ การออกแบบ แอปพลิเคชันเพื่อการศึกษา 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. โครงงานประจำภาคเรียน (Term Project)
   - โครงงานและการนำเสนอ 20%

4. การสอบปลายภาค (Final Examination)
   - ปรนัย 35 ข้อ (35 คะแนน) อัตนัย 1 ข้อ (5 คะแนน) 20%

ร้อยละ	ระดับผลการเรียน	ความหมาย
86 – 100	A	ดียอดเยี่ยม
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	**บทที่ 1 การโปรแกรมคอมพิวเตอร์ (Computer Programming)**
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 ทัพเพิล เซ็ต และดิกชันนารี (List, Tupple, and Dictionaries)
8	บทที่ 7 ฟังก์ชั่น (Function)

Measurement and Evaluation:

ครั้งที่ / สัปดาห์	บทเรียน / หัวข้อ
9	สอบกลางภาค (Midterm Examination)
10	บทที่ 7 ฟังก์ชั่น (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. ”1101001010101010101” เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?

A Low-Level Language

B High-Level Language

C Assembly Language

D Procedural Language

Question:

4. print(“Panda”) เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?

A Low-Level Language

B High-Level Language

C Assembly Language

D Procedural Language

Question:

5. ข้อใดไม่ใช่คุณลักษณะของอัลกอริทึม

A ไม่คลุมเครือ (Unambiguous)

B มีขั้นตอนตายตัว (Deterministic)

C มีจุดสิ้นสุดการทำงาน (Finiteness)

D มีหน่วยความจำที่เพียงพอ (Memory Capacity)

Question:

6. ข้อใดไม่ใช่เครื่องมือในการโปรแกรมภาษา Python

A Pycham

B Colab

C Jupyter

D Assembler

Question:

7. ให้ผลลัพธ์ถูกต้องตามเงื่อนไขที่กำหนดไว้เสมอ คือ คุณลักษณะข้อใด?

A ไม่คลุมเครือ (Unambiguous)

B มีขั้นตอนตายตัว (Deterministic)

C มีจุดสิ้นสุดการทำงาน (Finiteness)

D มีหน่วยความจำที่เพียงพอ (Memory Capacity)

Question:

8. วิธีการสอนใดเหมาะกับการเรียนรู้ด้านการโปรแกรมคอมพิวเตอร์

A Passivity-based Learning

B Cognitivism

C Experiential Learning

D Adaptive Learning

Question:

9. การสอน Coding สิ่งแรกที่ต้องสอนให้ผู้เรียนคำนึงถึงในการโปรแกรม คืออะไร?

A Input

C Output

B Process

D Memory

Question:

10. การสอน Coding สิ่งสุดท้ายที่ต้องสอนให้ผู้เรียนคำนึงถึงในการโปรแกรม คืออะไร?

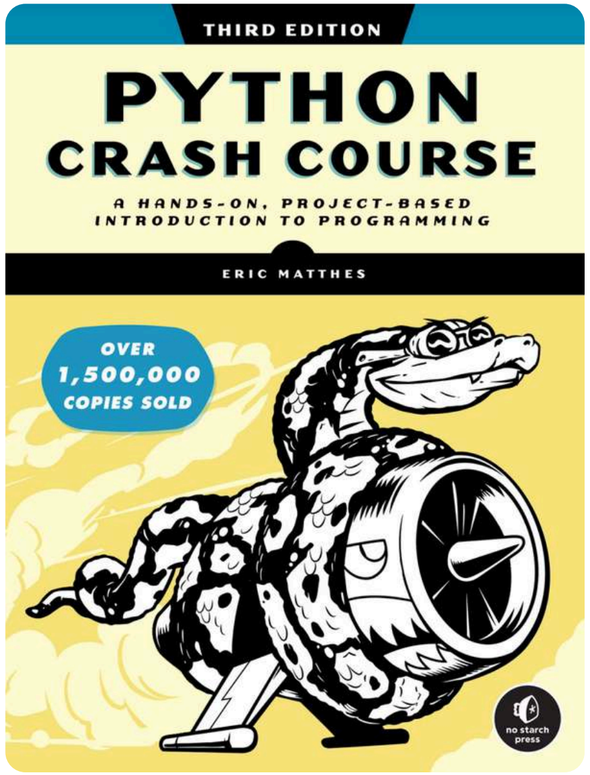
A Input

C Output

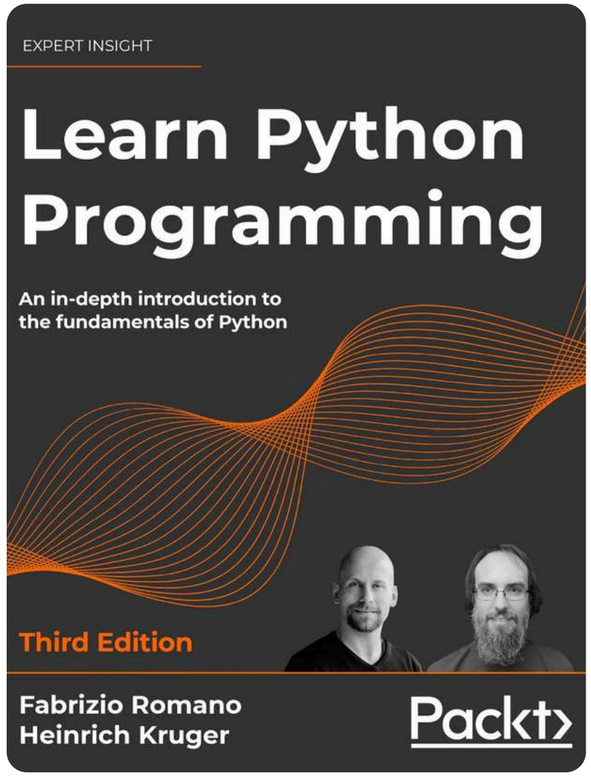
B Process

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

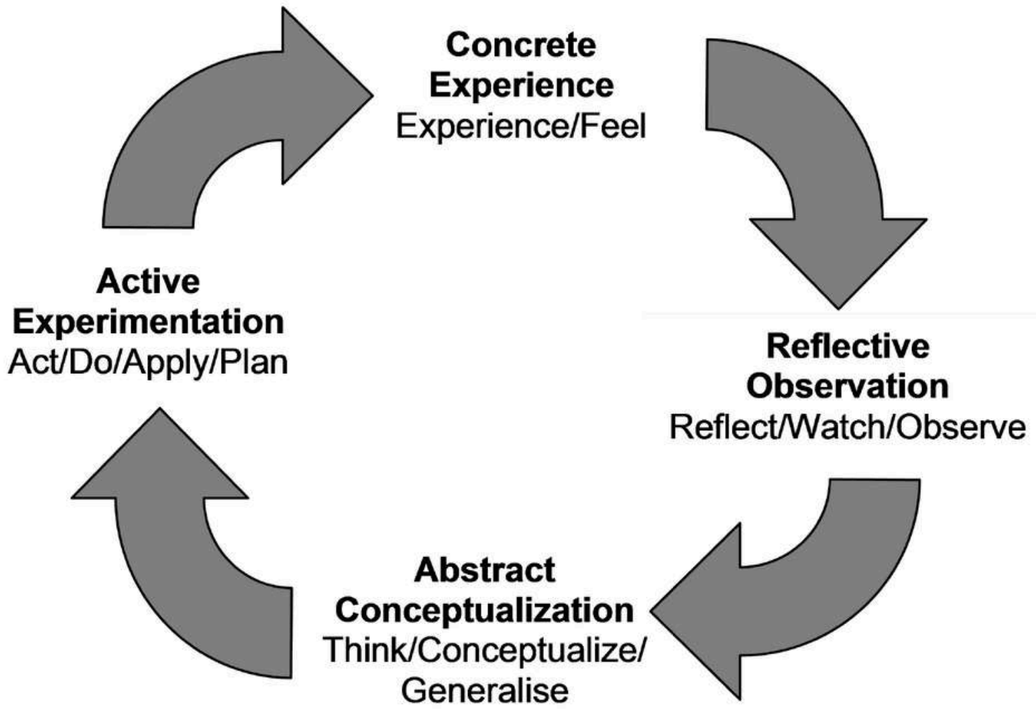
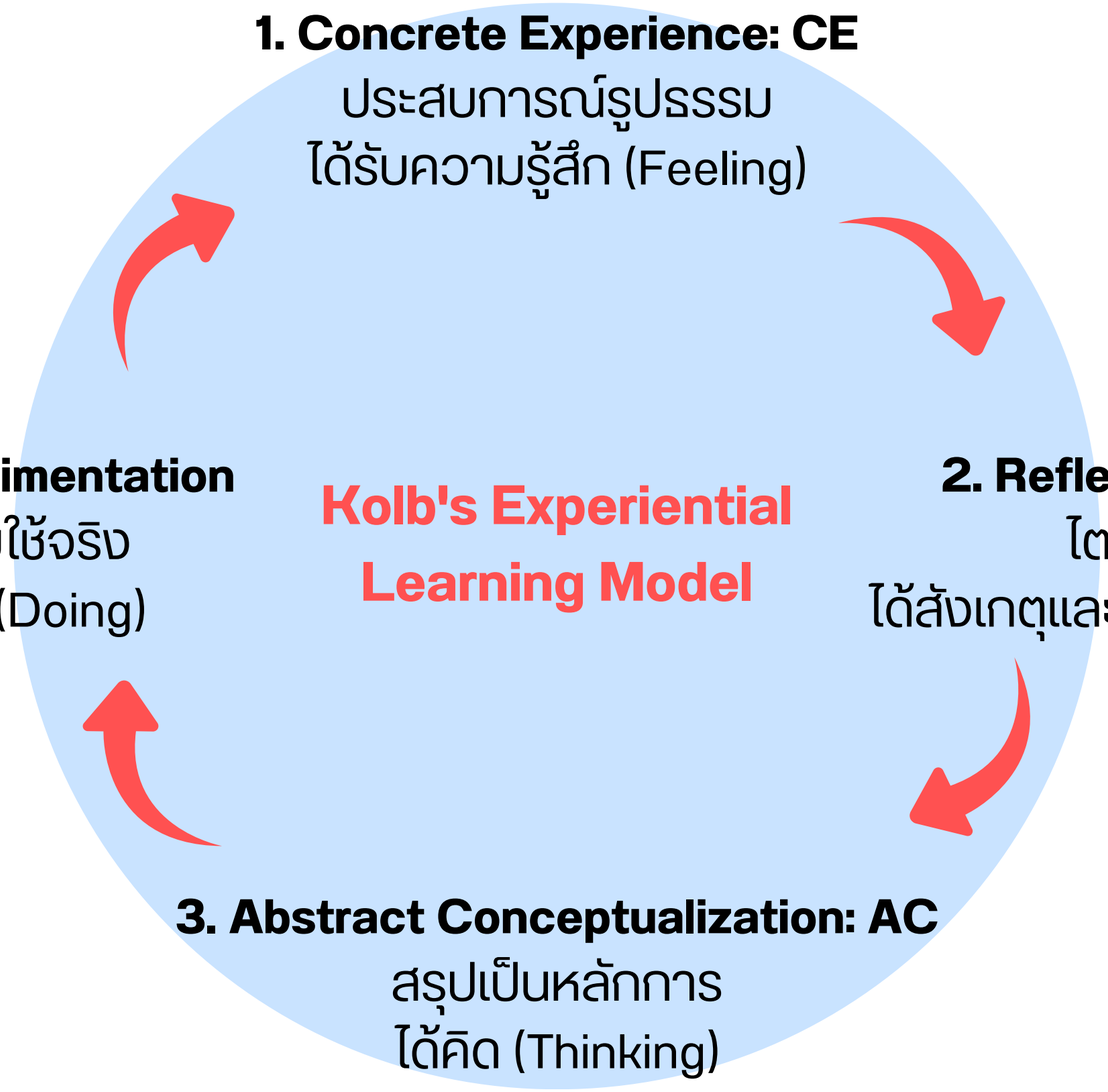


Figure 1. Model of Kolb's (1984) Experiential Learning

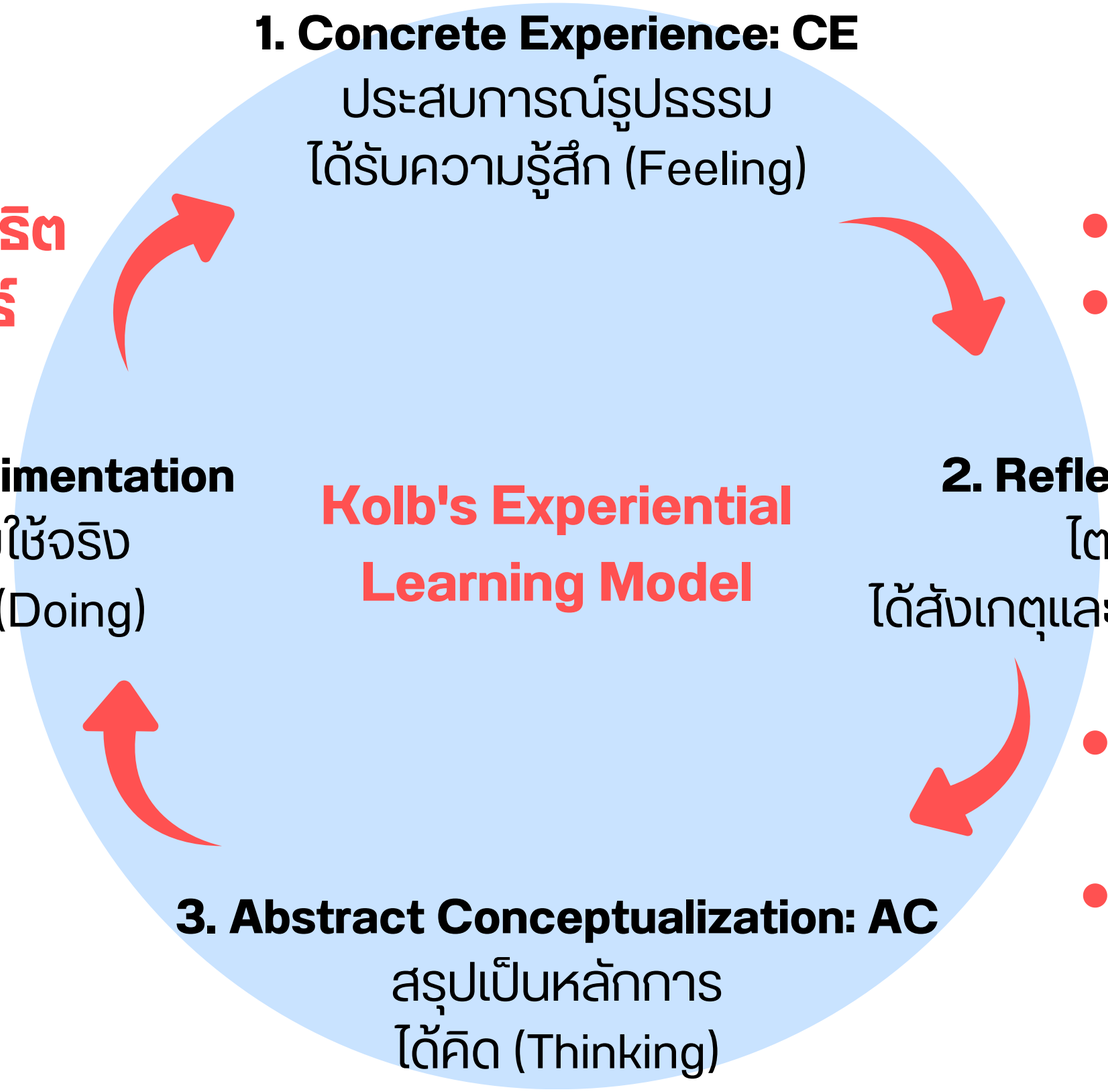
Experiential Learning หมายถึง การเรียนรู้ที่เกิดขึ้นจากประสบการณ์ที่ได้ทำและปฏิบัติจริง ไม่ใช่แค่การเรียนรู้จากหนังสือหรือบทเรียนที่ถูกสอนเท่านั้น เมื่อผู้เรียนได้มีโอกาสสัมผัสและทดลองทำสิ่งต่างแล้ว จะช่วยให้เกิดการเรียนรู้ที่มีความหมายและจดจำ ได้มากขึ้นเช่น

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

Learning by Doing – Experiential Learning



Learning by Doing – Experiential Learning



- ลองดูการสาธิต
- ลองดูผลลัพธ์

- ลองใช้ Program
- ลองดูส่วนประกอบ

- ลองคิดว่าแล้วมันเกิดผลลัพธ์ได้เพราะเหตุใด?
- ลองสร้างเลียนแบบหรือพัฒนาให้ดีกว่า

- ลองสรุปเป็นขั้นตอนหรือ Algorithm

Experiential Learning & Computer Programming

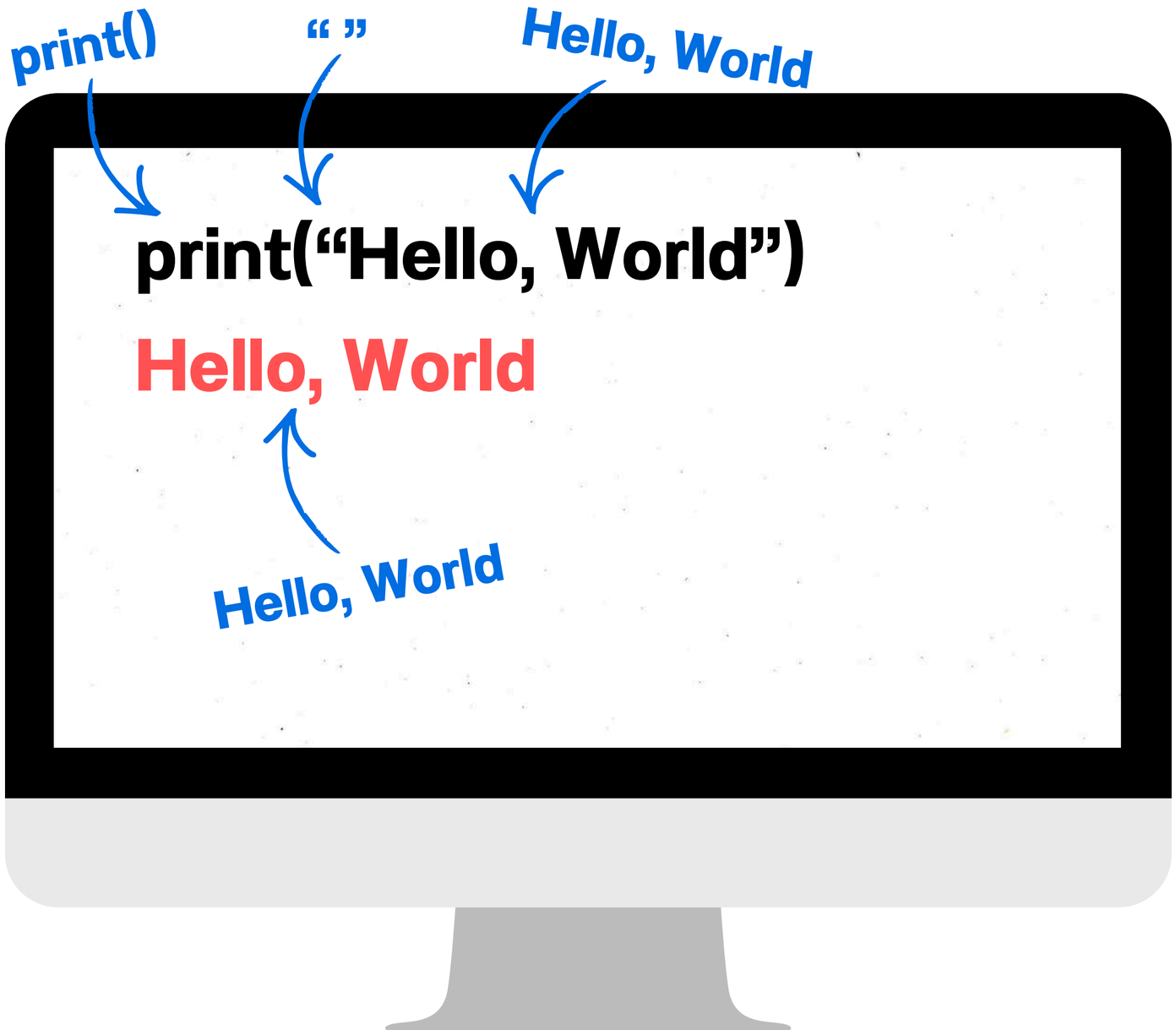
ขั้นตอนที่ 1

Concrete Experience: CE
ประสบการณ์รูปธรรม

Experiential Learning & Computer Programming

ขั้นตอนที่ 2

Reflective Observation: RO
ไตร่ตรอง ใคร่ครวญ

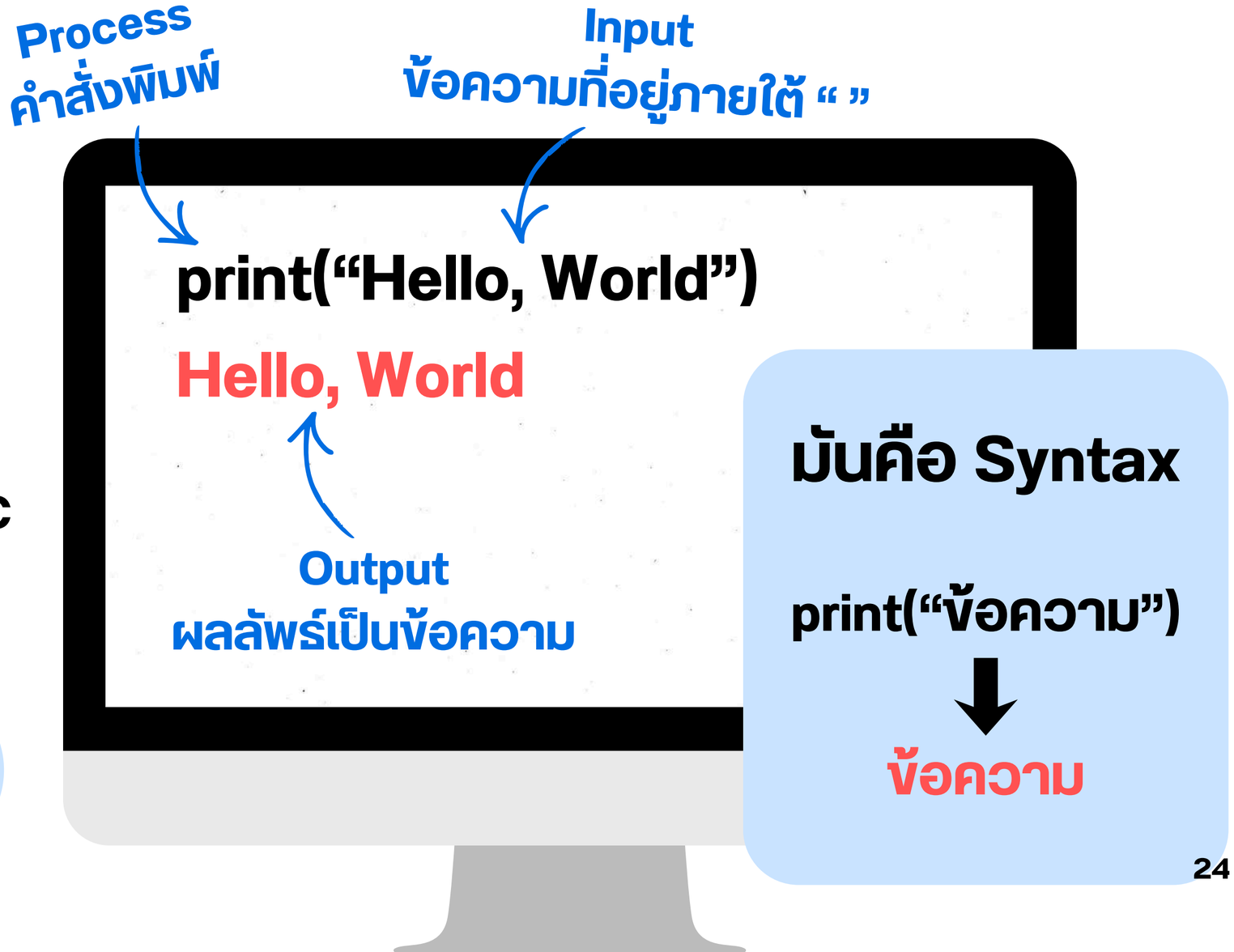


Experiential Learning & Computer Programming

ขั้นตอนที่ 3

Abstract Conceptualization: AC
สรุปเป็นหลักการ

Input → Process → Output



มันคือ Syntax

print(“ข้อความ”)

⬇

ข้อความ

Experiential Learning & Computer Programming

ขั้นตอนที่ 4

Active Experimentation
ทดลองปรับใช้จริง

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

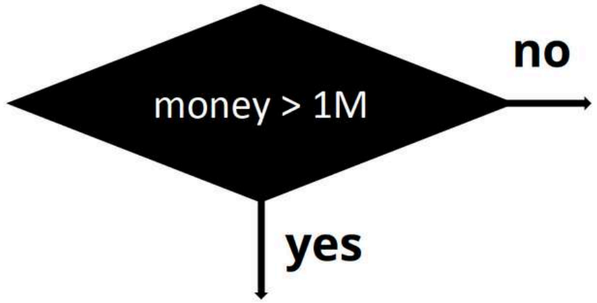
โปรแกรมทั่วไปจะถูกเขียนขึ้นด้วย**ภาษาโปรแกรม เช่น Python, Java, C++ และอื่น ๆ** ซึ่งภาษาเหล่านี้ใช้ไวยากรณ์และกฎเฉพาะ เพื่อสื่อสารกับคอมพิวเตอร์ ทำให้นักพัฒนาสามารถสร้างแอปพลิเคชัน เว็บไซต์ ซอฟต์แวร์ หรือระบบต่าง ๆ ที่มีความซับซ้อนตั้งแต่ง่ายไปจนถึงยากได้

โค้ดดิ้ง Coding คือข้อใด?

A 10011100110101

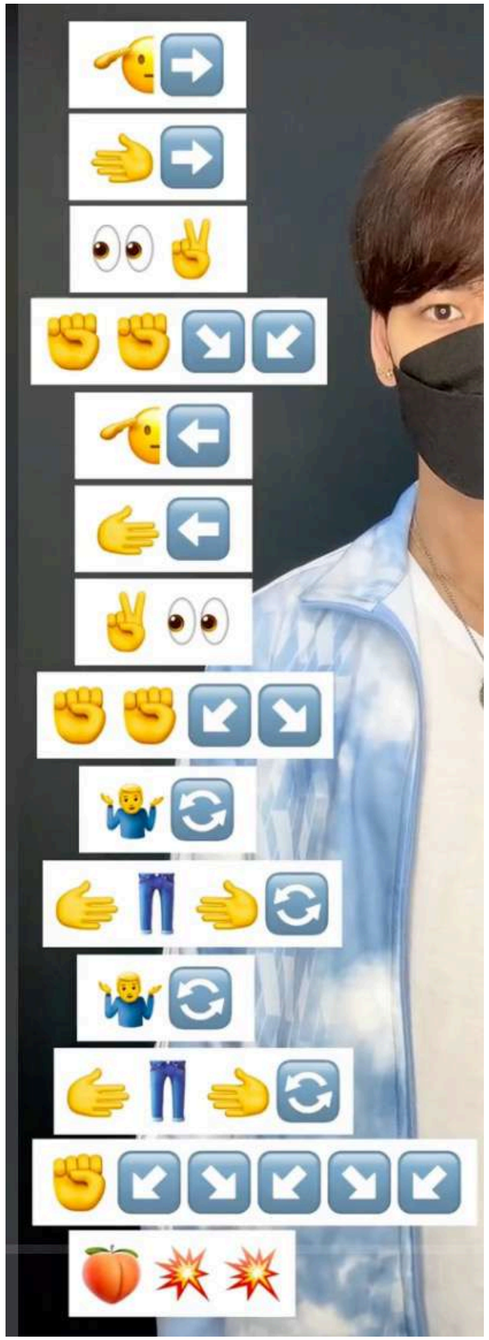
B print(“Hello World!”)

C

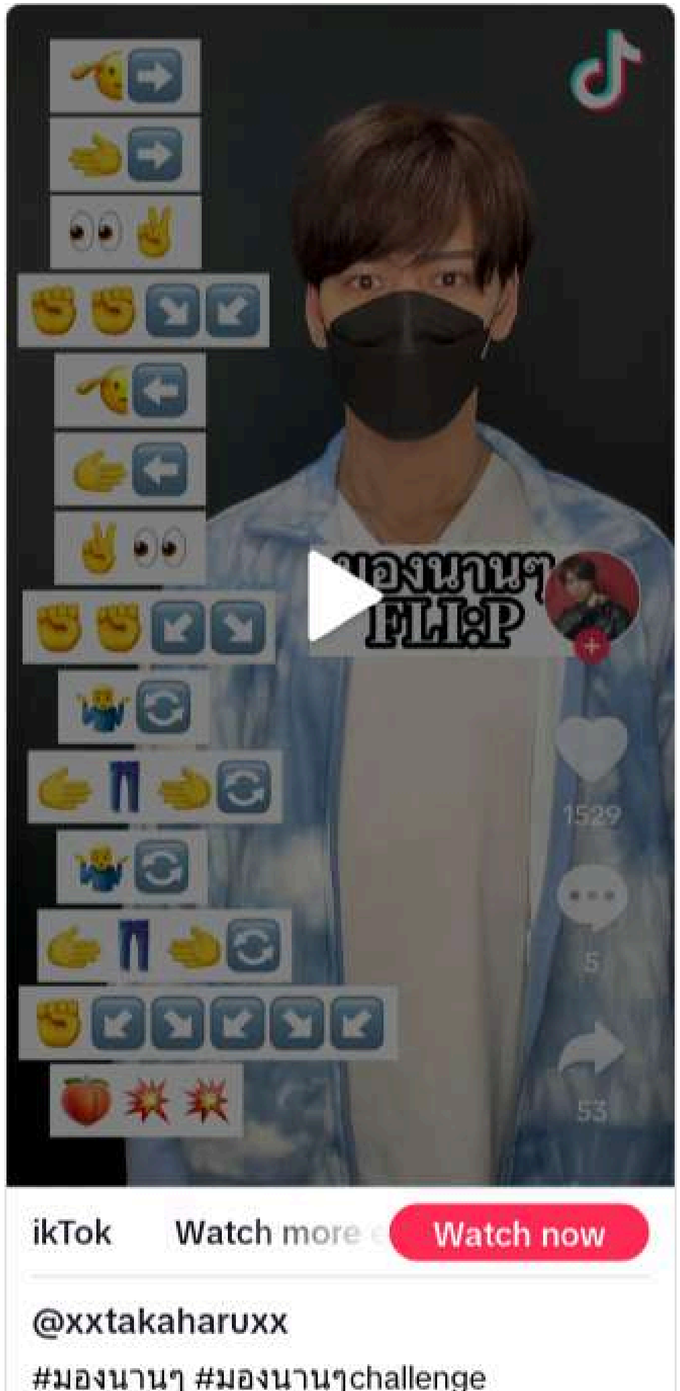


D 👉➡️ 🤜➡️

ท่าเต้นเพลงอะไร?



ท่าเต้นเพลงอะไร?



โค้ดดิ้ง (Coding)

- Code คือ ระบบของคำ อักษร หรือ สัญลักษณ์ ในการแทนข้อความในรูปแบบหนึ่งๆ หรืออาจเป็นระบบจำนวน/ตัวเลข อักษร หรือสัญญาณที่ใช้แทนอะไรบางอย่างในลักษณะที่สั้นกระชับ หรือ อยู่ในรูปแบบที่เข้าใจได้สะดวกขึ้น

- ดังนั้น พวก รหัส, เลขฐาน, สูตร, Pseudocode, Flowchart, Diagram, Computer Programming ต่าง ๆ นับว่าเป็น Code ในลักษณะหนึ่ง เพราะอยู่ในระบบของคำ สัญลักษณ์ที่สั้นกระชับ ชัดเจน สะดวกต่อการทำความเข้าใจ และการแปลงไปสู่หน่วยอื่น ๆ ได้

Coding - Debug

ADA LOVELACE

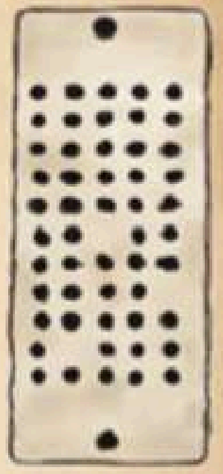
FIRST COMPUTER PROGRAMMER

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 want 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."

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



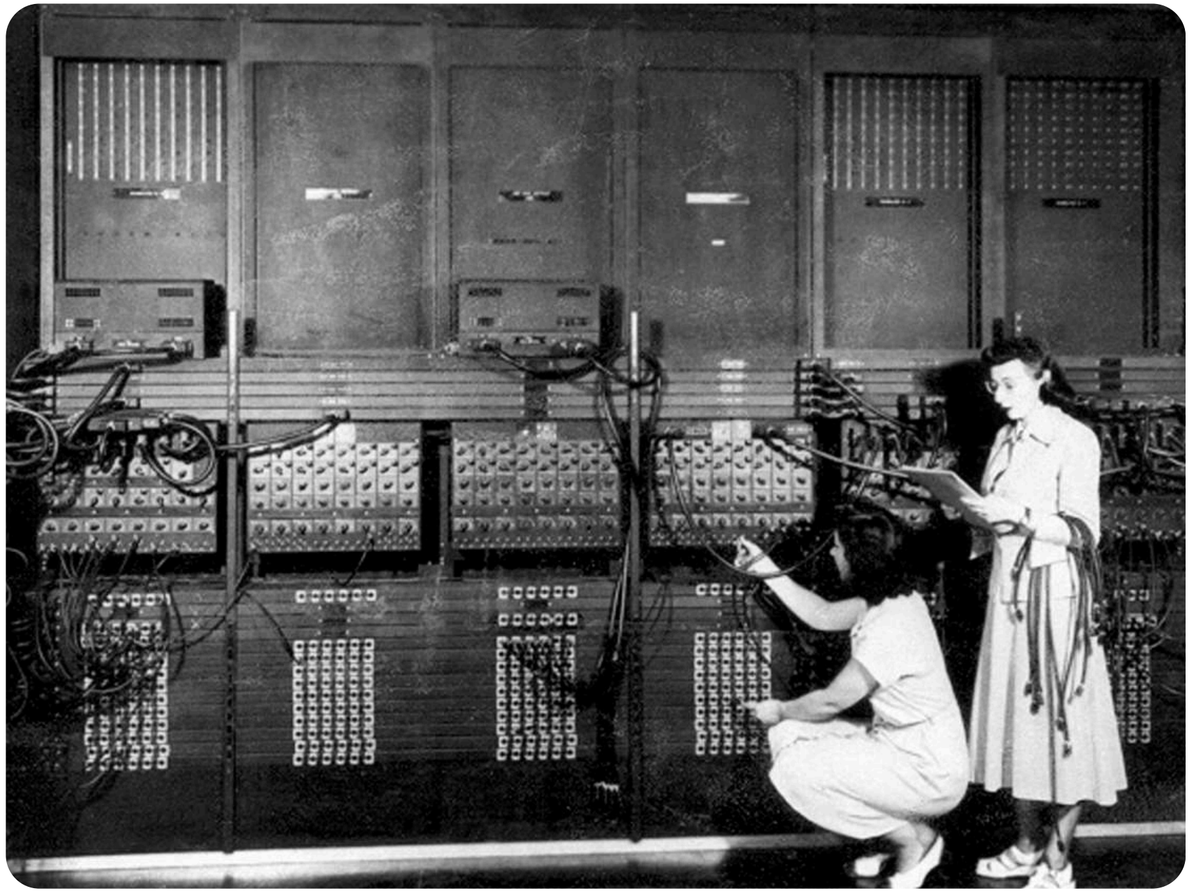
A Universal Computer

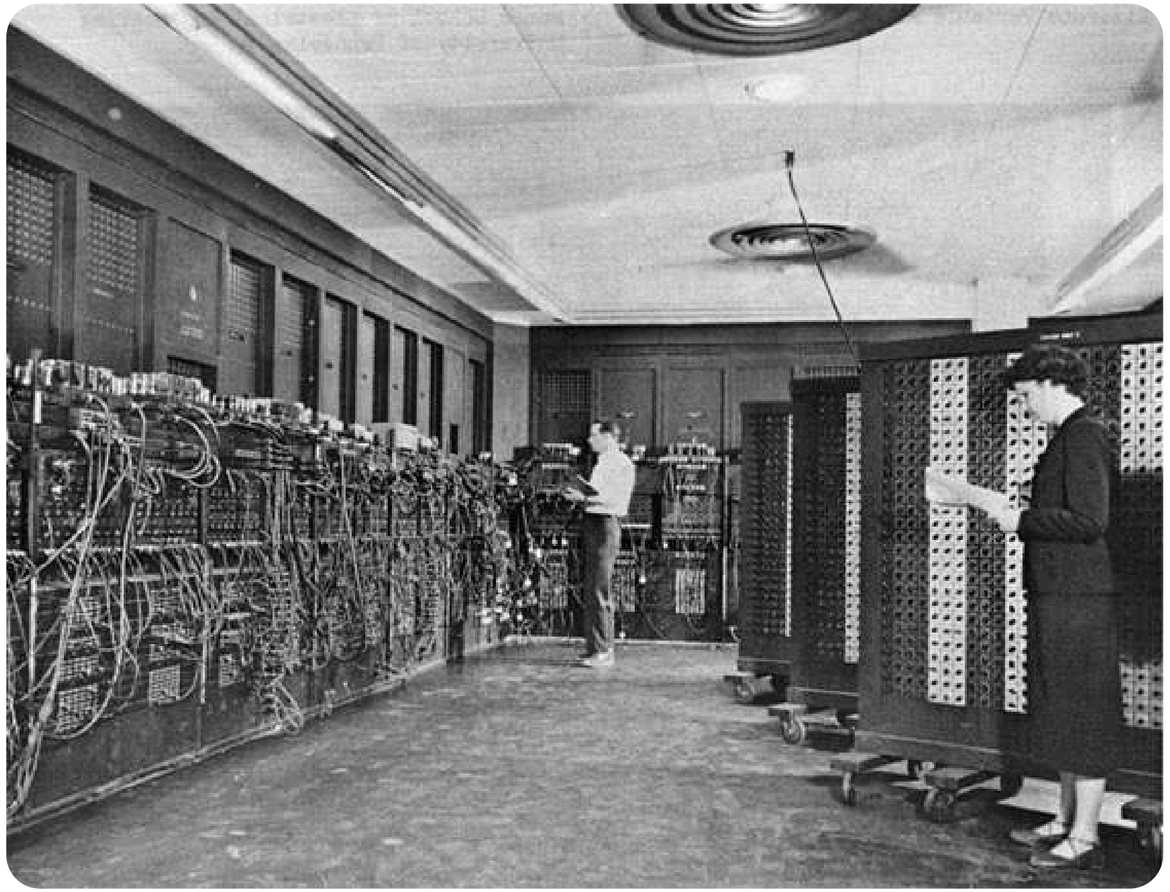
Lovelace did more than write the first computer program. She was also the first person to realise that a general purpose computer could do anything, given the right data and instructions.

"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."

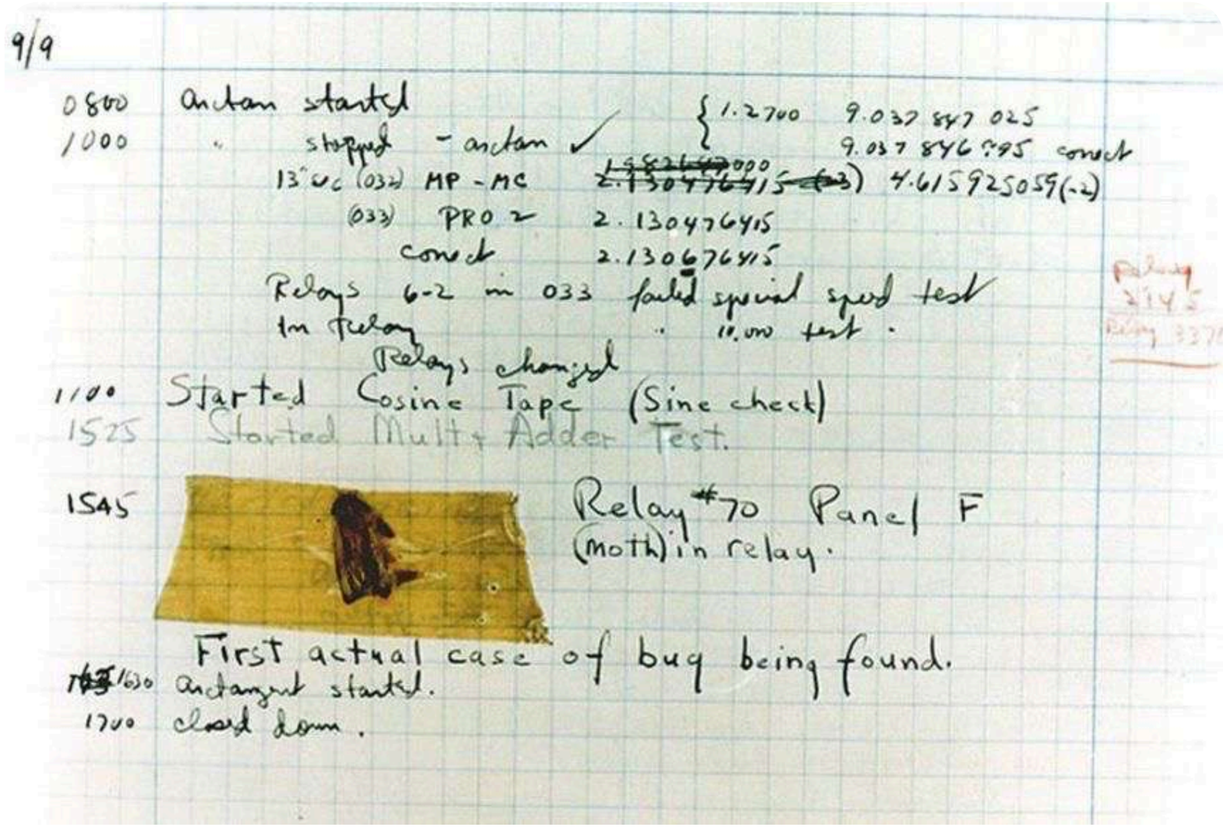
Coding - Debug





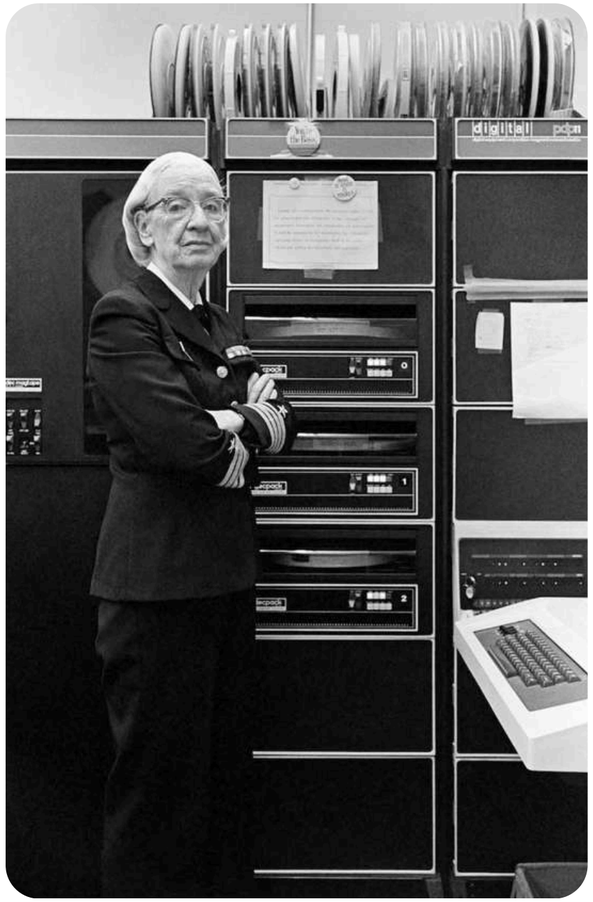
On the 9th of September of 1946,
a Harvard technical team was asked
to take a look at a malfunction that was
happening with the Mark II calculator.

Coding - Debug



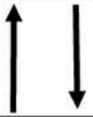
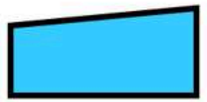
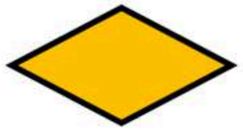
Grace Hopper, an American computer scientist and United States Navy rear admiral working on the Mark II calculator project at the time, added the caption “First actual case of bug being found,” and that’s the first time anyone used the word bug to describe a computer glitch. Naturally, the term debugging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

Grace Brewster Hopper



โค้ดดิ้ง (Coding)

รหัสจำลองสำหรับการเดินทาง	
รหัสจำลอง	**ความหมาย**
→	เดินไปข้างหน้า
↑	เดินขึ้นข้างบน
↓	เดินลงไปข้างล่าง
←	เดินไปทางซ้าย
↱	เดินเลี้ยวขวา
↰	เดินเลี้ยวซ้าย
↱	เดินเลี้ยวขึ้นข้างบน
↳	เดินเลี้ยวลงข้างล่าง

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 : ลากเส้นจากจุด

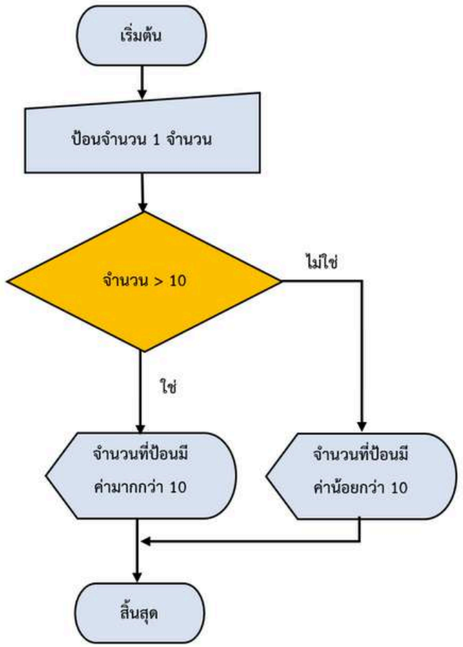
(C,2) → (A,4) → (A,7) → (B,7) → (B,5) → (C,5) → (C,9) → (D,9) → (D,7) → (G,7) → (G,9) → (H,9) → (H,6) → (I,7) → (I,6) → (H,5) → (H,2) → (C,2)

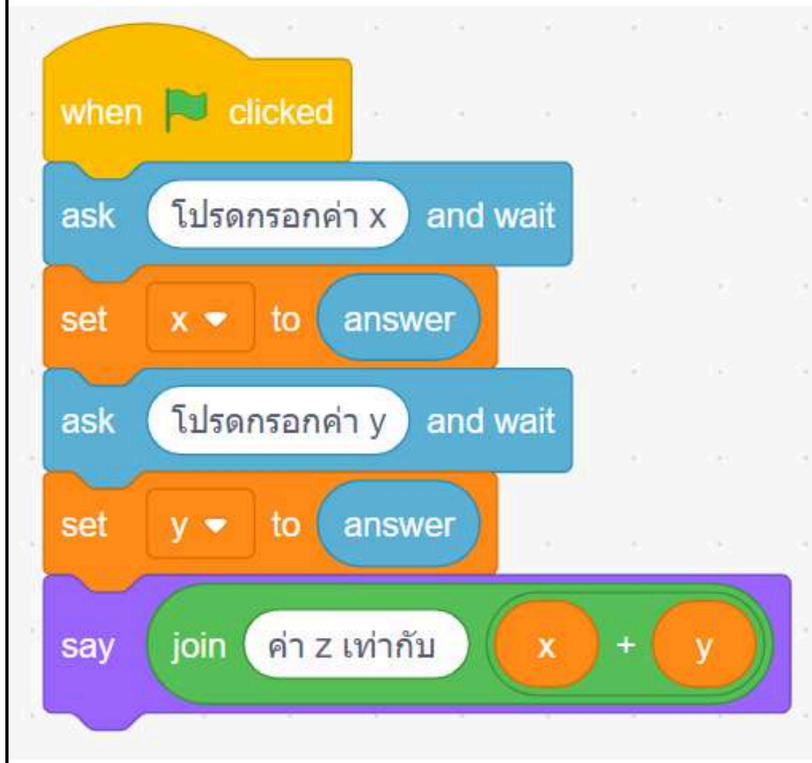
รหัสจำลองของชุดคำสั่งที่ 2 : ลากเส้นจากจุด

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

แสดงผล: ..... ? .....

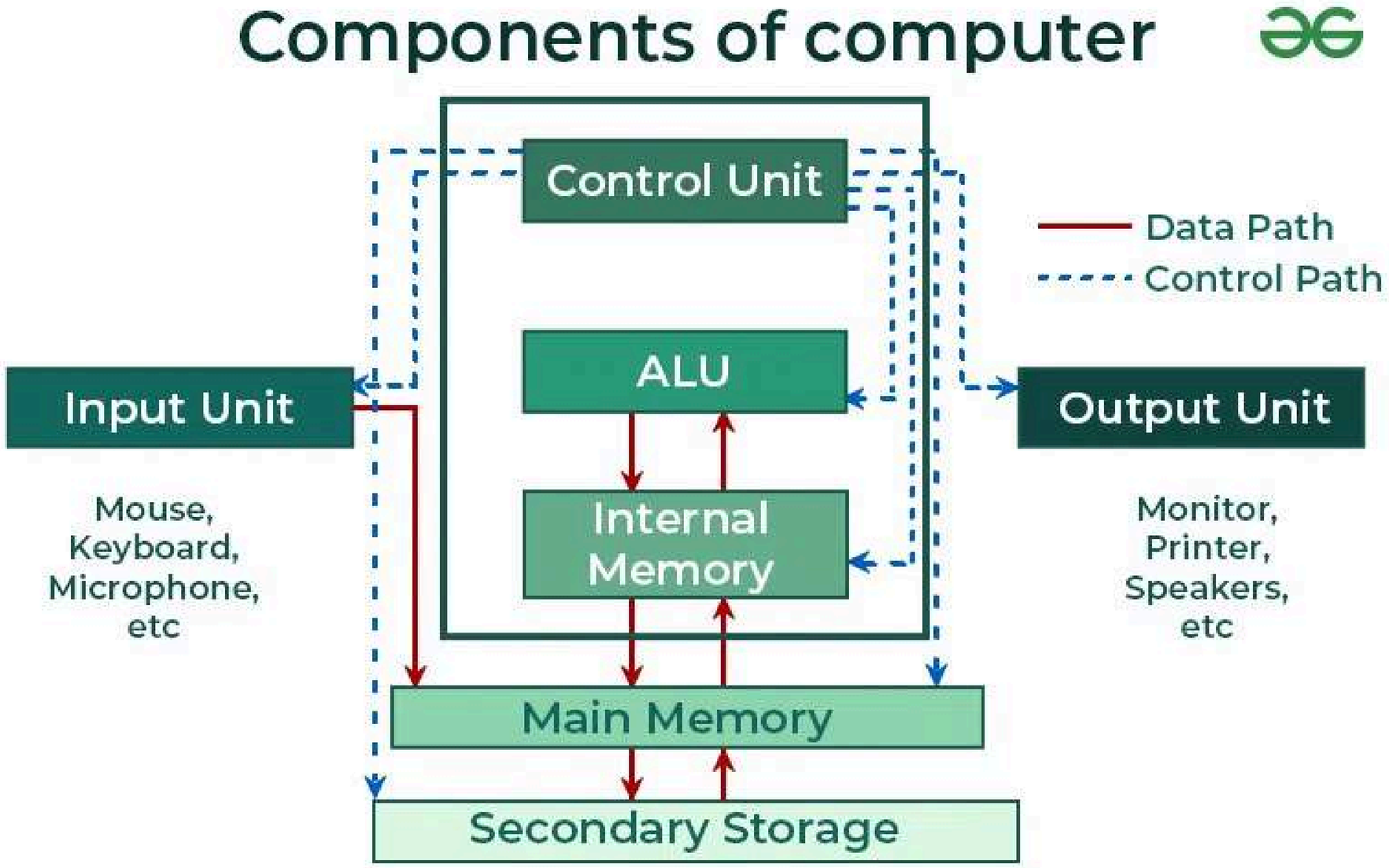
วัตถุประสงค์: ป้อนจำนวนด้วยแป้นพิมพ์และอยากทราบว่าจำนวนที่ป้อนเข้ามามากกว่า 10 หรือไม่



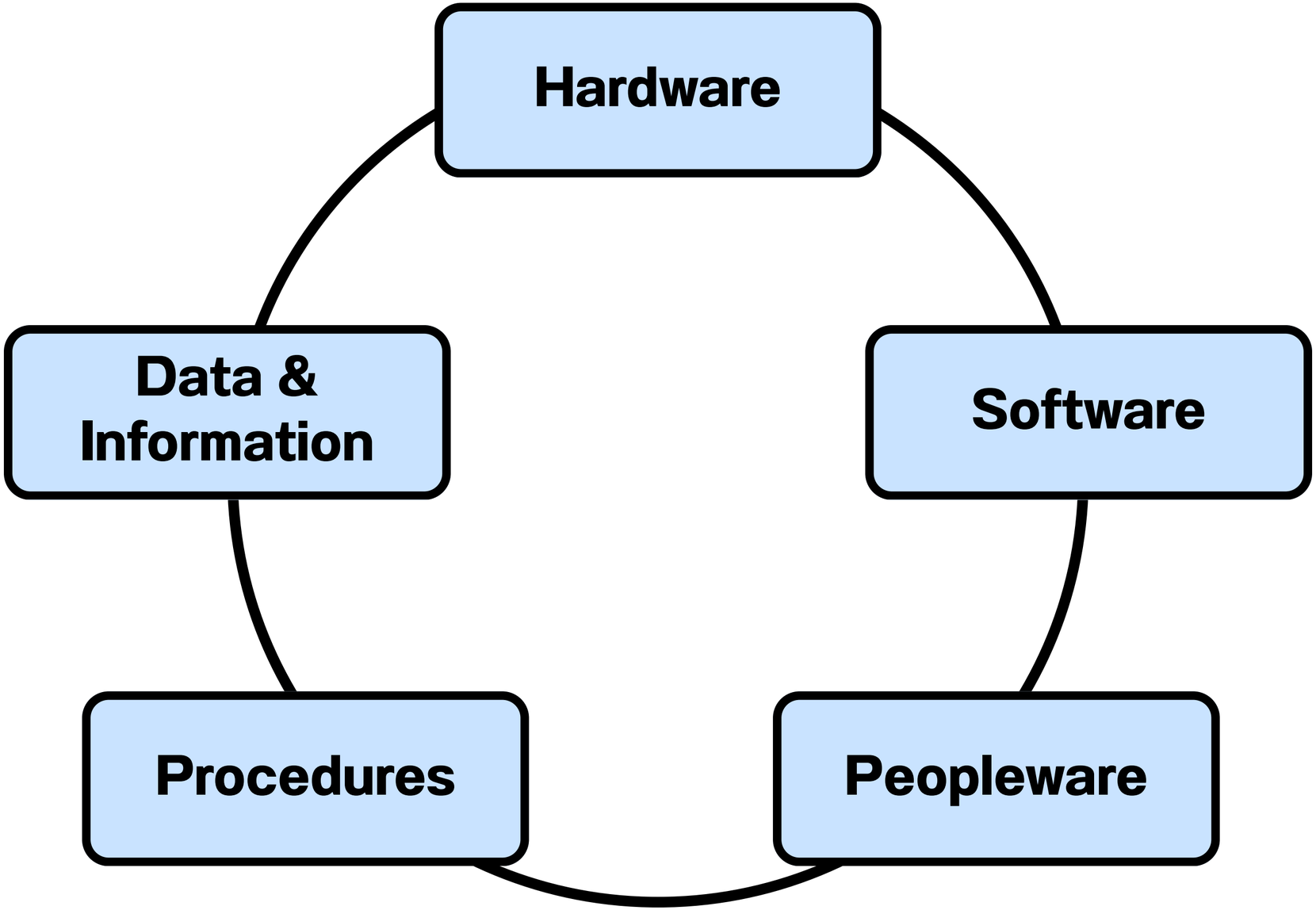


Computer System

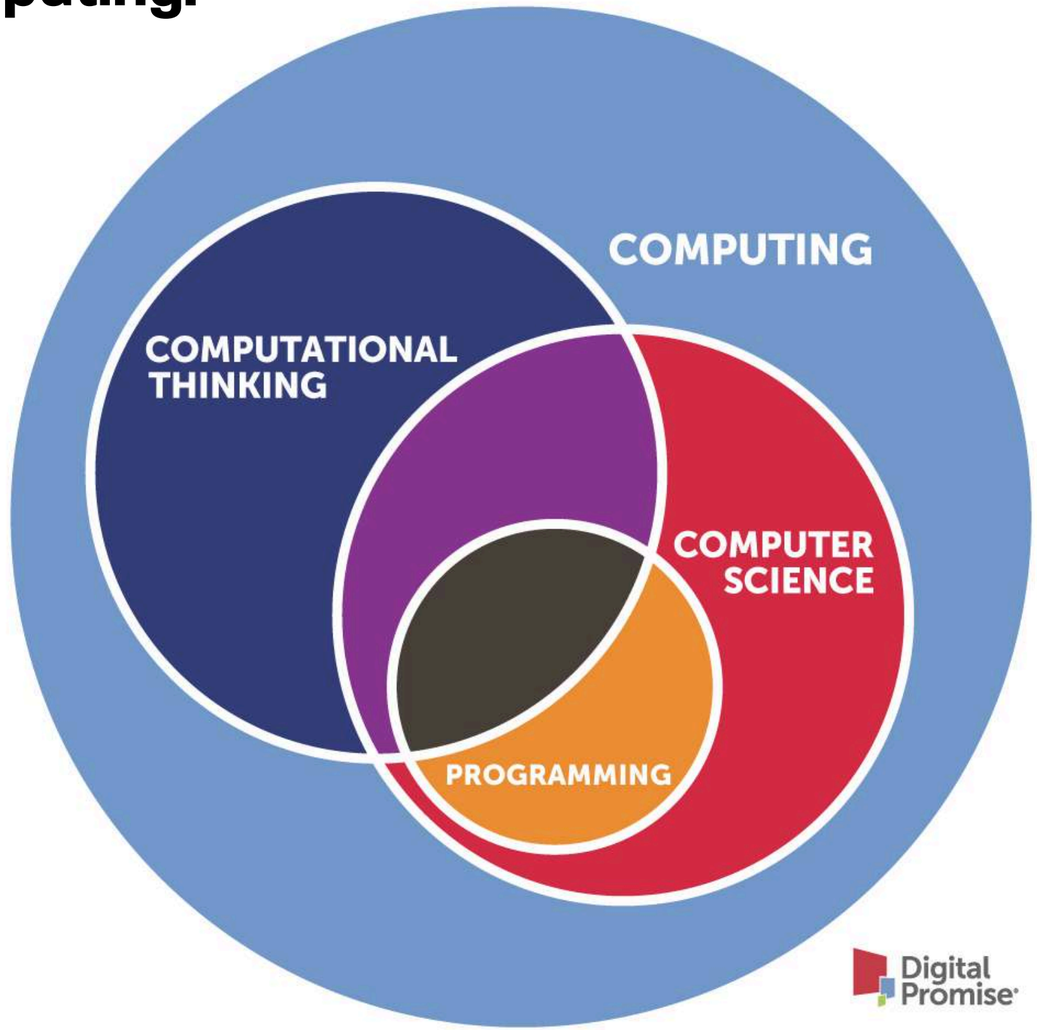
Computer Components



Computer Components

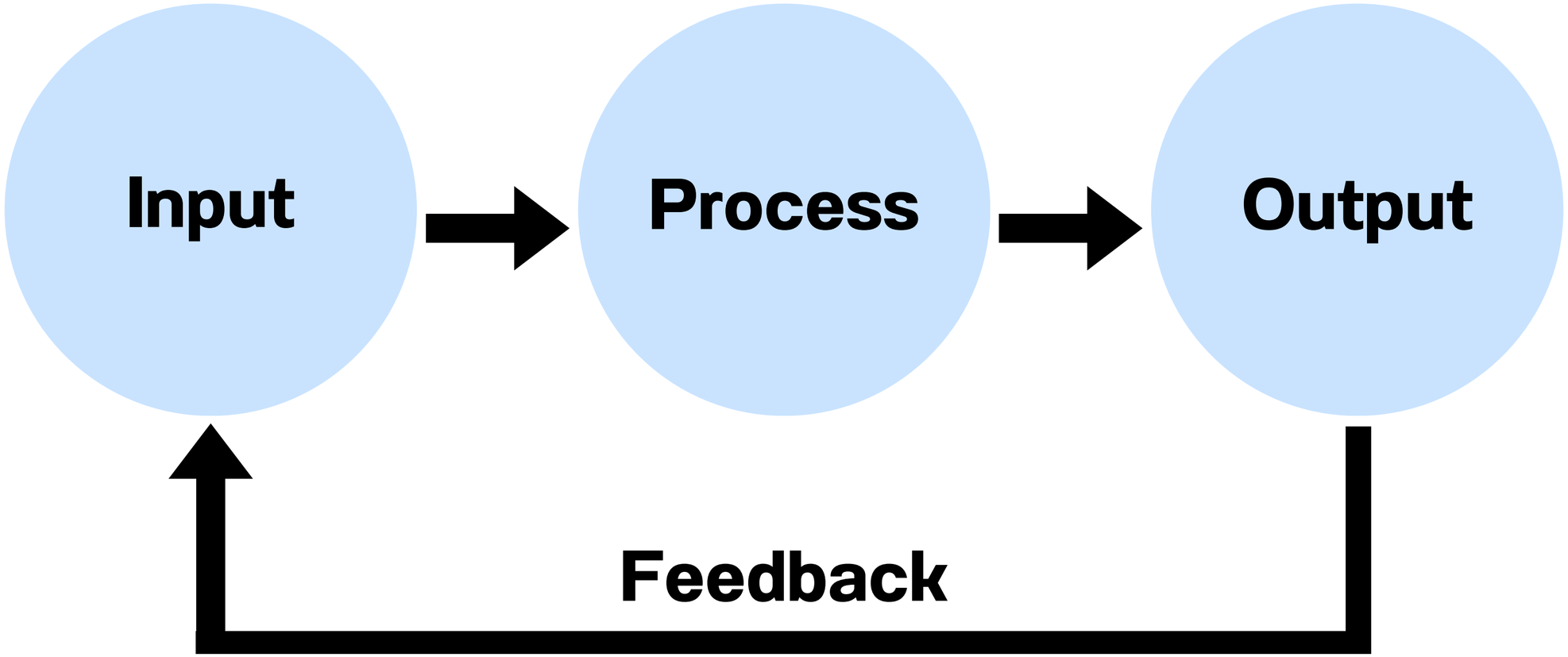


The relationship between computer science (CS), computational thinking (CT), programming and computing.



อ้างอิง: https://digitalpromise.org/initiative/computational-thinking/computational-thinking-for-nextgeneration-science/what-is-computationalthinking/

System Theory



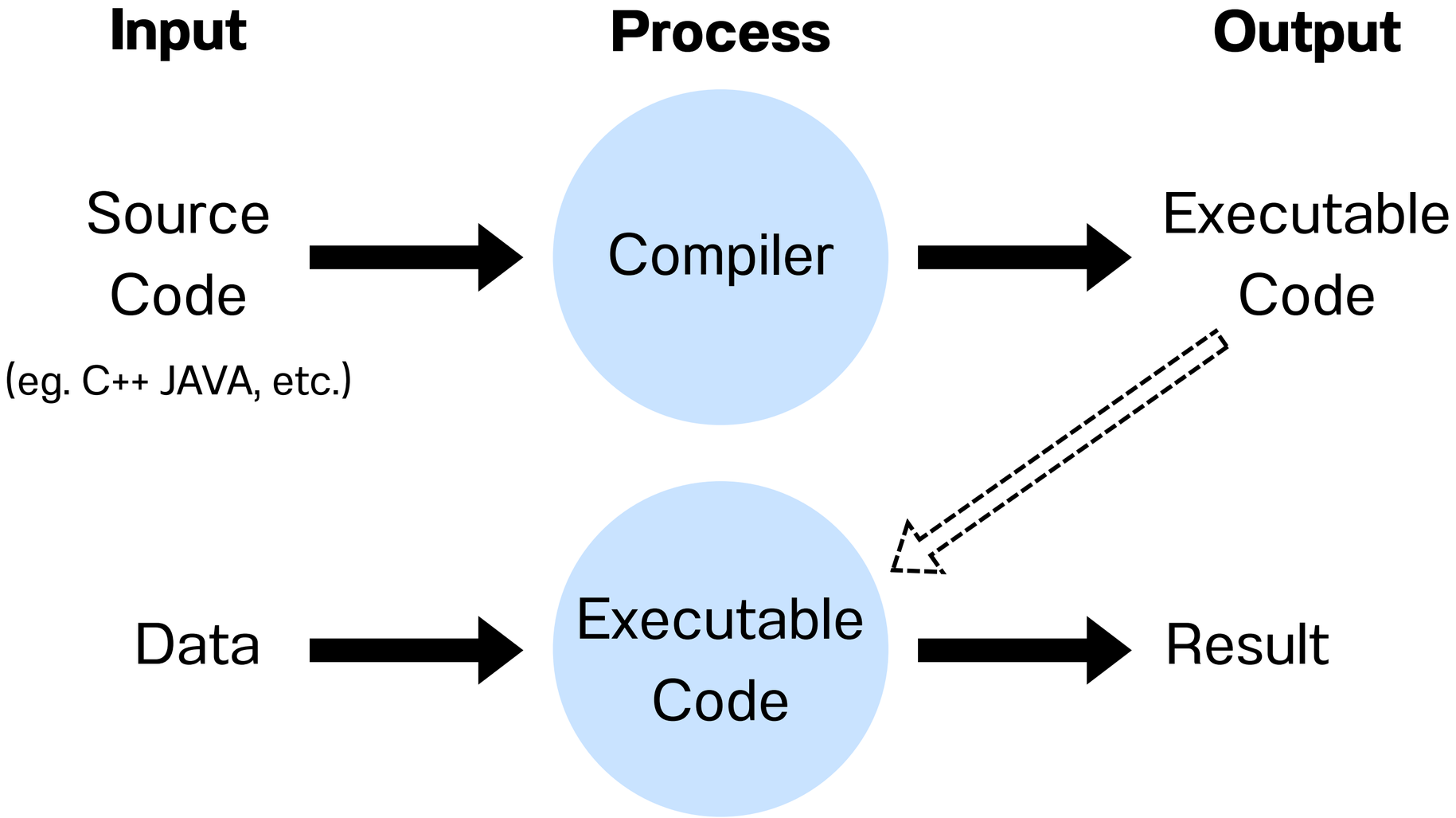
Compiler & Interpreter

- ในภาษาคอมพิวเตอร์หลากหลายภาษา จะมีกระบวนการแปลภาษาคอมพิวเตอร์อยู่ 2 แบบ ได้แก่

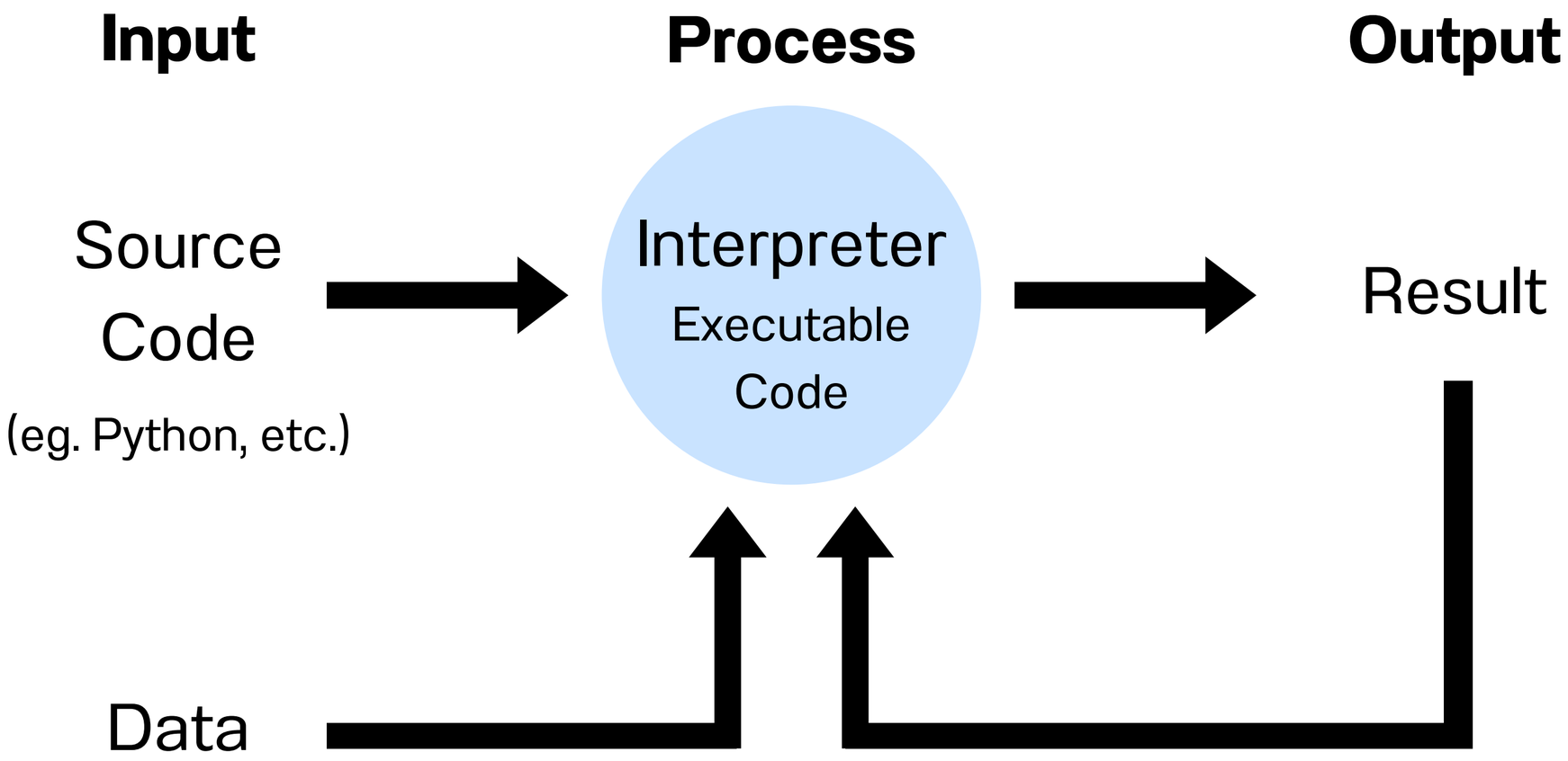
Compiler

Interpreter

Compiler

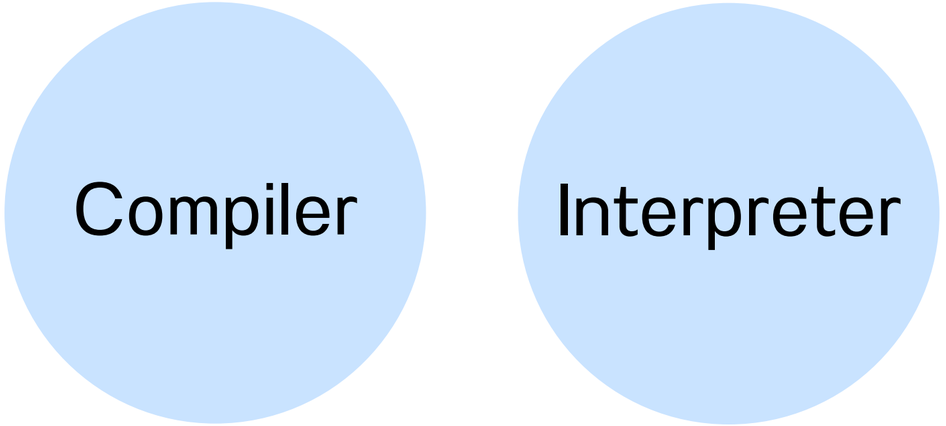

Input
Process
Output
Source Code
(eg. C++ JAVA, etc.)
Compiler
Executable Code
Data
Executable Code
Result

Interpreter – Line Interpretation



Compiler & Interpreter

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



Difference Between High-Level Language & Low-Level Language

High-Level Language

Python, C, C++, C# Language

Procedural Language

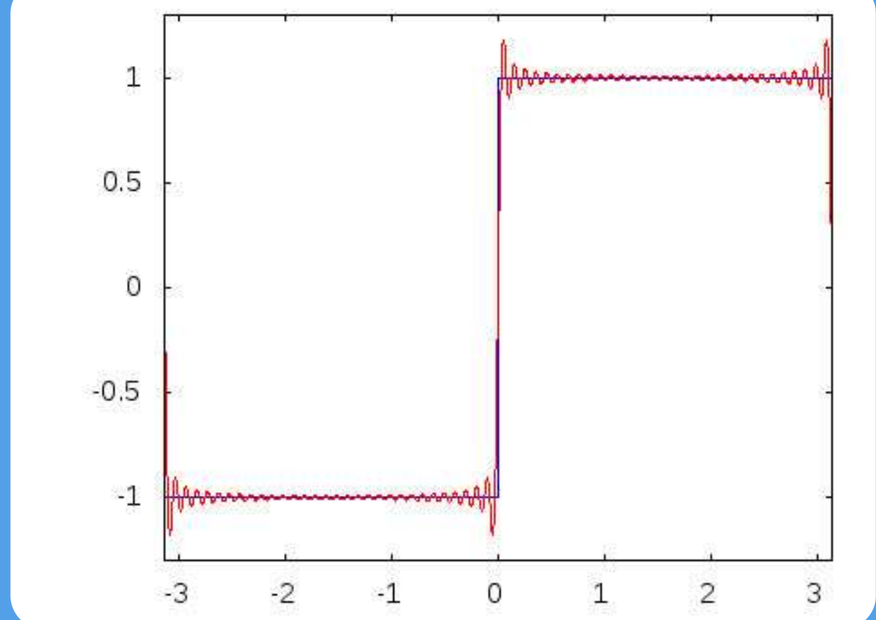
Object Oriented: OO

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

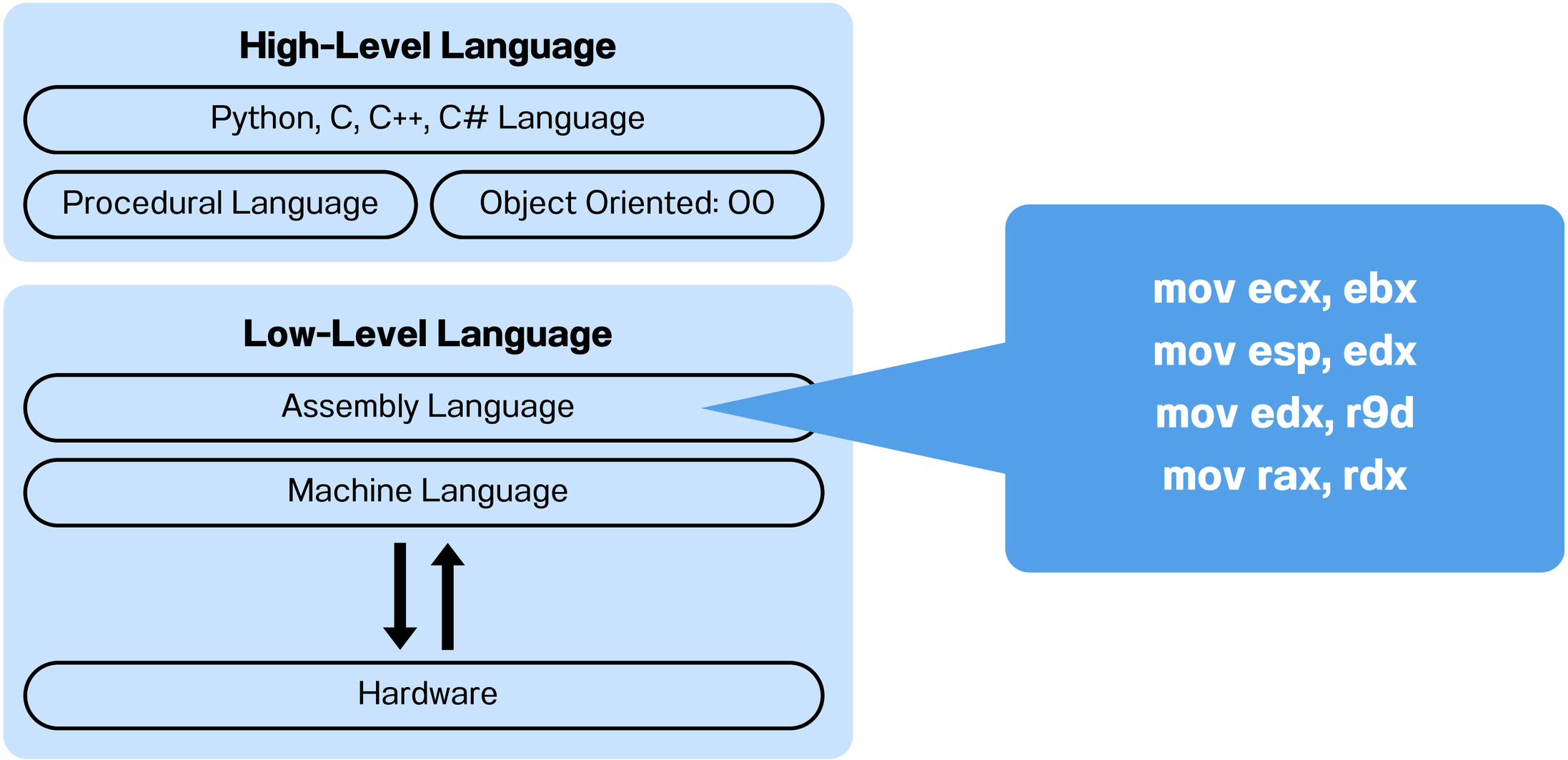
Assembly Language

Machine Language

Hardware

10010010010101111101010101

Difference Between High-Level Language & Low-Level Language



Difference Between High-Level Language & Low-Level Language

High-Level Language

Python, C, C++, C# Language

Procedural Language

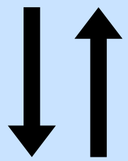
Object Oriented: OO

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

Low-Level Language

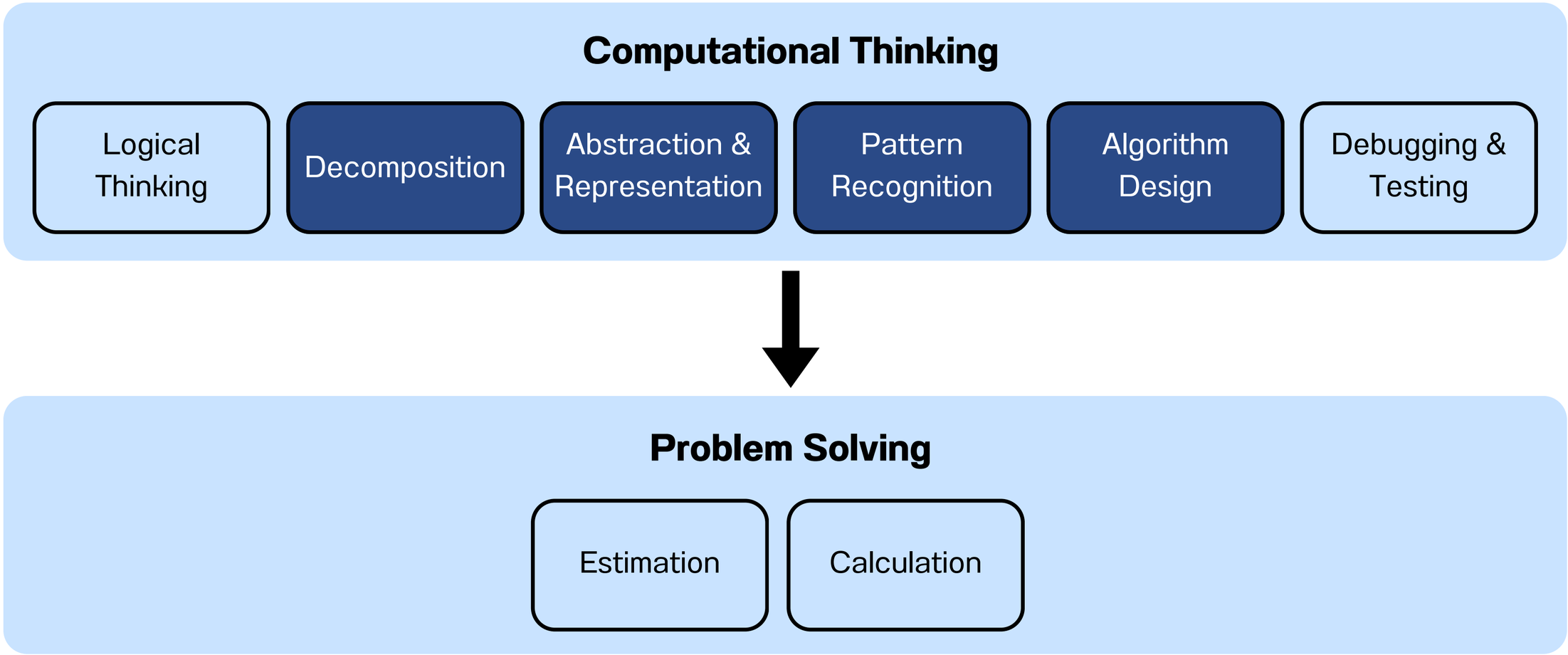
Assembly Language

Machine Language

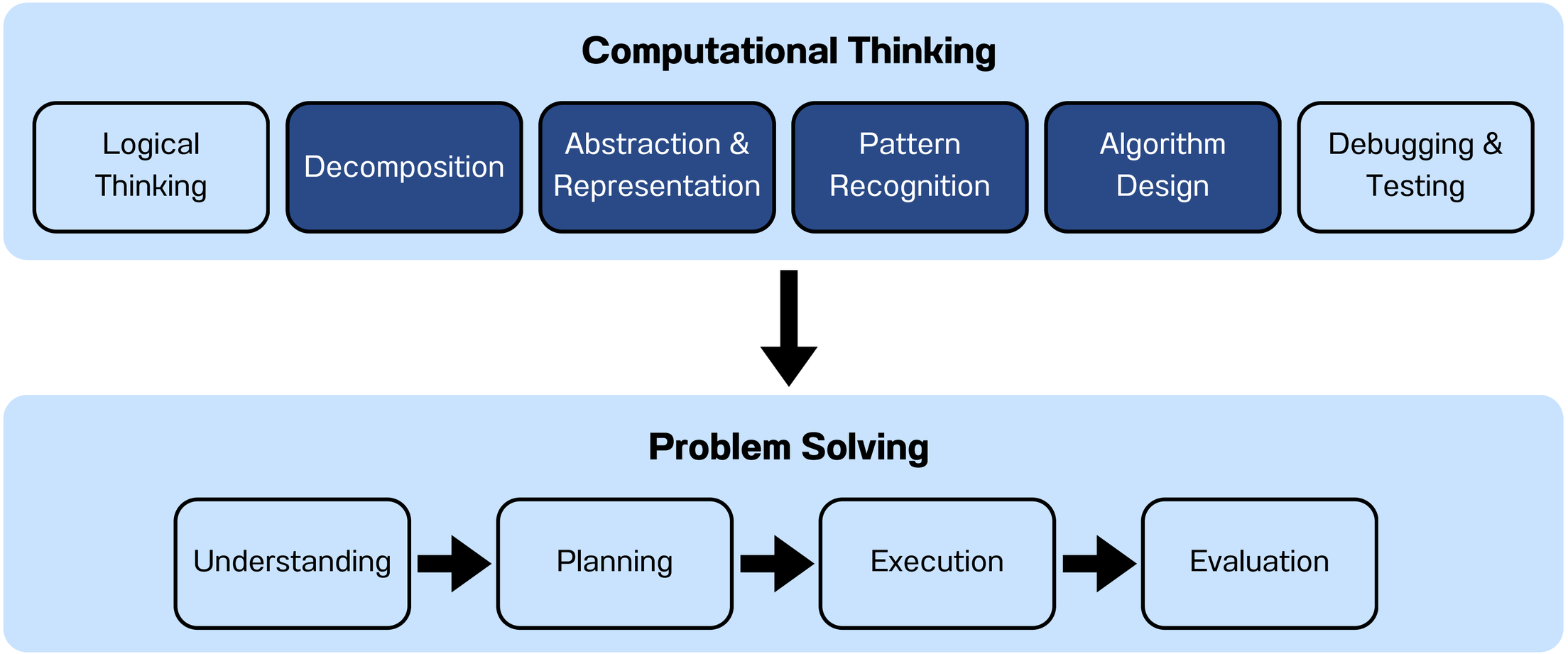


Hardware

Computational Thinking & Problem Solving



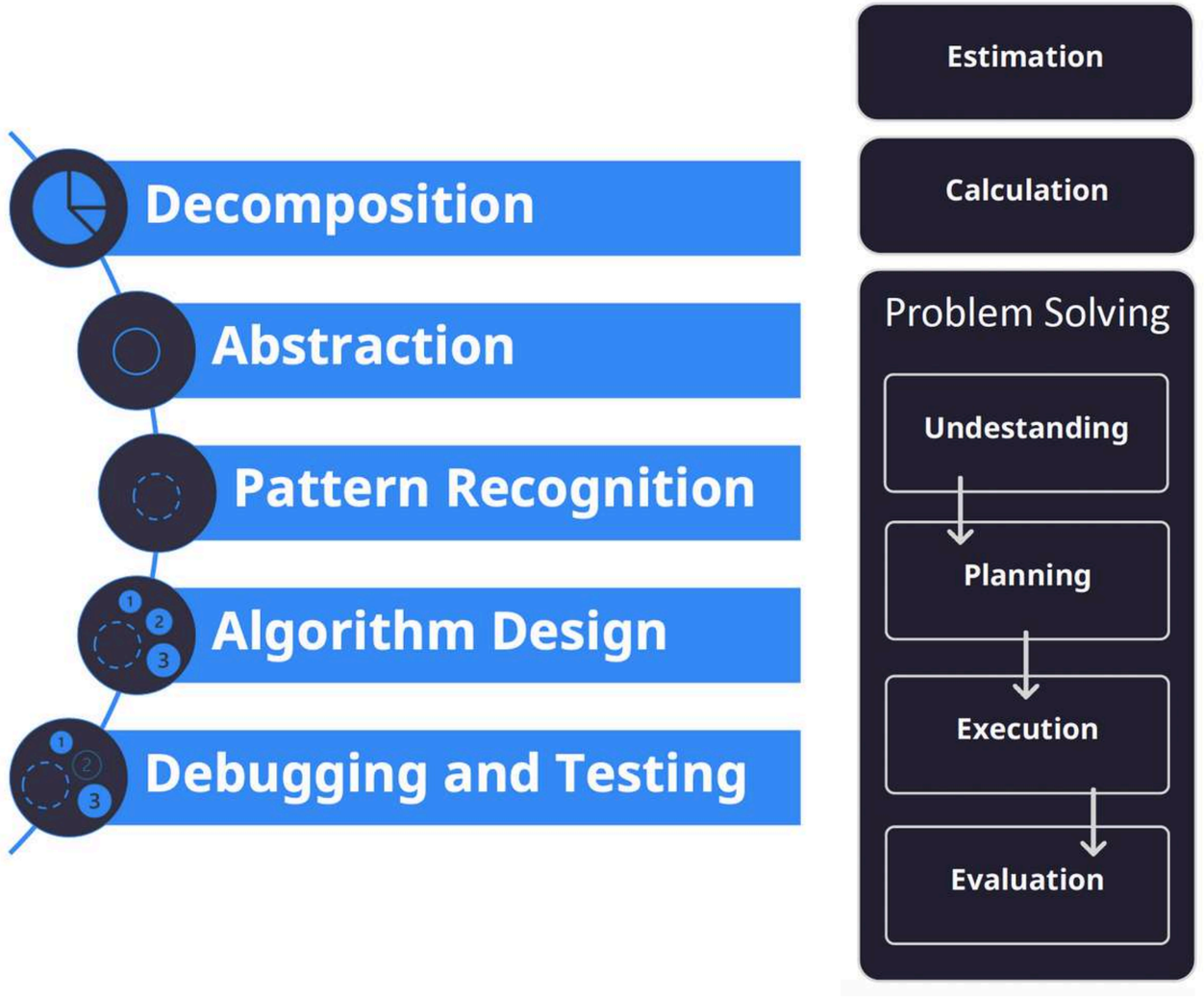
Computational Thinking & Problem Solving



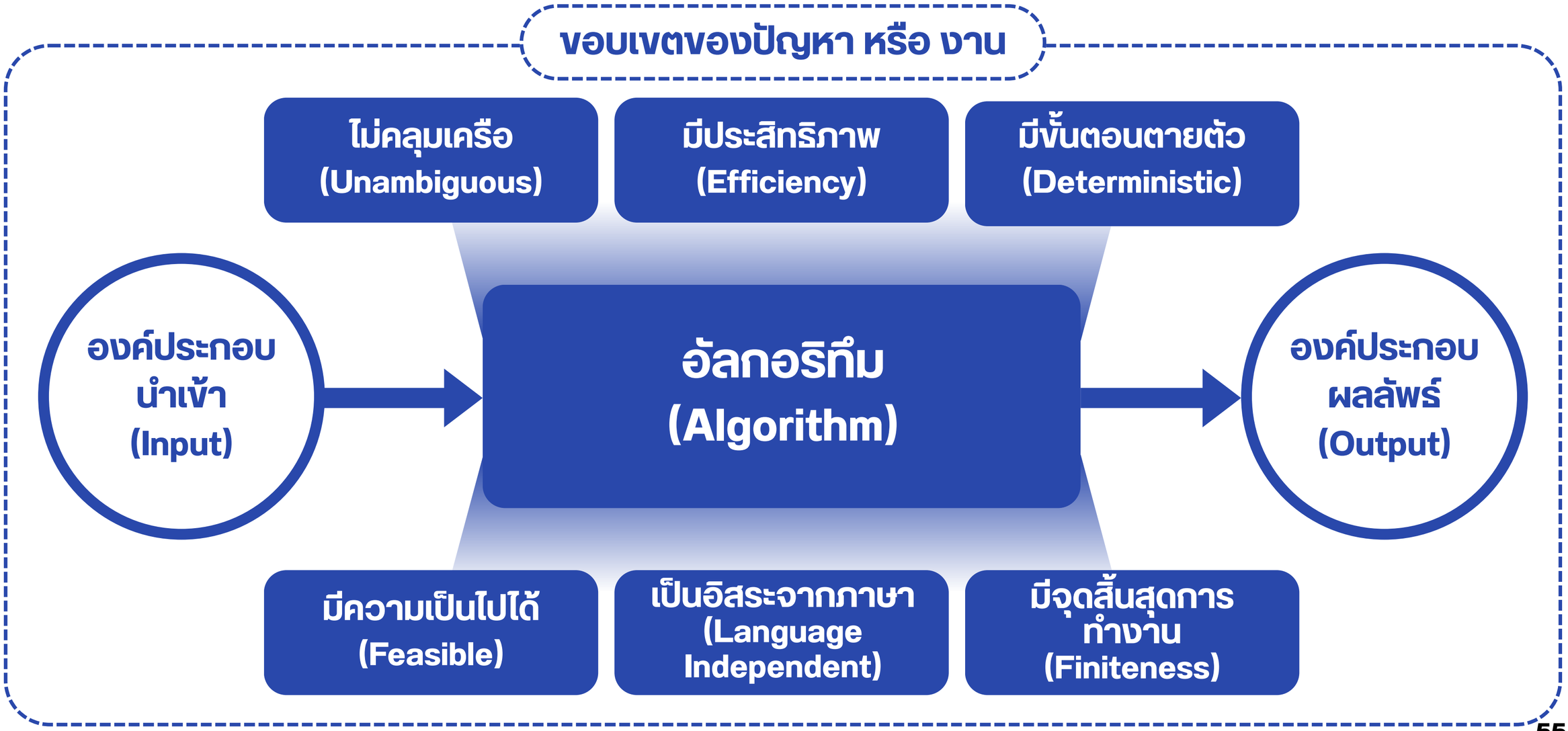
Python & Computational Thinking

Computational thinking is a problem solving method that involves

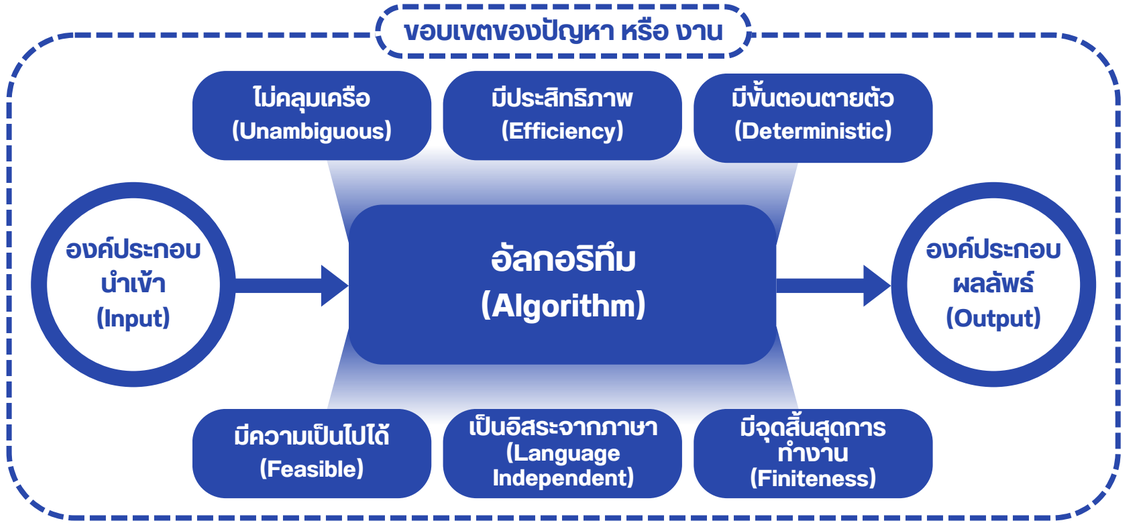
- breaking down complex problems into smaller, more manageable parts, and using logical reasoning and algorithms to solve them.
- Python is a programming language that can be used to implement computational thinking.



คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



ลักษณะของอัลกอริทึมที่ดีที่ประกอบด้วยคุณสมบัติ ดังนี้

1. ไม่คลุมเครือ (Unambiguous) อัลกอริทึมต้องไม่มีความกำหนดที่ไม่ชัดเจน และต้องมีชุดคำสั่งที่เข้าใจง่ายและไม่ทำให้เกิดความสับสนในการประมวลผล

2. มีประสิทธิภาพ (Efficiency) อัลกอริทึมควรมีประสิทธิภาพในการทำงาน คือต้องให้ผลลัพธ์ที่ถูกต้อง และใช้เวลาน้อยที่สุดในการแก้ปัญหา

3. มีขั้นตอนตายตัว (Deterministic) อัลกอริทึมต้องเป็น Deterministic ทำหมายความว่าจะต้องให้ผลลัพธ์ที่เหมือนกันสำหรับข้อมูลนำเข้าที่เหมือนกันทุกครั้งที่รัน

4. มีความเป็นไปได้ (Feasible) อัลกอริทึมต้องเป็นไปได้ในการนำไปใช้งานจริง สามารถทำงานบนเครื่องคอมพิวเตอร์หรือสภาพแวดล้อมที่เหมาะสม

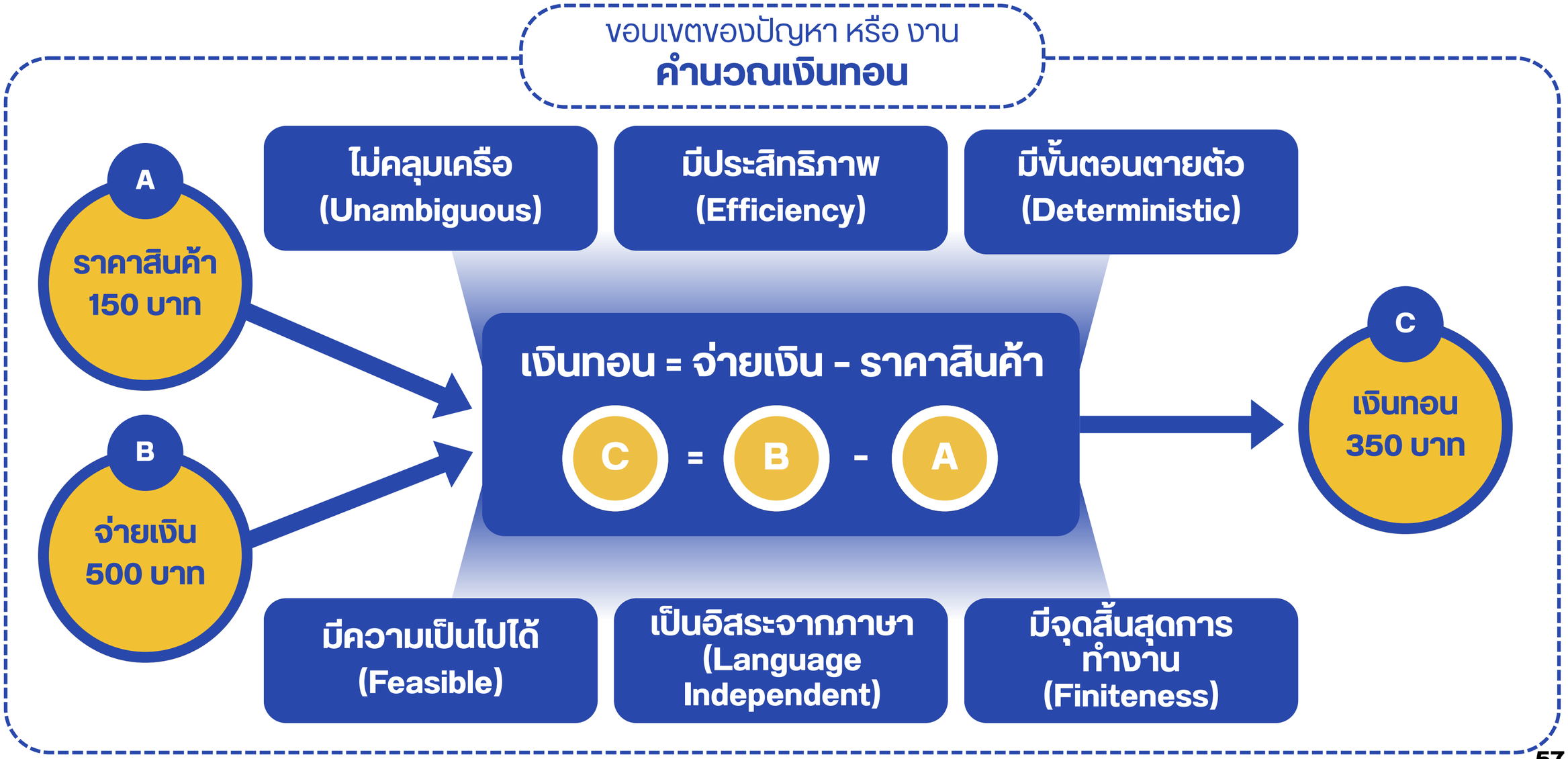
5. เป็นอิสระจากภาษา (Language Independent) อัลกอริทึมควรไม่เกี่ยวข้องกับภาษาเฉพาะเสียเพียงอย่างใด ๆ และสามารถนำไปใช้งานในหลายภาษาโปรแกรมได้

6. มีจุดสิ้นสุดการทำงาน (Finiteness) อัลกอริทึมต้องสามารถสิ้นสุดการทำงานหลังจากทำขั้นตอนทุกขั้นตอนเสร็จสมบูรณ์ และไม่ต้องเข้าสู่วงจรไม่สิ้นสุด (Infinity Loop)

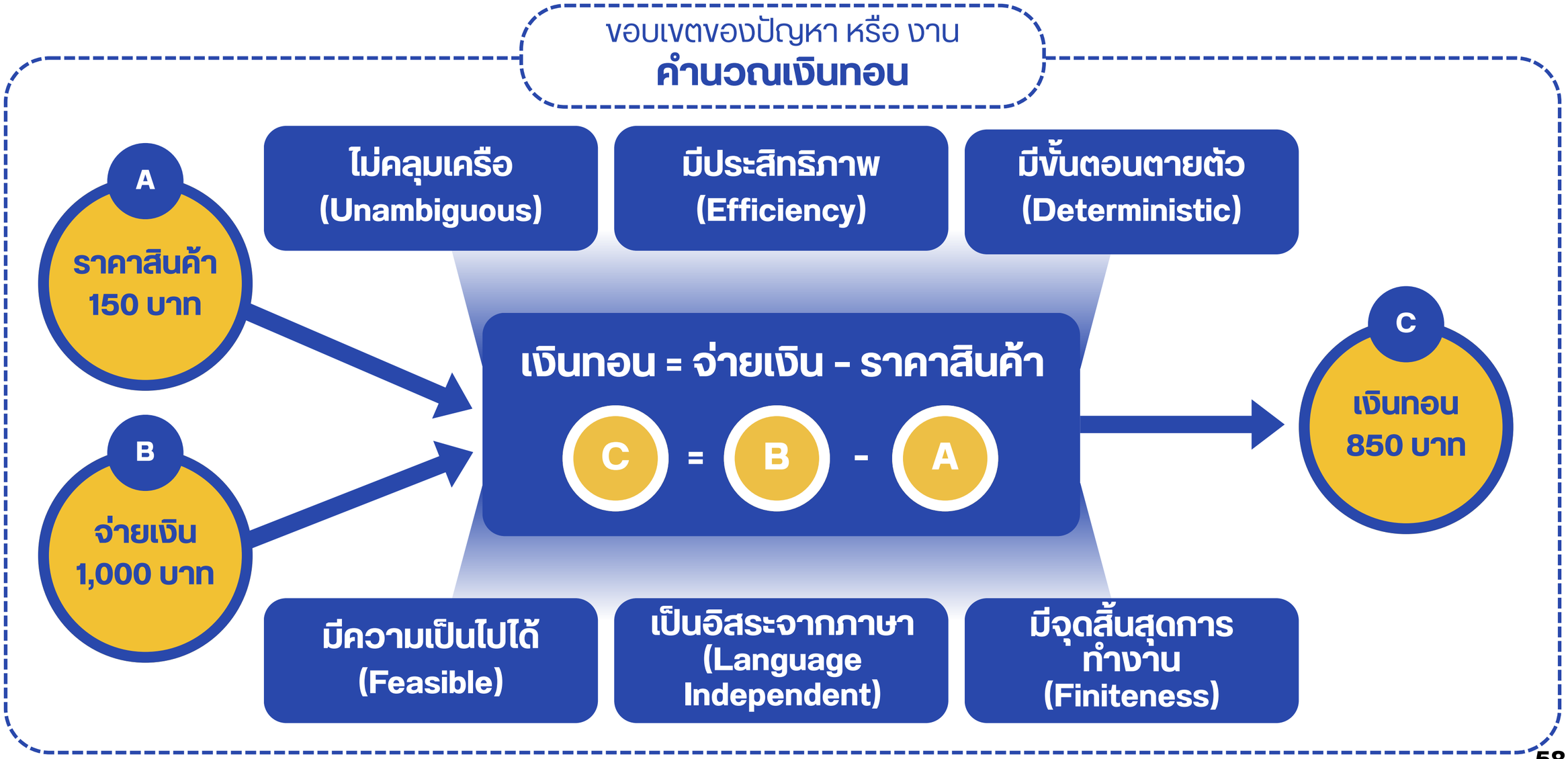
อ้างอิง

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

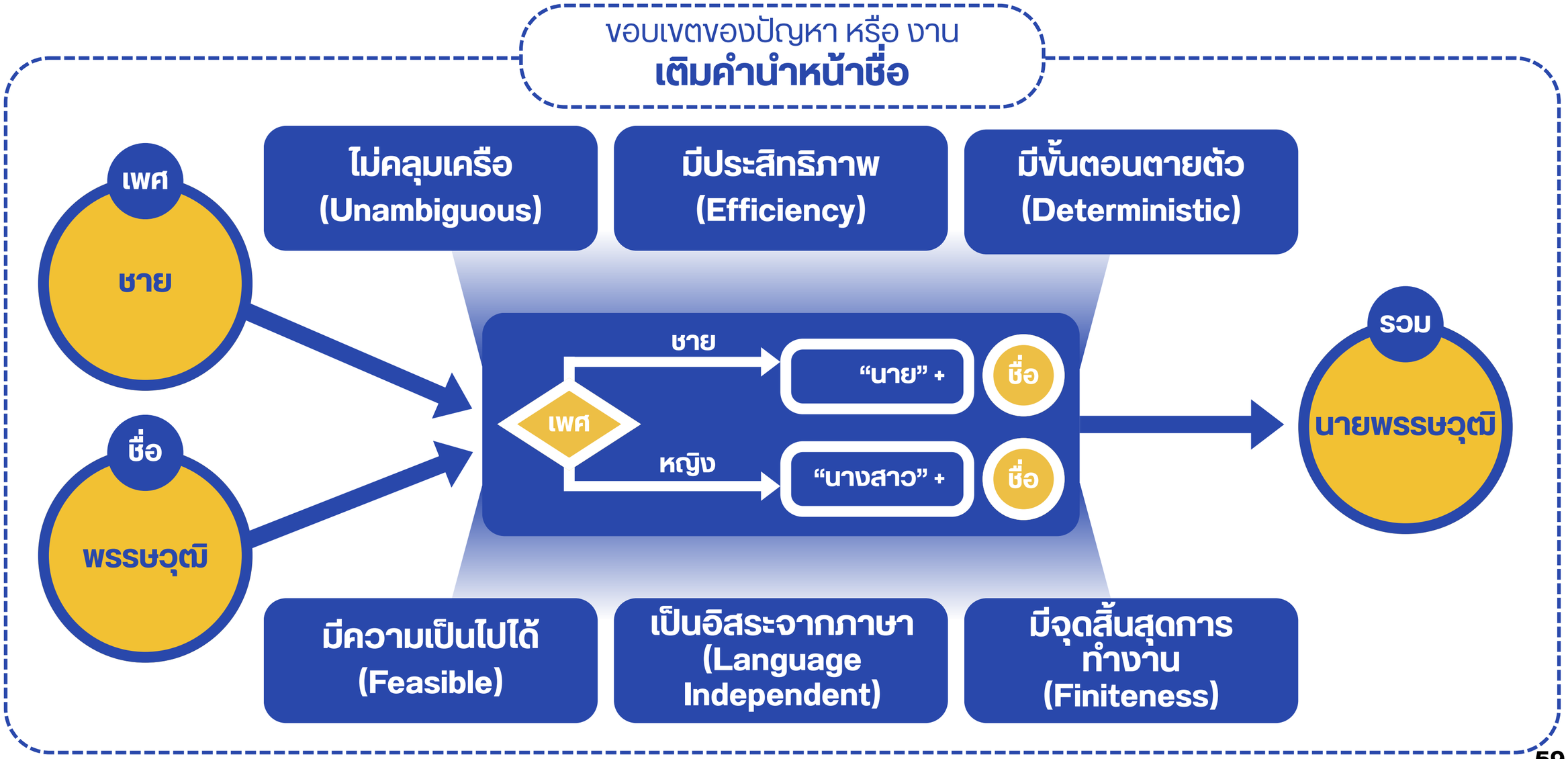
คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



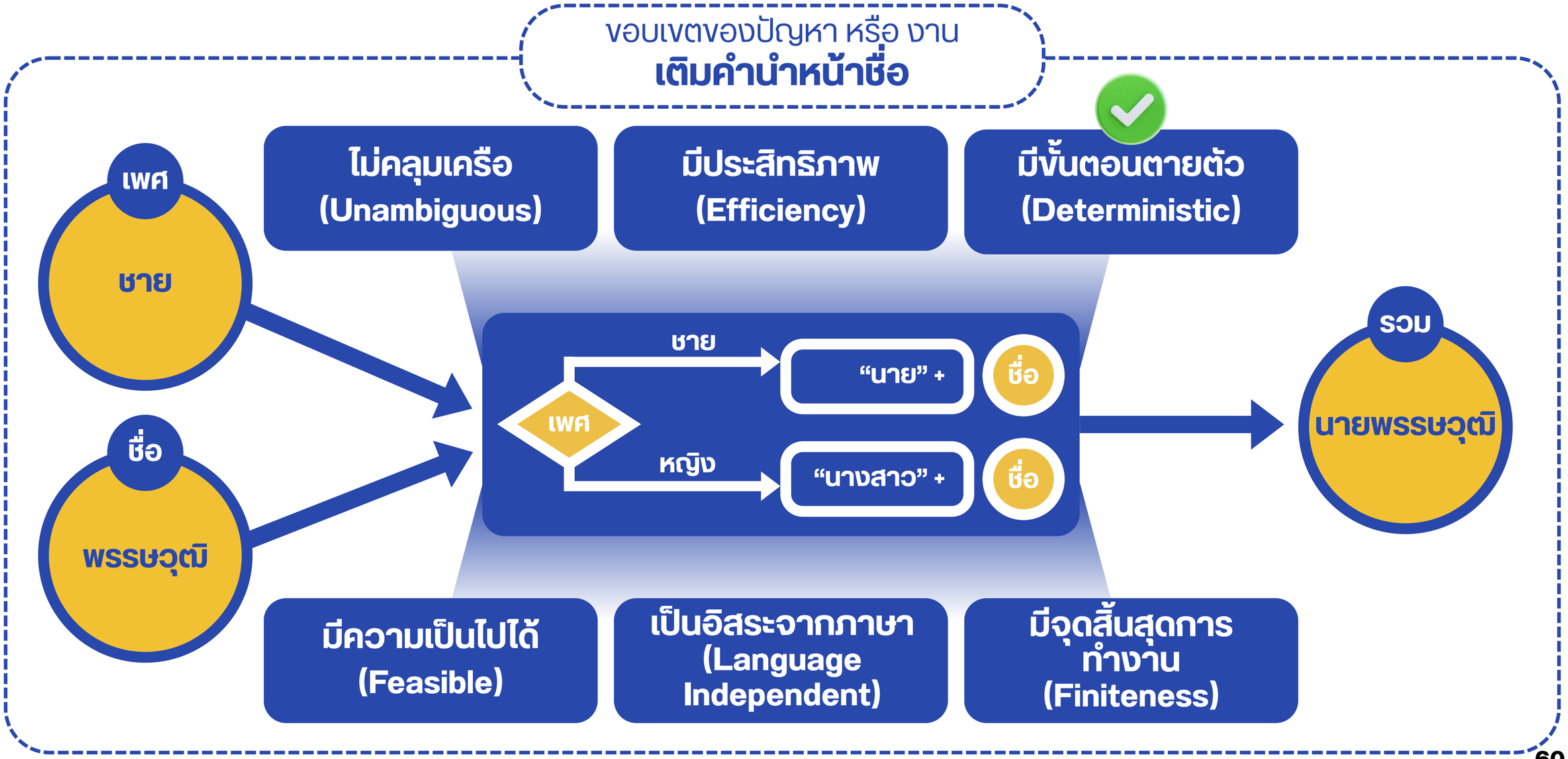
คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



คุณลักษณะของอัลกอริทึม (Characteristics of an Algorithm)



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	3. Process การประมวลผล	2. Output ผลผลิต
จำนวนที่ 1 จำนวนที่ 2	Operation Variable ตัวแปรดำเนินการ	

ต้องการผลรวมของค่า 2 จำนวน

1. Input การนำเข้า	3. Process การประมวลผล	2. Output ผลผลิต
จำนวนที่ 1 จำนวนที่ 2	Operation Variable ตัวแปรดำเนินการ	ผลลัพธ์

ต้องการผลรวมของค่า 2 จำนวน

1. Input การนำเข้า	3. Process การประมวลผล	2. Output ผลผลิต
จำนวนที่ 1 จำนวนที่ 2	Operation Variable ตัวแปรดำเนินการ 1. ผลลัพธ์ ← จำนวนที่ 1 + จำนวนที่ 2 2. แสดง ผลลัพธ์	ผลลัพธ์

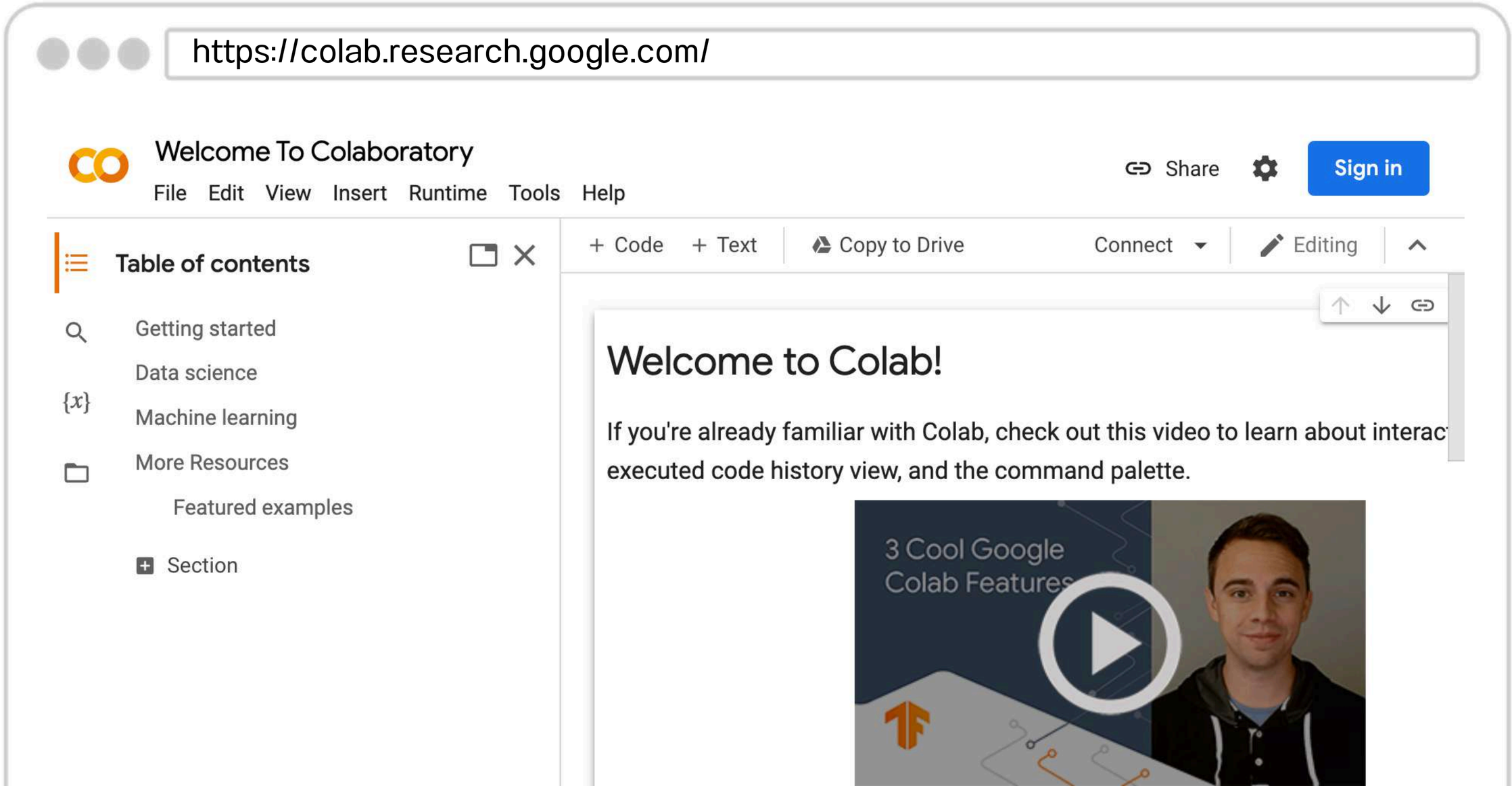
ต้องการคำนวณส่วนลดราคาสินค้า
ราคาสินค้า 200 บาท ส่วนลด 10% ต้องจ่ายกี่บาท

1. Input การนำเข้า	3. Process การประมวลผล	2. Output ผลผลิต
ราคาสินค้น ส่วนลด	Operation Variable ตัวแปรดำเนินการ 1. ส่วนลด 20 บาท ← ราคาสินค้า 200 X ส่วนลด 10 / 100 2. ราคาที่ต้องจ่าย 180 บาท ← ราคาสินค้า 200 - ส่วนลด 20 บาท 3. แสดง ราคาที่ต้องจ่าย 180 บาท ตัวแปร (Variable) เช่น ราคาสินค้า ส่วนลด ส่วน 50, 100 เป็นค่าของตัวแปร (Value)	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/



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 Sign in

Table of contents

Getting started

Data science

Machine learning

More Resources

Featured examples

Section

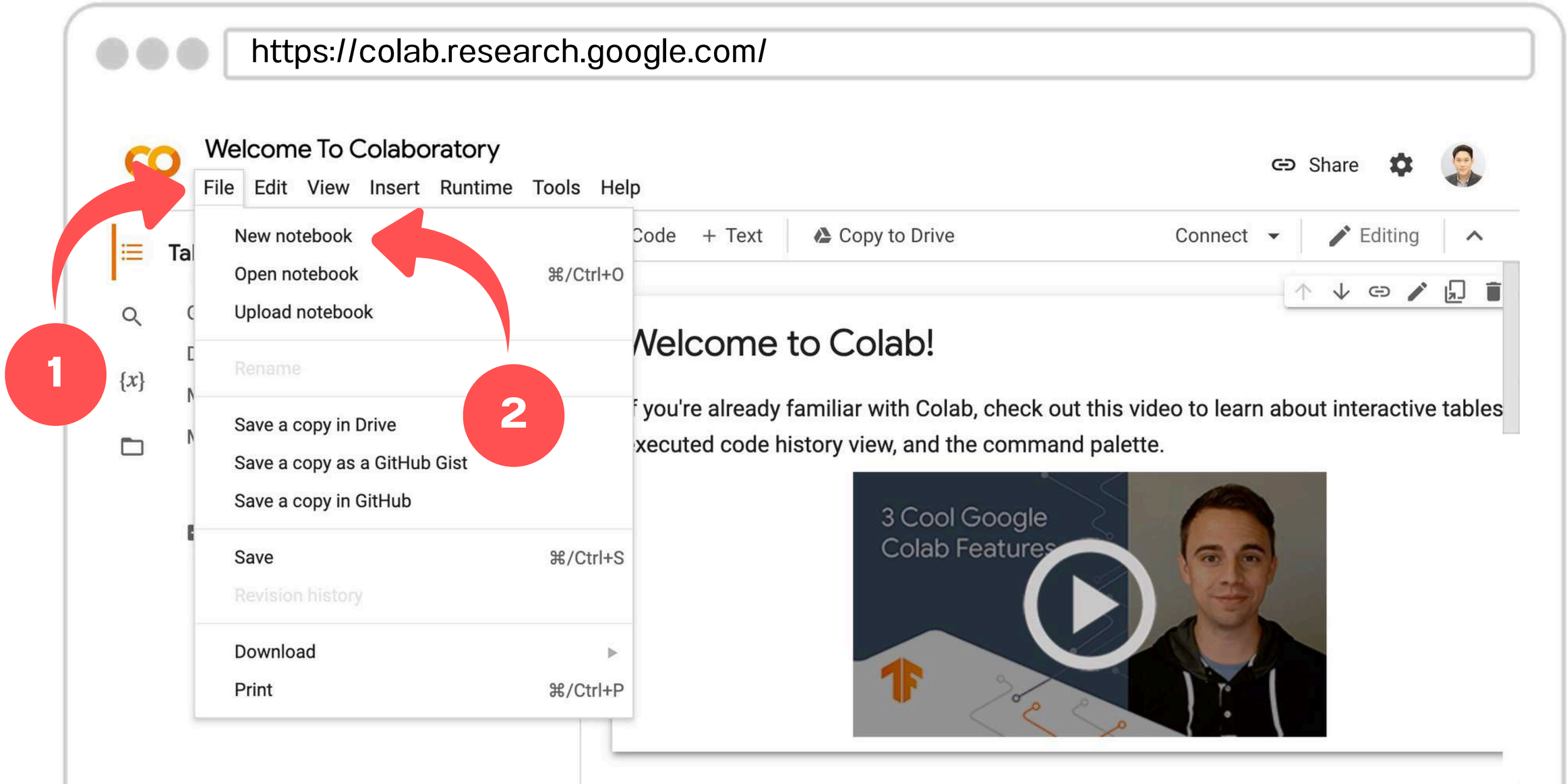
+ Code + Text Copy to Drive Connect Editing

Welcome to Colab!

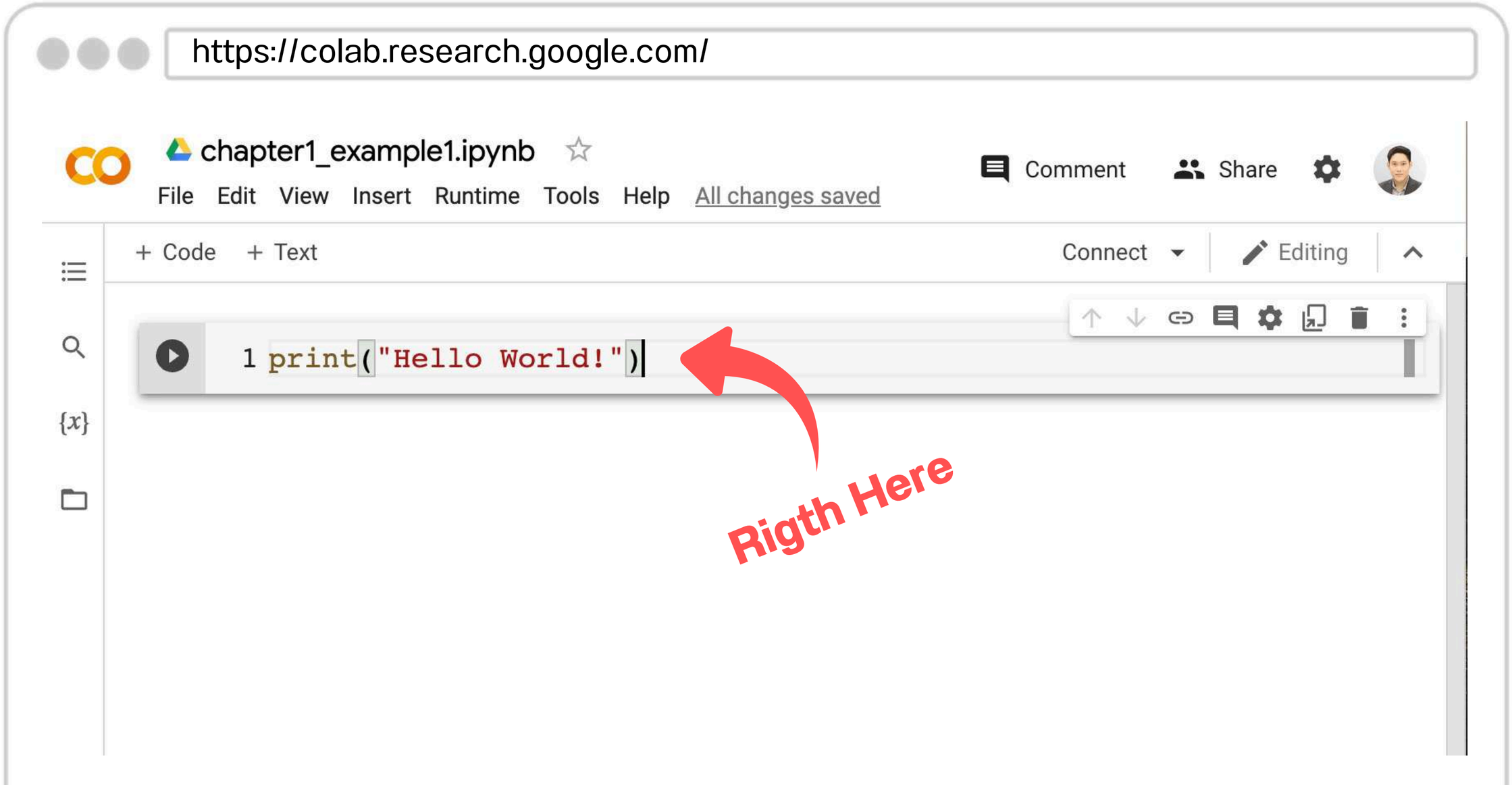
If you're already familiar with Colab, check out this video to learn about interac
executed code history view, and the command palette.

3. File -> New Notebook

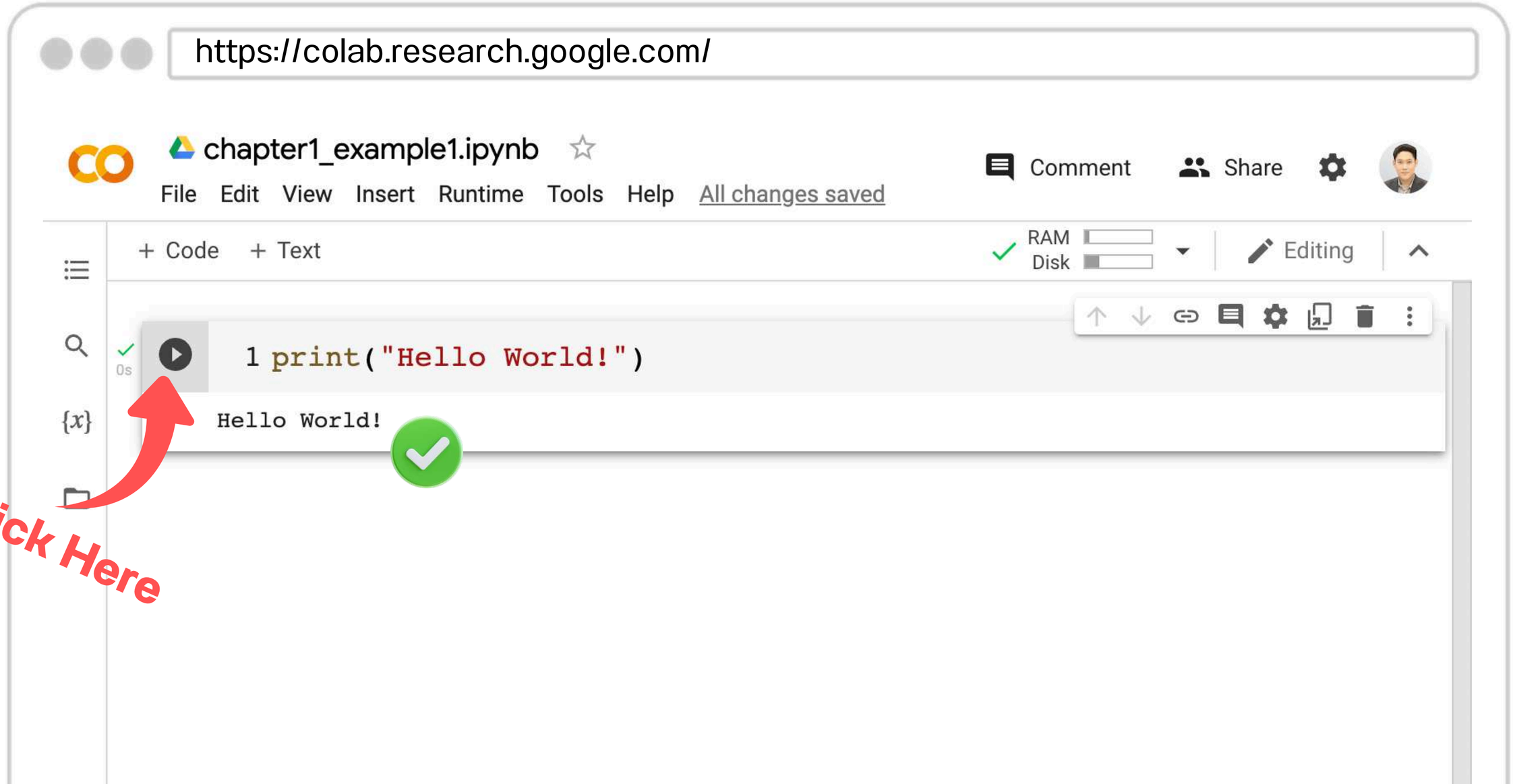
https://colab.research.google.com/



4. Coding with python commands on cell



5. Run cell


https://colab.research.google.com/
chapter1_example1.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
1 print("Hello World!")
0s
Hello World!
Click Here

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

Post 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. ”1101001010101010101” เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?

A Low-Level Language

B High-Level Language

C Assembly Language

D Procedural Language

Question:

4. print(“Panda”) เทียบเคียงได้กับภาษาคอมพิวเตอร์ระดับใด?

A Low-Level Language

B High-Level Language

C Assembly Language

D Procedural Language

Question:

5. ข้อใดไม่ใช่คุณลักษณะของอัลกอริทึม

A ไม่คลุมเครือ (Unambiguous)

B มีขั้นตอนตายตัว (Deterministic)

C มีจุดสิ้นสุดการทำงาน (Finiteness)

D มีหน่วยความจำที่เพียงพอ (Memory Capacity)

Question:

6. ข้อใดไม่ใช่เครื่องมือในการโปรแกรมภาษา Python

A Pycham

B Colab

C Jupyter

D Assembler

Question:

7. ให้ผลลัพธ์ถูกต้องตามเงื่อนไขที่กำหนดไว้เสมอ คือ คุณลักษณะข้อใด?

A ไม่คลุมเครือ (Unambiguous)

B มีขั้นตอนตายตัว (Deterministic)

C มีจุดสิ้นสุดการทำงาน (Finiteness)

D มีหน่วยความจำที่เพียงพอ (Memory Capacity)

Question:

8. วิธีการสอนใดเหมาะกับการเรียนรู้ด้านการโปรแกรมคอมพิวเตอร์

A Passivity-based Learning

B Cognitivism

C Experiential Learning

D Adaptive Learning

Question:

9. การสอน Coding สิ่งแรกที่ต้องสอนให้ผู้เรียนคำนึงถึงในการโปรแกรม คืออะไร?

A Input

C Output

B Process

D Memory

Question:

10. การสอน Coding สิ่งสุดท้ายที่ต้องสอนให้ผู้เรียนคำนึงถึงในการโปรแกรม คืออะไร?

A Input

C Output

B Process

D Memory

Computer Programming and
Developing Applications for Education

Thank You

Pasawut Cheerapakorn
Suan Sunandha Rajabhat University

DTI3302 Computer Programming and Developing Applications for Education
Department of Digital Technology for Education
Faculty of Education, Suan Sunandha Rajabhat University