

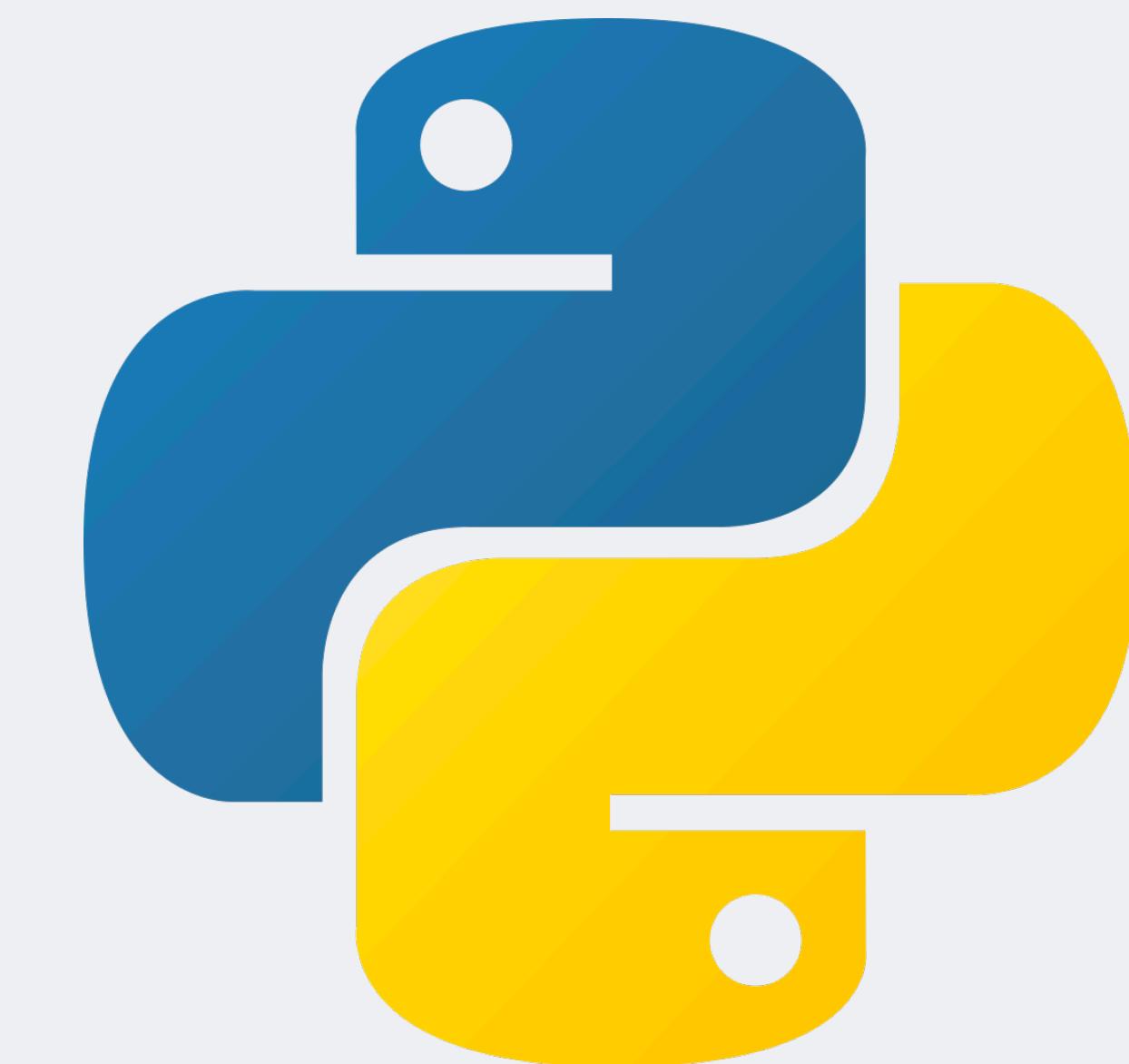
# INE-PYTHON

## LECTURE - 02 DATA TYPES, VARIABLES AND EXPRESSION

### What You Will Learn

- Computer and Programming
- Program Design and IPO (Input Process Output)
- Variable, Data Type
- Numbers, Boolean, String
- Python Operators
- Input
- Comments

# INTRODUCTION



## Program Design

# PROGRAM DEVELOPMENT CYCLE

วงจรการพัฒนาโปรแกรมคือกระบวนการที่มีโครงสร้างชัดเจน เพื่อช่วยให้นักพัฒนาเขียนโปรแกรมอย่างเป็นระบบ ซึ่งจะช่วยให้โปรแกรมที่พัฒนาขึ้นมีความน่าเชื่อถือและมีประสิทธิภาพมากยิ่งขึ้น ขั้นตอนหลักของวงจรการพัฒนาโปรแกรมมีดังนี้:

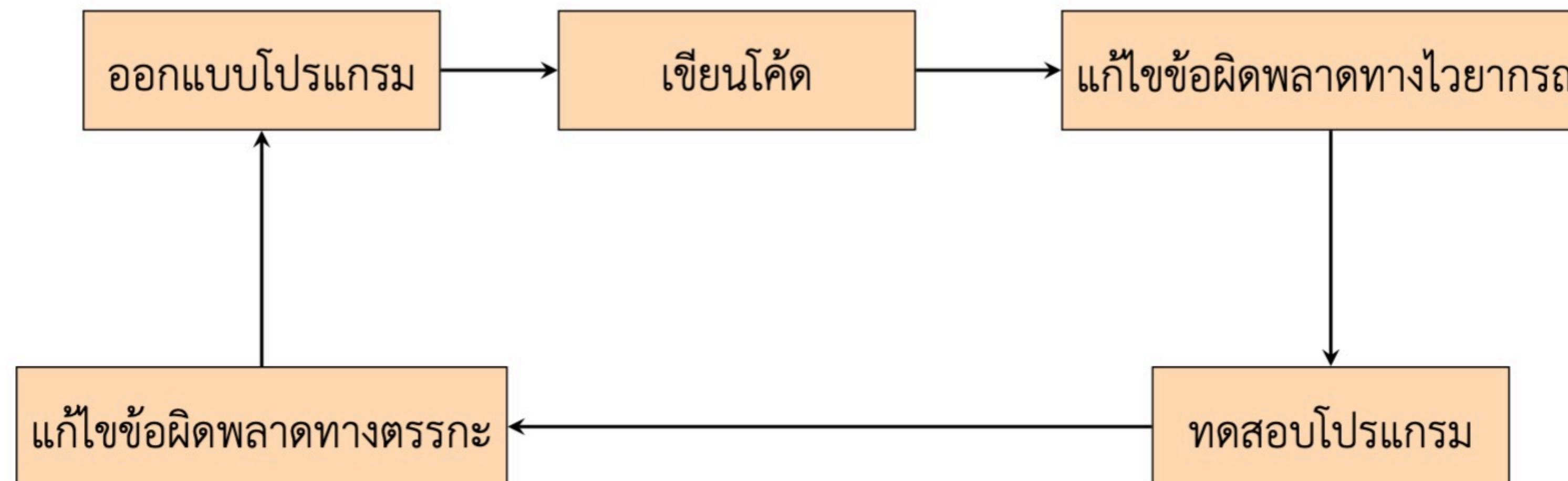


Figure 2.1: วงจรการพัฒนาโปรแกรม

## DESIGNING A PROGRAM

การออกแบบโปรแกรมถือเป็นส่วนที่สำคัญที่สุด ในวงจรการพัฒนาโปรแกรมก็ว่าได้ คุณสามารถมองว่า การออกแบบโปรแกรมคือ "รากฐาน" ของโปรแกรม หากคุณสร้างบ้านบนรากฐานที่ไม่แข็งแรง ในที่สุดคุณก็จะต้องเสียเวลาและแรงงานจำนวนมากในการซ่อมแซมบ้านนั้น! การออกแบบโปรแกรมก็เช่นเดียวกัน หากออกแบบโปรแกรมอย่างไม่เหมาะสม สุดท้ายคุณก็จะต้องเสียเวลาและแรงงานในการแก้ไข โปรแกรม เช่นกัน

กระบวนการในการออกแบบ โปรแกรมสามารถสรุปได้เป็นสองขั้นตอน ดังนี้:

- ทำความเข้าใจงานหรือหน้าที่ที่โปรแกรมต้องดำเนินการ
- กำหนดขั้นตอนที่ต้องปฏิบัติเพื่อให้สามารถดำเนินงานนั้นได้สำเร็จ

## TASK AND STEPS EXAMPLE

นักเขียนโปรแกรมจะทำการแยกย่อยงานที่โปรแกรมต้องดำเนินการออกเป็นส่วน ๆ แล้วจึงสร้างอัลกอริทึมขึ้นมา ซึ่งเป็นการลำดับขั้นตอนทางตรรกะทั้งหมดที่จำเป็นต้องดำเนินการ ตัวอย่างเช่น สมมติว่าคุณได้รับมอบหมายให้เขียน **โปรแกรมคำนวนพื้นที่ของสี่เหลี่ยมผืนผ้า** ขั้นตอนที่คุณจะดำเนินการมีดังนี้:

1. รับค่าความกว้างของสี่เหลี่ยมผืนผ้า
2. รับค่าความสูงของสี่เหลี่ยมผืนผ้า
3. คูณความกว้างของสี่เหลี่ยมผืนผ้ากับความสูง
4. แสดงผลลัพธ์ของการคำนวนจากขั้นตอนที่ 3

แน่นอนว่า อัลกอริทึมนี้ยังไม่สามารถนำไปประมวลผลบนคอมพิวเตอร์ได้โดยตรง **ขั้นตอน** ในรายการนี้จะต้องแปลงออกมาเป็นโค๊ด ก่อน นักเขียนโปรแกรมมักจะใช้เครื่องมือสองอย่างเพื่อช่วยในการดำเนินการนี้ ได้แก่ รหัสเทียม (pseudocode) และ ผังงาน (flowchart) เราจะมาทำความเข้าใจแต่ละเครื่องมือให้ละเอียดมากขึ้นต่อไป

# PSEUDOCODE

ก่อนจะเริ่มเขียนโค้ดจริง การร่างลำดับตรรกะของโปรแกรมด้วยรหัสเทียมหรืออัลกอริทึม (pseudocode) เป็นสิ่งที่มีประโยชน์อย่างยิ่ง รหัสเทียมเป็นการเขียนขั้นตอนการทำงานของโปรแกรมในรูปแบบภาษาธรรมชาติ ช่วยในการวางแผนโครงสร้างและลำดับขั้นตอนของโปรแกรมโดยไม่ต้องกังวลเรื่องไวยากรณ์ ทำให้มั่นใจได้ว่าตรรกะของโปรแกรมถูกต้องและชัดเจน ตัวอย่างด้านล่างเป็นรหัสเทียมของโปรแกรมคำนวณพื้นที่สี่เหลี่ยมผืนผ้า:

---

## Algorithm 1: Algorithm to Calculate the Area of a Rectangle

---

**Input:** Width and height of the rectangle

**Output:** The area of the rectangle

1 **begin**

2     Print "Enter the width of the rectangle:";  
3     Read width;  
4     Print "Enter the height of the rectangle:";  
5     Read height;  
6     area = width \* height;  
7     Print "The area of the rectangle is", area;

---

# FLOWCHART

ผังงาน (Flowchart) เป็นอีกหนึ่งเครื่องมือที่นักเขียนโปรแกรมใช้ในการออกแบบโปรแกรม โดยผังงานคือแผนภาพที่แสดงขั้นตอนต่าง ๆ ของโปรแกรมในรูปแบบกราฟิก

ในผังงานจะมีสัญลักษณ์พื้นฐานอยู่ 3 ประเภทหลัก ได้แก่ วงรี (Ovals), รูปสี่เหลี่ยมด้านไม่เท่า (Parallelograms) และ สี่เหลี่ยมผืนผ้า (Rectangle) โดยแต่ละสัญลักษณ์ มีความหมายแตกต่างกันดังนี้:

- วงรี (Ovals) ที่ปรากฏอยู่ด้านบนและด้านล่างของผังงาน เรียกว่า สัญลักษณ์จุดเริ่มต้นและจุดสิ้นสุด (Terminal Symbols)
  - วงรีที่เขียนว่า “Start” หมายถึง จุดเริ่มต้นของโปรแกรม
  - วงรีที่เขียนว่า “End” หมายถึง จุดสิ้นสุดของโปรแกรม
- รูปสี่เหลี่ยมด้านไม่เท่า (Parallelograms) ใช้แทน การรับข้อมูลเข้า (Input) หรือ การแสดงผลลัพธ์ (Output)
  - เป็นขั้นตอนที่โปรแกรมรับค่าจากผู้ใช้หรือแสดงค่าผลลัพธ์ออกมา
- สี่เหลี่ยมผืนผ้า (Rectangles) ใช้แทน การประมวลผล (Processing)
  - เป็นขั้นตอนที่โปรแกรมทำการคำนวณหรือดำเนินการใด ๆ กับข้อมูลที่รับเข้ามา เช่น การคูณ การบวก หรือการเปลี่ยนแปลงค่า

การใช้ผังงานช่วยให้สามารถมองภาพรวมของกระบวนการในโปรแกรมได้ชัดเจนและเข้าใจง่ายยิ่งขึ้น โดยเฉพาะในขั้นตอนการออกแบบก่อนเขียนโค้ด

# FLOWCHART

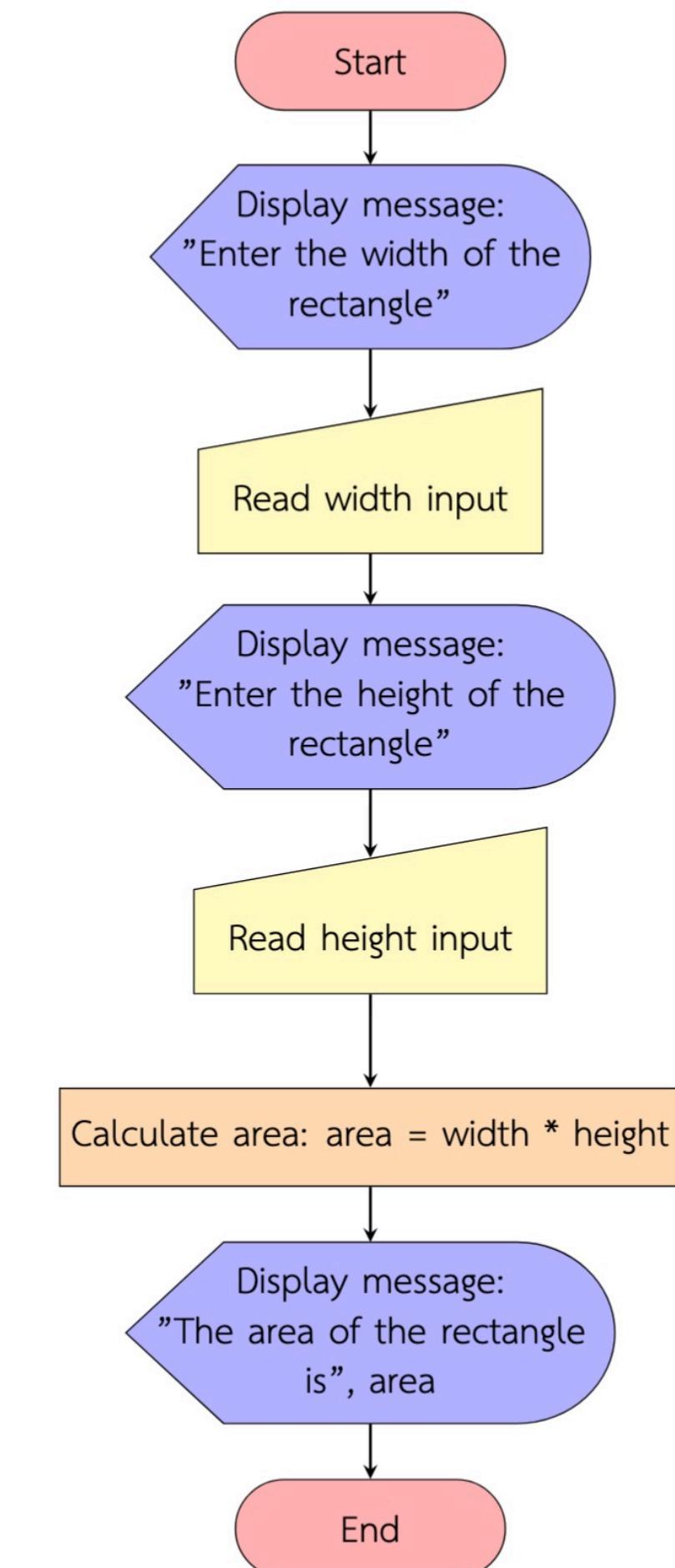


Figure 2.2: ผังงานของโปรแกรมคำนวณพื้นที่สี่เหลี่ยมผืนผ้า

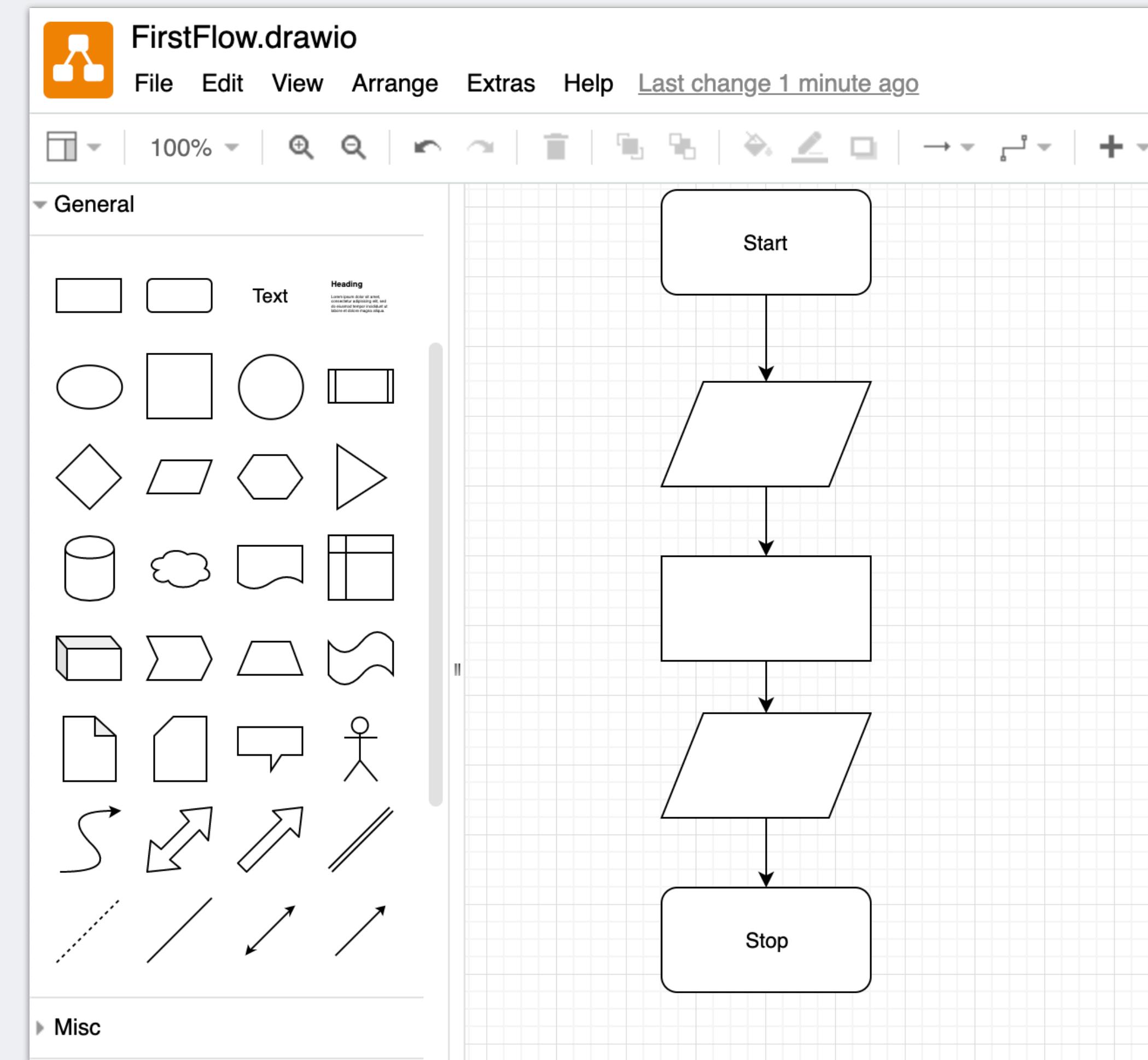
---

DRAW.IO

---



# DRAW.IO



# FLOWCHART

Search Extensions in Mark...

INSTALLED 51

- Draw.io Integration** This unofficial extension... Henning Dieterichs
- ESLint** Integrates ESLint JavaS... Microsoft
- Git Graph** View a Git Graph of your... mhutchie
- GitHub Classroom** Browse, edit and collabo... GitHub
- GitHub Copilot** Your AI pair programmer GitHub
- GitHub Pull Requests** GitHub



## Draw.io Integration v1.6.6

Henning Dieterichs | ⚡ 1,463,485 | ★★★★★ (125)

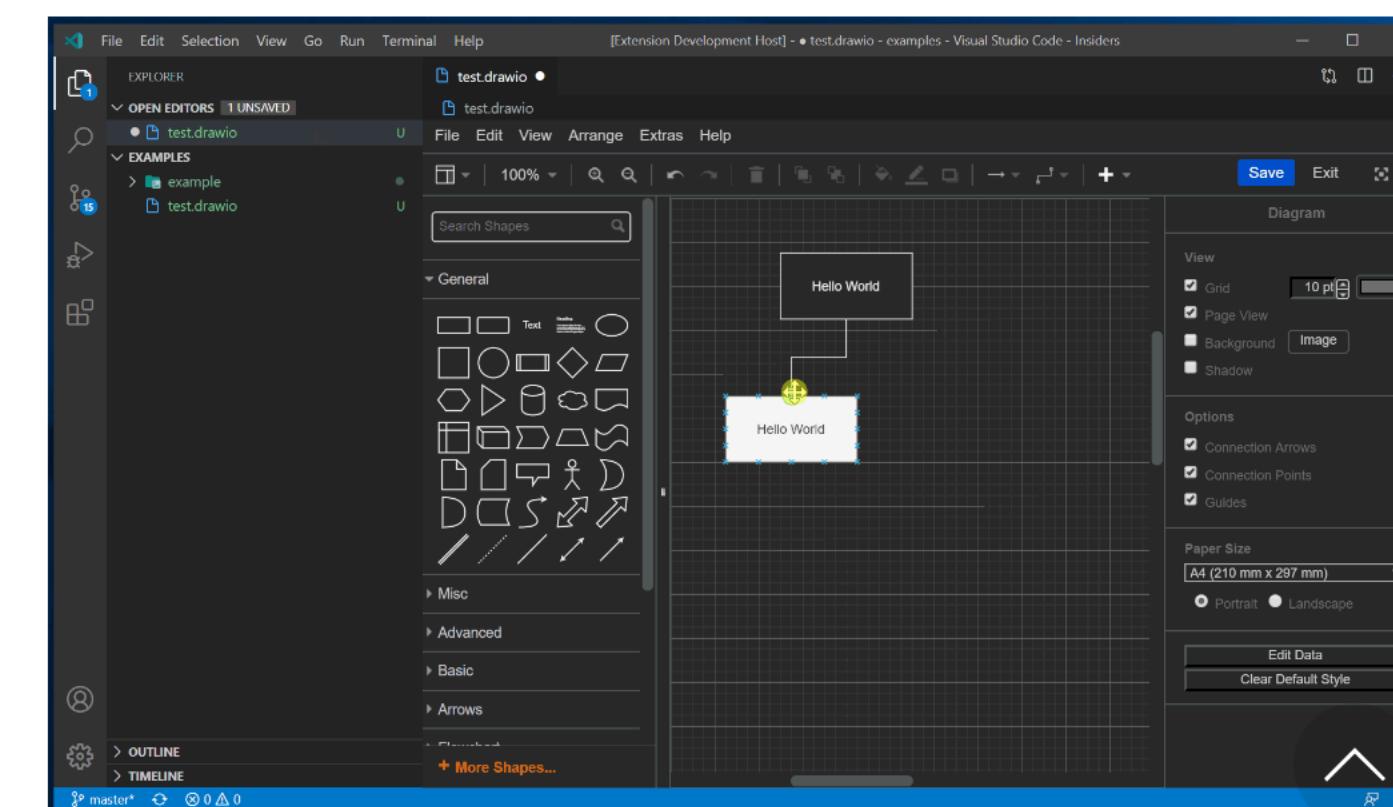
This unofficial extension integrates Draw.io into VS Code.

[Disable](#) | [Uninstall](#) | [Settings](#)

This extension is enabled globally.

DETAILS FEATURE CONTRIBUTIONS CHangelog RUNTIME STATUS

Demo



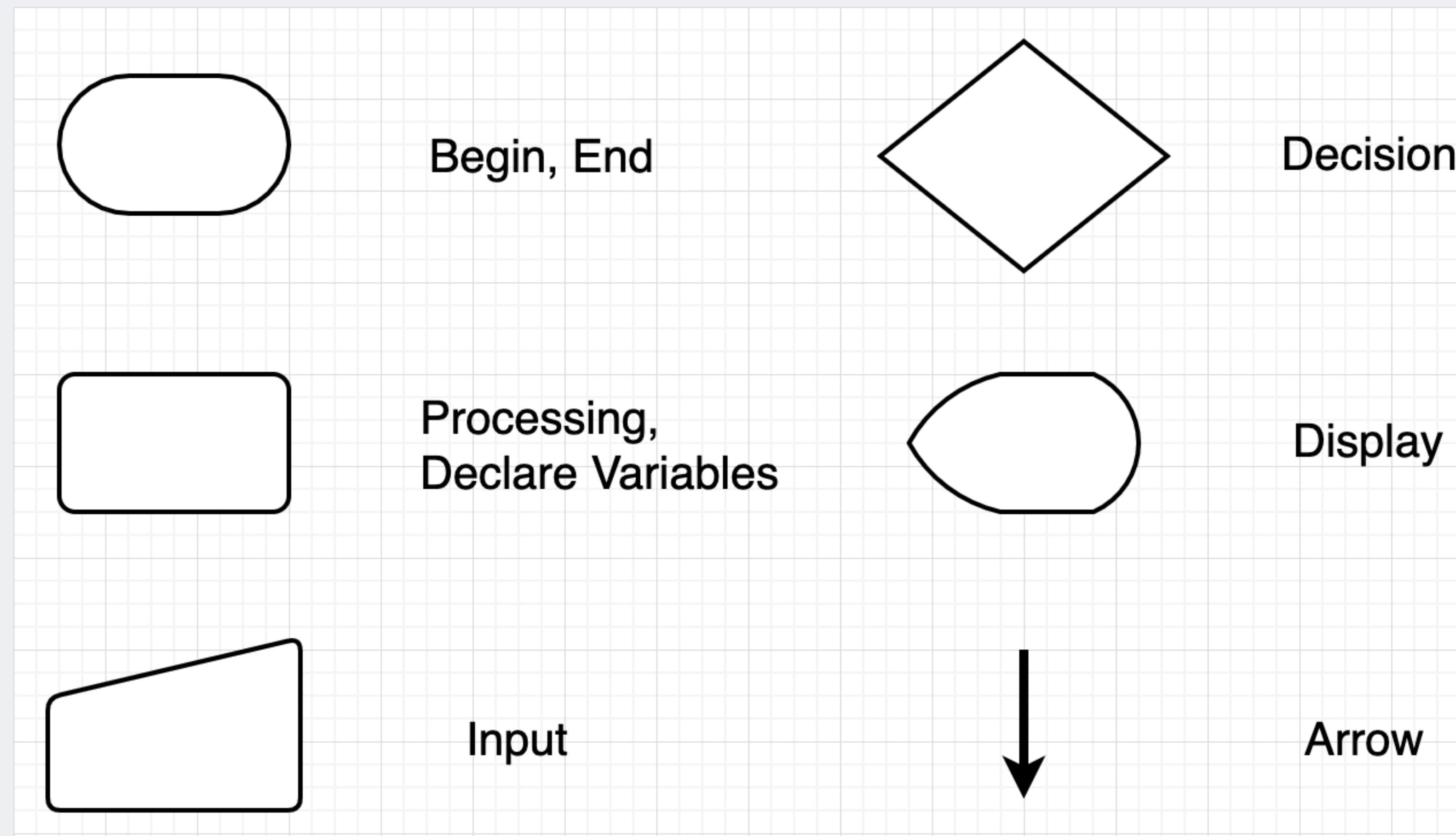
Categories

Visualization

Extension Resources

Marketplace Repository Henning Dieterichs

# FLOWCHART SYMBOLS



# IPO



Input Process Output

# INPUT PROCESS OUTPUT

โปรแกรมคอมพิวเตอร์มักดำเนินการตามกระบวนการ 3 ขั้นตอนหลัก ดังนี้:

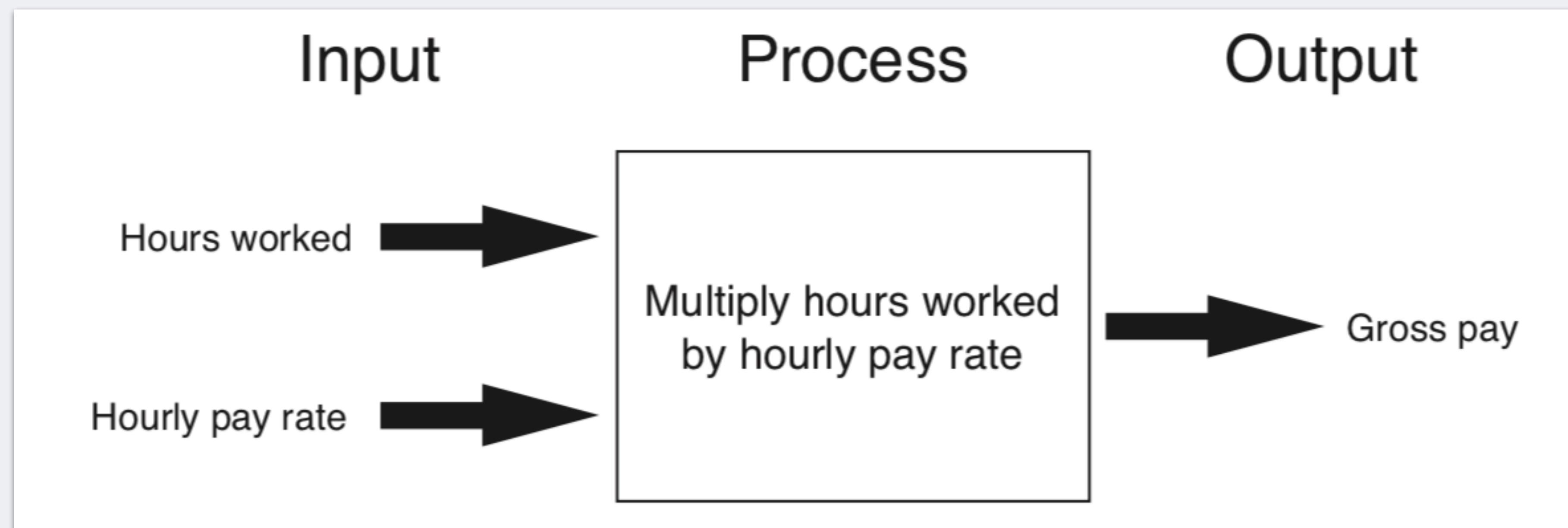
1. รับข้อมูลเข้า (Input)
2. ประมวลผลข้อมูล (Processing)
3. แสดงผลลัพธ์ (Output)

Input หมายถึง ข้อมูลใด ๆ ที่โปรแกรมได้รับ ในระหว่างการทำงาน ตัวอย่างรูปแบบของข้อมูลเข้าที่พบบ่อยคือ ข้อมูลที่ผู้ใช้พิมพ์ผ่านแป้นพิมพ์ เมื่อโปรแกรมได้รับข้อมูลเข้าแล้ว ก็มักจะมีการประมวลผลบางอย่างกับข้อมูลนั้น เช่น การคำนวณทางคณิตศาสตร์ จากนั้นผลลัพธ์ของการประมวลผลก็จะถูกส่งออกมาเป็น Output หรือผลลัพธ์ของโปรแกรมนั้นเอง

# INPUT PROCESS OUTPUT

Table 2.1: ตัวอย่างแผนภูมิ IPO

Input	Process	Output
Width of the rectangle	Multiply width by height	Area of the rectangle
Height of the rectangle		



# VARIABLES



# VARIABLES

## ตัวแปร (Variable) คือ:

- ตำแหน่งเก็บข้อมูลในหน่วยความจำที่มีชื่อกำกับ
- เป็นคู่ของชื่อและค่า (Name-Value Pair) ซึ่งช่วยให้โปรแกรมสามารถเรียกใช้งานหรือเปลี่ยนค่าที่จัดเก็บไว้ได้อย่างสะดวก

กล่าวง่าย ๆ คือ ตัวแปรทำหน้าที่เหมือนกล่องที่มีชื่อรับไว้ภายนอก ภายในกล่องเก็บค่าบางอย่างไว้ ซึ่งค่านั้นสามารถเปลี่ยนแปลงได้ตามการทำงานของโปรแกรม

### ตัวอย่างการประกาศและกำหนดค่าตัวแปร:

```
1 # Declaring variables
2 age = 25
3 name = "Alice"
4 height = 5.6
```

Listing 2.2: Example of Declaring and Assigning Variables in Python

### Note:

\* Pick variable name that represent data that variable will hold

# VARIABLES

Table 2.3: ตัวอย่างการตั้งชื่อในรูปแบบต่าง ๆ

Purpose	PascalCase	snake_case
Class name	CustomerAccount	—
Variable name	—	customer_account
Function name	—	calculate_total()
Constant name	—	MAX_VALUE
File name	—	user_profile.py

หมายเหตุ: ตามแนวทาง PEP 8 ซึ่งเป็นมาตรฐานการเขียนโค้ดของภาษา Python แนะนำให้ใช้ snake\_case สำหรับชื่อตัวแปรและฟังก์ชัน และใช้ PascalCase สำหรับชื่อคลาส เพื่อให้โค้ด อ่านง่ายและมีความสอดคล้องกันในชุมชนผู้ใช้ Python การใช้ camelCase เมمจะไม่ผิดทางไวยากรณ์ แต่ถือว่าไม่สอดคล้องกับธรรมเนียมของภาษา Python และควรหลีกเลี่ยงเว้นแต่จำเป็นต้องใช้งานร่วม กับไลบรารีจากภาษาที่ใช้รูปแบบดังกล่าว (เช่น JavaScript)

# VARIABLES

## Reserved Word

and	except	lambda	with
as	finally	nonlocal	while
assert	FALSE	None	yield
break	for	not	await
class	from	or	async
continue	global	pass	
def	if	raise	
del	import	return	
elif	in	TRUE	
else	is	try	

\*\*\* Function name also not recommend

# VARIABLES

## Reserved Word

```
for = 10
```

```
File "<ipython-input-68-597287a2d762>", line 1
```

```
  for = 10
      ^
```

```
SyntaxError: invalid syntax
```

\*\*\* Function name also not recommend

# BEST PRACTICES FOR NAMING VARIABLES

- ตั้งชื่อให้สื่อความหมาย:
  - ใช้ชื่อที่บอกถึงหน้าที่หรือเนื้อหาของตัวแปร
  - ตัวอย่าง: counter แทนที่จะใช้ c, total\_sum แทน ts
- ใช้ `snake_case` สำหรับชื่อที่มีหลายคำ (นิยมในภาษา Python):
  - ใช้ขีดล่างเชื่อมคำทั้งหมดเป็นตัวพิมพ์เล็ก
  - ตัวอย่าง: user\_name, total\_amount

```
1 def calculate_total_price(product_price, tax_rate):  
2     total_price = product_price + (product_price * tax_rate)  
3     return total_price
```

Listing 2.3: ตัวอย่างการใช้ `snake_case` ใน Python

Note:

\* Pick variable name that represent data that variable will hold

# BEST PRACTICES FOR NAMING VARIABLES

- ใช้ PascalCase สำหรับชื่อคลาสหรือชนิดข้อมูล:
  - ตัวอักษรตัวแรกของทุกคำเป็นพิมพ์ใหญ่ ไม่มีตัวคั่น
  - นิยมใช้ในภาษา C#, Java, Python (ชื่อคลาส)
  - ตัวอย่าง: CustomerAccount, TotalAmount

```
1 class CustomerAccount:  
2     def __init__(self, name, balance):  
3         self.name = name  
4         self.balance = balance
```

Listing 2.4: ตัวอย่างการใช้ PascalCase สำหรับชื่อคลาส

Note:

\* Pick variable name that represent data that variable will hold

# VARIABLES

```
1 username = "john_doe"
2 total_price = 150.75
3 _is_valid = True
4 user1 = "Alice"
5 max_value = 100
```

Listing 2.5: 例ตัวอย่างชื่อตัวแปรที่ถูกต้อง

Note:

\* Pick variable name that represent data that variable will hold

# VARIABLES

```
1 variable = 10          # Starts with a digit
2 user-name = "Alice"    # Contains a hyphen
3 total$amount = 100.0   # Contains a special character ($)
4 class = "Math"         # Reserved keyword
```

Listing 2.6: ตัวอย่างชื่อตัวแปรที่ไม่ถูกต้อง

Note:

\* Pick variable name that represent data that variable will hold

# DATA TYPE



# DATA TYPE

- Basic Data Types

- Numeric
  - Integer
  - Floating point
- Boolean
- String

- Composite Data Types

- List
- Tuple
- Dictionary
- Set



Later lessons

# DATA TYPE

## 1. ชนิดตัวเลข (Numeric Types)

- **Integer (int)**: ตัวเลขจำนวนเต็ม ไม่รวมทศนิยม

```
1 age = 25
```

- **Float (float)**: ตัวเลขที่มีจุดทศนิยม

```
1 height = 5.6
```

- **Complex (complex)**: จำนวนเชิงซ้อน มีทั้งส่วนจริงและส่วนจินตภาพ

```
1 complex_num = 3 + 4j
```

# DATA TYPE

## 2. ชนิดลำดับข้อมูล (Sequence Types)

- **String (str):** ลำดับของตัวอักษร

```
1 name = "Alice"  
2 class_name = 'Computer Programming'
```

- **List (list):** ชุดข้อมูลที่เรียงลำดับ สามารถเปลี่ยนแปลงค่าได้

```
1 fruits = ["apple", "banana", "cherry"]  
2 mix_list = ['car', 3, "mango", 7.5]
```

- **Tuple (tuple):** ชุดข้อมูลที่เรียงลำดับ แต่ไม่สามารถเปลี่ยนแปลงค่าได้

```
1 coordinates = (10, 20)
```

# DATA TYPE

## 3. ชนิดข้อมูลบูลีน (Boolean Type)

- Boolean (bool): ใช้แทนค่า True หรือ False

```
1 is_student = True
```

## 4. ชนิดเซต (Set Type)

- Set (set): ชุดข้อมูลที่ไม่เรียงลำดับและไม่มีค่าซ้ำ

```
1 my_set = {1, 2, 3, 4}
2 another_set = {'apple', 'banana', 'cherry'}
```

## 5. ชนิดดิกชันนารี (Dictionary Type)

- Dictionary (dict): เก็บข้อมูลในรูปแบบคู่ key-value

```
1 student = {"name": "Alice", "age": 25}
```

# DATA TYPE

Table 2.4: ชนิดข้อมูลที่ใช้บ่อยในภาษา Python พร้อมตัวอย่าง

ชนิดข้อมูล (Data Type)	คำอธิบาย	ตัวอย่างโค้ด
<code>int</code> (จำนวนเต็ม)	ตัวเลขที่ไม่มีจุดทศนิยม	<code>x = 10</code>
<code>float</code> (ทศนิยม)	ตัวเลขที่มีจุดทศนิยม	<code>pi = 3.14</code>
<code>str</code> (สตริง)	ข้อความหรืออักขระ	<code>name = "Alice"</code>
<code>bool</code> (ตรรกะ)	ค่า <code>True</code> หรือ <code>False</code>	<code>is_valid = True</code>
<code>list</code> (ลิสต์)	ลำดับข้อมูลที่แก้ไขได้	<code>numbers = [1, 2, 3]</code>
<code>tuple</code> (ทูเพิล)	ลำดับข้อมูลที่แก้ไขไม่ได้	<code>point = (10, 20)</code>
<code>dict</code> (딕ชันนารี)	คู่คี่ย์กับค่า (Key-Value)	<code>user = {"name": "Bob", "age": 25}</code>
<code>set</code> (เซต)	ชุดข้อมูลที่ไม่ซ้ำกัน	<code>colors = {"red", "blue", "green"}</code>

# TYPE CONVERSION



## NUMBERS

- Use numbers directly in your source code
  - Do not use quotation marks as they are for strings.

```
integer = 42  
float = 4.2  
a = b = c = d = 0.0
```

Note:

\* Pick variable name that represent data that variable will hold

# TYPE CONVERSION

ในภาษา Python ความสามารถในการแปลงชนิดของข้อมูล (Type Conversion) โดยใช้ฟังก์ชันในตัว (built-in functions) ช่วยเพิ่มความยืดหยุ่นในการประมวลผลข้อมูล ฟังก์ชันเหล่านี้ช่วยให้สามารถแปลงข้อมูลจากชนิดหนึ่งไปเป็นอีกชนิดหนึ่งได้อย่างราบรื่น ไม่ว่าจะเป็นการแปลงจากจำนวนเต็มเป็นทศนิยม จากสตริงเป็นจำนวนเต็ม หรือในทางกลับกัน การแปลงเหล่านี้ช่วยให้โค้ดมีความยืดหยุ่นและซัดเจนมากขึ้น ส่งเสริมแนวทางการเขียนโปรแกรมที่มีประสิทธิภาพ และสามารถประยุกต์ใช้กับโจทย์ที่หลากหลายได้อย่างมีประสิทธิผล

```
1 # Converting int to float
2 age = 25
3 height = float(age)
4
5 # Converting float to int
6 height = 5.6
7 age = int(height)
8
9 # Converting string to int
10 num_str = "123"
11 num = int(num_str)
```

Listing 2.7: Example of Python type conversion

# ARITHMETIC OPERATORS



# ARITHMETIC OPERATORS

ภาษา Python เป็นภาษาการเขียนโปรแกรมที่มีความยืดหยุ่นและทรงพลัง โดยมีตัวดำเนินการทางคณิตศาสตร์หลากหลายที่ใช้ในการดำเนินการคำนวณทั้งระดับพื้นฐานและขั้นสูง การเข้าใจตัวดำเนินการเหล่านี้ถือเป็นสิ่งสำคัญสำหรับผู้ที่ใช้ Python ไม่ว่าจะเพื่อการคำนวณง่าย ๆ หรือการวิเคราะห์ข้อมูลที่ซับซ้อน ตัวดำเนินการหลักใน Python ได้แก่ การบวก การลบ การคูณ การหาร การหารเอาเศษ การยกกำลัง และการหารปัดเศษลง

$$y = x^2 + 2x + 1$$

$y, x, 1$	คือ ตัวถูกดำเนินการ (Operand)
$=, +, \text{exponent}$	คือ ตัวดำเนินการ (Operator)
$y = x^2 + 2x + 1$	คือ นิพจน์ (Expression)

# ARITHMETIC OPERATORS

Table 2.5: การเปรียบเทียบตัวดำเนินการทางคณิตศาสตร์

Operator	ความหมาย	ตัวอย่าง
+	การบวกค่าของสองตัวแปร หรือบวกค่าบวก	$x + y, +2$
-	การลบค่าของตัวแปรด้านขวาออกจากด้านซ้าย หรือค่าลบ	$x - y, -2$
*	การคูณตัวแปรสองตัว	$x * y$
/	การหาร ตัวแปร ด้านซ้าย ด้วย ด้านขวา (ผลลัพธ์เป็นทศนิยมเสมอ)	$x/y$
%	การหารเอาเศษ (modulus)	$x \% y$
//	การหารแบบปัดเศษลง (floor division)	$x // y$
**	การยกกำลัง (exponentiation)	$x * * y$

# ARITHMETIC OPERATORS

```
x = 15
y = 4

# Output: x + y = 19
print('x + y =',x+y)

# Output: x - y = 11
print('x - y =',x-y)

# Output: x * y = 60
print('x * y =',x*y)

# Output: x / y = 3.75
print('x / y =',x/y)

# Output: x % y = 3
print('x % y =',x%y)

# Output: x // y = 3
print('x // y =',x//y)

# Output: x ** y = 50625
print('x ** y =',x**y)
```

```
x + y = 19
x - y = 11
x * y = 60
x / y = 3.75
x % y = 3
x // y = 3
x ** y = 50625
```

# OPERATOR PRECEDENCE

45 / 4

11.25

45 % 4

1

45 // 4

11

# OPERATOR PRECEDENCE

( )

\*\*

\* , / , % , //

+ , -

<= , < , > , >=

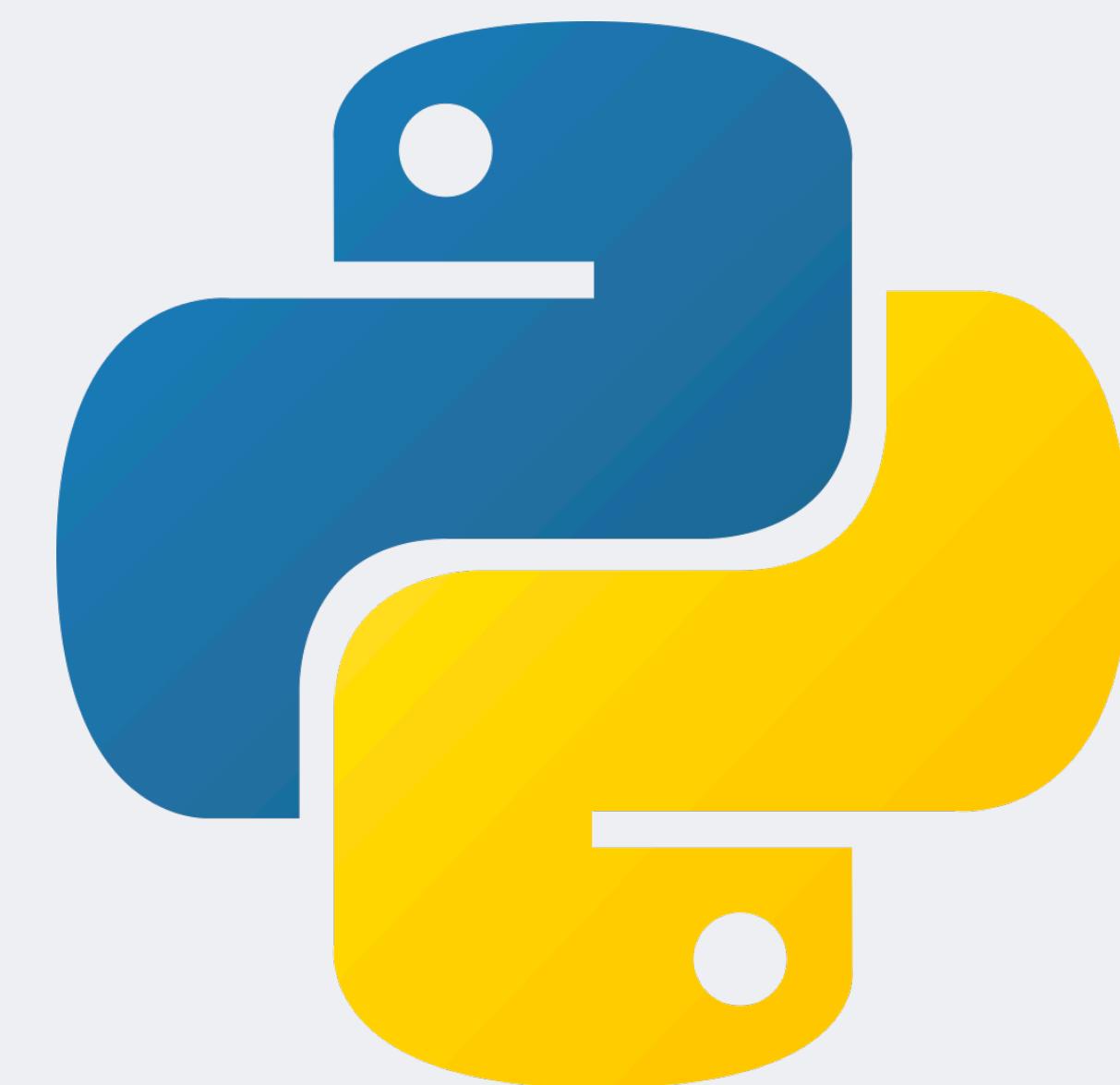
== , !=

=

# OPERATOR PRECEDENCE

$5+18//2-25\%4 = ?$

# INPUT



`input()`

## INPUT

โปรแกรมโดยทั่วไปมักจำเป็นต้องรับข้อมูลจากผู้ใช้ผ่านแป้นพิมพ์ เราจะใช้ฟังก์ชันในภาษา Python เพื่อทำสิ่งนี้ Python มีฟังก์ชันในตัวชื่อว่า `input()` สำหรับอ่านข้อมูลจากแป้นพิมพ์ ฟังก์ชันนี้จะรับค่าที่ผู้ใช้พิมพ์เข้ามา และส่งค่ากลับคืนไปยังโปรแกรมในรูปแบบของ สตริง (string)

โดยปกติจะใช้ฟังก์ชัน `input()` ควบคู่กับคำสั่งกำหนดค่า (assignment statement) ซึ่งมีรูปแบบทั่วไปดังนี้:

```
variable = input(prompt)
```

## INPUT

```
name = input('What is your name? ')
```

```
: # Get the user's first name.  
first_name = input('Enter your first name: ')  
  
# Get the user's last name.  
last_name = input('Enter your last name: ')  
  
# Print a greeting to the user.  
print('Hello', first_name, last_name)
```

```
Enter your first name: Anirach  
Enter your last name: Mingkhwan  
Hello Anirach Mingkhwan
```

# INPUT NUMBERS

ฟังก์ชัน `input()` จะ ส่งค่ากลับเป็นสตริง (string) เสมอ ไม่ว่าผู้ใช้จะป้อนข้อมูลเป็นตัวอักษรหรือตัวเลขก็ตาม ตัวอย่างเช่น ถ้าคุณเรียกใช้ `input()` และผู้ใช้พิมพ์ตัวเลข 87 และกด Enter ค่าที่ฟังก์ชัน `input()` ส่งกลับจะเป็น สตริง '87' ไม่ใช่ ตัวเลขจำนวนเต็ม 87

สิ่งนี้อาจก่อให้เกิดปัญหาเมื่อคุณต้องการนำค่าที่ผู้ใช้ป้อนมาใช้ในการ คำนวณทางคณิตศาสตร์ เพราะการดำเนินการทางคณิตศาสตร์ ไม่สามารถใช้กับค่าประเภทสตริงได้

ดังนั้น หากต้องการใช้ค่าที่ได้จาก `input()` ในการคำนวณ จำเป็นต้อง แปลงค่าจากสตริง ให้เป็นตัวเลข ด้วยฟังก์ชัน เช่น:

- `int()` สำหรับแปลงเป็นจำนวนเต็ม (integer)
- `float()` สำหรับแปลงเป็นจำนวนจริง (floating-point number)

```
number = input("กรอกตัวเลข: ")
result = int(number) + 10
print(result)
```

## INPUT

### Method 1

```
string_value = input('How many hours did you work? ')
hours = int(string_value)
```

### Method 2

```
hours = int(input('How many hours did you work? '))
```

# INPUT

```
1 # Get the user's name, age, and income.  
name = input('What is your name? ')  
age = int(input('What is your age? '))  
income = float(input('What is your income? '))  
  
# Display the data.  
print('Here is the data you entered: ')  
print('Name:', name)  
print('Age:', age)  
print('Income: ', format(income, '12,.2f'))
```

```
What is your name? Anirach Mingkhwan  
What is your age? 49  
What is your income? 100000  
Here is the data you entered:  
Name: Anirach Mingkhwan  
Age: 49  
Income: 100,000.00
```

# PRACTICAL EXAMPLES



## EXAMPLE I

---

**Algorithm 2:** Algorithm to Convert User Input and Display Information

---

**Input:** User's age and height

**Output:** Displays user's age and height

```
1 begin
2     Print "Enter your age:";
3     Read age;
4     age = int(age);
5     Print "Enter your height:";
6     Read height;
7     height = float(height);
8     Print "You are " + str(age) + " years old and " + str(height) +
9         " centimetre tall."
9 end
```

---

## EXAMPLE I

```
1 # Getting user input and converting to int
2 age = input("Enter your age: ")
3 age = int(age)
4
5 # Getting user input and converting to float
6 height = input("Enter your height: ")
7 height = float(height)
8
9 print("You are " + str(age) + " years old and " + str(height) + " feet tall
. ")
```

Listing 2.16: ตัวอย่างการแปลงข้อมูลจากผู้ใช้

## EXAMPLE II

---

### Algorithm 3: อัลกอริทึมสำหรับคำนวณ BMI

---

**Input:** น้ำหนักและส่วนสูงของผู้ใช้

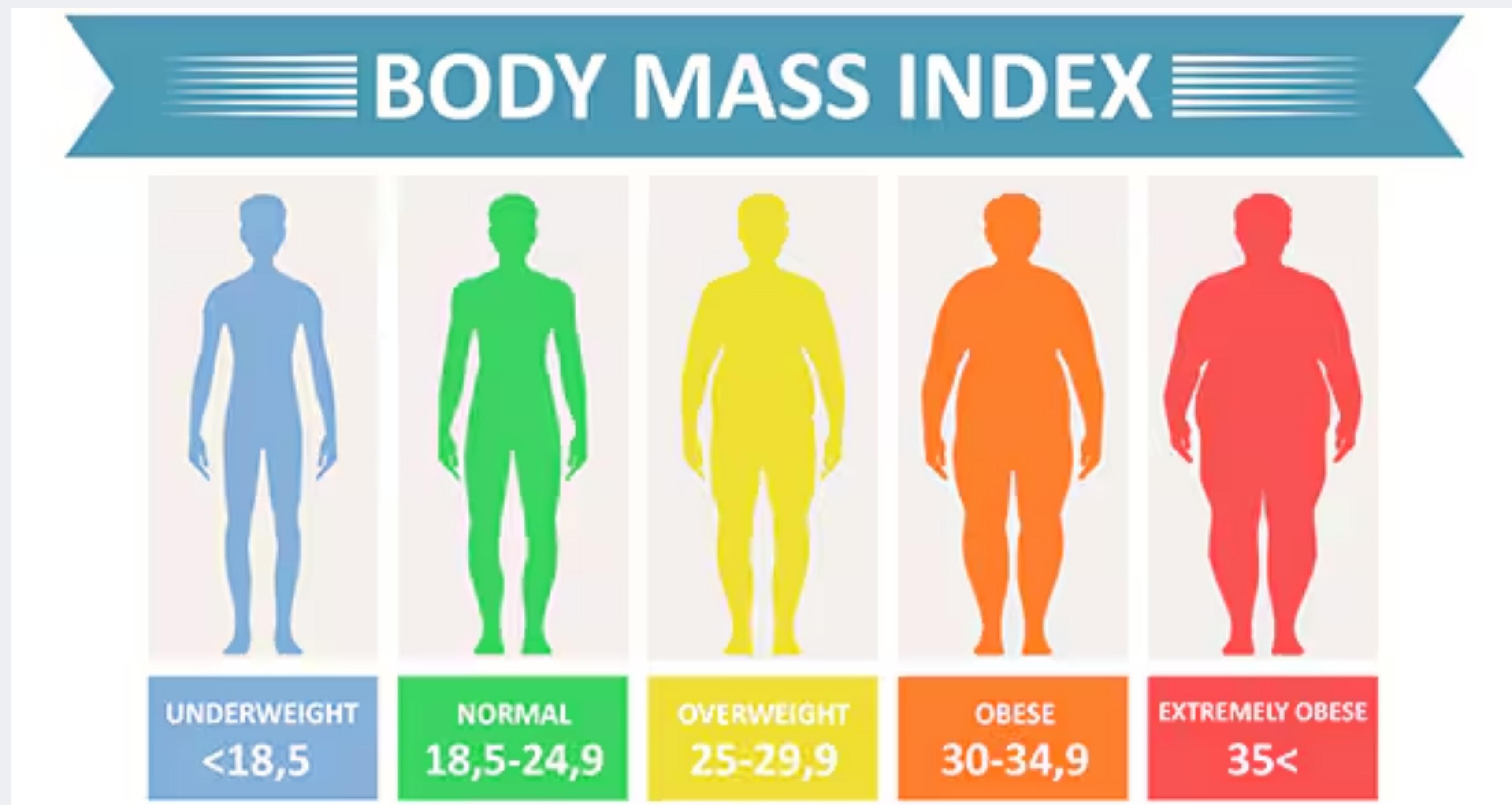
**Output:** ค่าดัชนีมวลกาย

```
1 begin
2   Print "Enter your weight in kilograms:";
3   Read weight;
4   Print "Enter your height in meters:";
5   Read height;
6   bmi = weight / (height * height);
7   Print "Your BMI is", bmi;
```

---

$$\text{BMI} = \frac{\text{weight in kg}}{\text{height}^2 \text{ in m}}$$

## EXAMPLE II



## EXAMPLE

---

### Algorithm 4: อัลกอริทึมแปลงอุณหภูมิจาก Celsius เป็น Fahrenheit

---

**Input:** อุณหภูมิในหน่วยเซลเซียส

**Output:** อุณหภูมิในหน่วยฟาร์นไฮต์

```
1 begin
2     Print "Enter temperature in Celsius:";
3     Read celsius;
4     fahrenheit = (celsius * 9/5) + 32;
5     Print "Temperature in Fahrenheit is", fahrenheit;
```

---

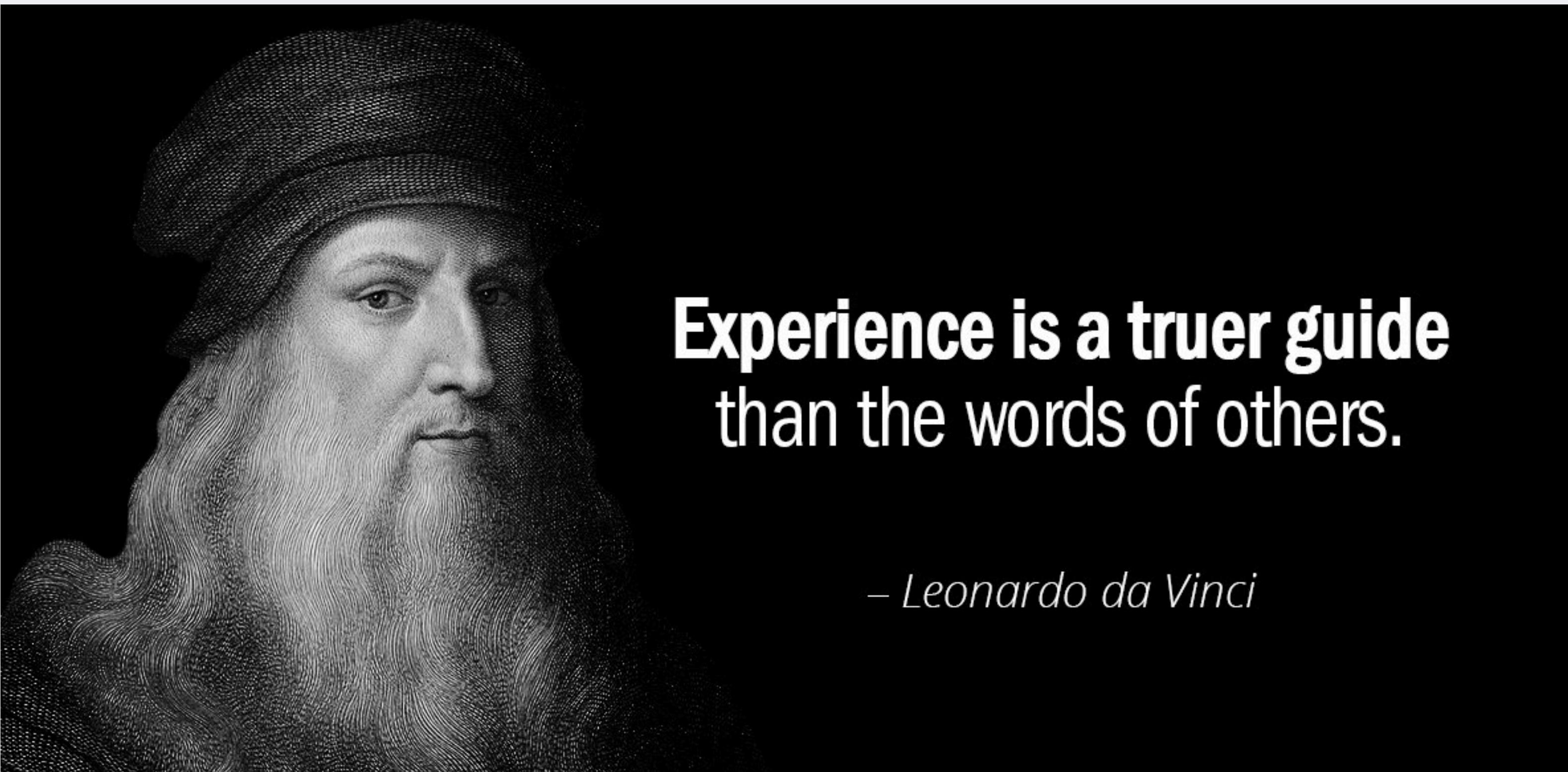


# THANKS

---

## WORD OF THE WISE

---



**Experience is a truer guide  
than the words of others.**

*– Leonardo da Vinci*