Computational Science

# Chapter 7: Introduction to Python Programming

Act 2: Conditional Statement & Iteration Statement

**Pasawut Cheerapakorn**
Suan Sunandha Rajabhat University

**DTI1306 Computational Science**
Department of Digital Technology for Education
Faculty of Education, Suan Sunandha Rajabhat University

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

# Course Description:

วิเคราะห์ เทคนิค วิธีการขั้นตอนการแก้ปัญหา ทักษะการคิดเชิงคำนวณ เชิงนามธรรม ฝึกทักษะในการแก้ปัญหาโดยใช้ขั้นตอนการแก้ปัญหา การย่อยปัญหา การแสดงขั้นตอน การแก้ปัญหา โดยการเขียน บอกเล่า วาดภาพ หรือใช้สัญลักษณ์ ออกแบบและเขียนโปรแกรมโดยใช้ซอฟต์แวร์หรืออุปกรณ์ เทคโนโลยีเบื้องต้น เพื่อไปประยุกต์ใช้ในการแก้ปัญหาในชีวิตประจำวัน ในการตัดสินใจได้อย่างมีประสิทธิภาพและตระหนักถึงการใช้งานสารสนเทศอย่างปลอดภัย พัฒนาโครงงานทางเทคโนโลยีดิจิทัล เพื่อการศึกษาที่มีการบูรณาการกับสาขาอื่น ๆ อย่างสร้างสรรค์และเชื่อมโยงกับชีวิตจริง

The study analyzed how the process solutions, abstract thinking skills, computational skills to solve problems by using the steps to solve the problem of small steps to solve the problem by writing a story or painting the symbol, designers and programmers using software or technology introduction, to use the solution on a daily basis, decisions efficiently and realize the information securely, technological development project.

- System Theory
- Computational Thinking
- Decomposition
- Abstraction
- Pattern Recognition
- Algorithm Design
- Design Thinking
- Flowchart Design Standard
- Flowgorithm
- Computer Programming

## Course Outline:

- Chapter 1 – Fundamental of Computational Science
- Chapter 2 – Digital Technology
- Chapter 3 – Digital Literacy
- Chapter 4 – Algorithm Design and Analysis
- Chapter 5 – Block-Based Programming
- Chapter 6 – Microbit for Learning
- **Chapter 7 – Introduction to Computer Programming**
- Chapter 8 – Project Design

# 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 พื้นฐานวิทยาการคำนวณ (Fundamental of Computational Science) |
| 3 | บทที่ 2 พื้นฐานเทคโนโลยีดิจิทัล (Digital Technology) |
| 4 | บทที่ 3 พื้นฐานการรู้เท่าทันดิจิทัล (Digital Literacy) |
| 5 | บทที่ 4 พื้นฐานการวิเคราะห์และออกแบบอัลกอริทึม (Algorithm Design and Analysis) |
| 6 | บทที่ 4 พื้นฐานการวิเคราะห์และออกแบบอัลกอริทึม (Algorithm Design and Analysis) [ต่อ] |
| 7 | บทที่ 5 การโปรแกรมแบบ Block-Based ด้วย Scratch (Block-Based Programming) |
| 8 | บทที่ 5 การโปรแกรมแบบ Block-Based ด้วย Scratch (Block-Based Programming) [ต่อ] |

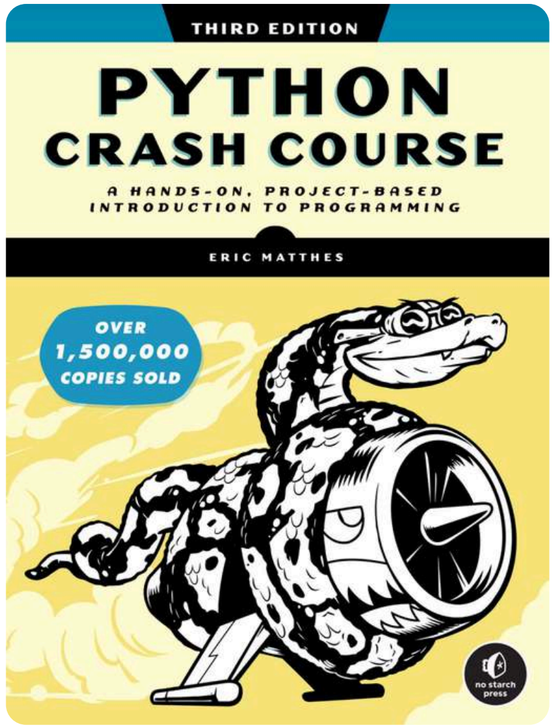
# Measurement and Evaluation:

| ครั้งที่ / สัปดาห์ | บทเรียน / หัวข้อ |
|---|---|
| 9 | สอบกลางภาค (Midterm Examination) |
| 10 | บทที่ 6 การโปรแกรมไมโครคอนโทรลเลอร์เบื้องต้นด้วย Microbit (Microbit for Learning) |
| 11 | บทที่ 6 การโปรแกรมไมโครคอนโทรลเลอร์เบื้องต้นด้วย Microbit (Microbit for Learning) [ต่อ] |
| 12 | **บทที่ 7 การเขียนโปรแกรมคอมพิวเตอร์เบื้องต้น (Introduction to Computer Programming)** |
| 13 | บทที่ 8 การออกแบบโครงงานทางเทคโนโลยีเพื่อการศึกษา (Project Design) |
| 14 | สอบปลายภาค (Final Examination) |
| 15 | นำเสนอและส่งโครงงาน (Project Pitching and Presentation) |
| 16 | |

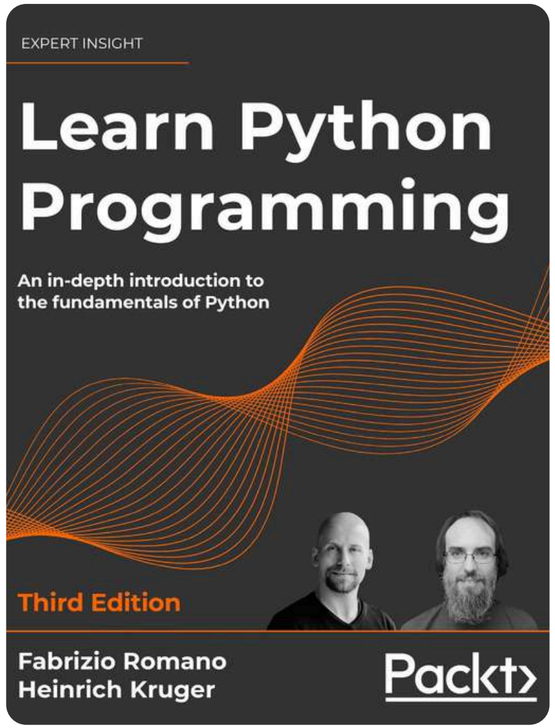
# Chapter 7: Introduction to Python Programming

1. Comparison Operator
2. Conditional Statement
3. if
4. if else
5. if elif
6. if elif else
7. Random number
8. unicode
9. Iteration Statement
10. while loop
11. range
12. for loop
13. break

# 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.]

**Website**

- https://www.w3schools.com/python/
- https://realpython.com/
- https://docs.python.org/3/tutorial/index.html
- https://pythontutor.com/

# Pre Test

**Question:**

# 1. ต้องการแสดงผลคำว่า Hello, World ต้องเติมคำสั่งใด

**A** Print(“Hello, World”)

**B** Print(Hello, World)

**C** print(“Hello, World”)

**D** print(Hello, World)

**E** ถูกทุกข้อ

**F** ไม่มีข้อถูก

**Question:**

## 2. ข้อใดไม่ใช่ Reserved Word

**A** True

**B** None

**C** break

**D** class

**E** Control

**Question:**

# 3. ข้อใดให้ผลลัพธ์ Date:11/DEC/2023

**A**

```
print('Date:')
print('11', 'DEC','2023', sep='/')
```

**B**

```
print('Date:', end='/n')
print('11'/ 'DEC'/'2023', sep='/')
```

**C**

```
print('Date:')
print(‘11/', ‘DEC/','2023')
```

**D**

```
print('Date:', end='')
print('11', 'DEC', '2023', sep='/')
```

**Question:**

# 4. ข้อใด คือ การตั้งชื่อตัวแปรแบบ Pascal Case

**A** My_Product_Price

**B** MyProductPrice

**C** myProductPrice

**D** my_product_price

**E** MYPRODUCTPRICE

**Question:**

## 5. จาก Code แสดงผลตามข้อใด

```
1.  data1 = ”50”
2.  data1 = float(data1)
3.  print(data1, 'is', type(data1))
```

**A** 50.0 is <class 'float'>

**B** 50 is <class 'float'>

**C** “50” is <class ‘str'>

**D** “50.0” is <class 'float'>

**Question:**

## 6. จาก Code ข้อใดให้ค่า Boolean เป็น True

```
1.  student1 = 50
2.  student2 = 100
```

**A** print(student1>student2)

**B** print(student1=student2)

**C** print(student1<student2)

**D** print(student1!student2)

**Question:**

## 7. จาก Code ประเภทข้อมูลที่รับเข้ามาเป็นประเภทใด

```
1.  year = input(“Birth Year:”)
Birth Year: 1983
```

**A** int

**C** string

**B** float

**D** real

**Question:**

## 8. ข้อใดไม่ใช่ Keywords ใน Conditional Statement ในภาษา Python

**A** if

**B** if elif

**C** else

**D** if else

**E** if elif else

**F** if elseif else

**Question:**

## 9. จากคำสั่ง ไม่สามารถ แสดงผลข้อใด

```
import random
num = random.randrange(5,10)
print(num)
```

**A** 5

**B** 7

**C** 10

**D** 9

**E** 8

**F** 6

**Question:**

## 10. จากคำสั่ง ไม่สามารถ แสดงผลข้อใด

```
1. count = 1
2. while count<=5:
3.    print(count)
4.    count+=1
```

**A** 1

**B** 2

**C** 3

**D** 4

**E** 5

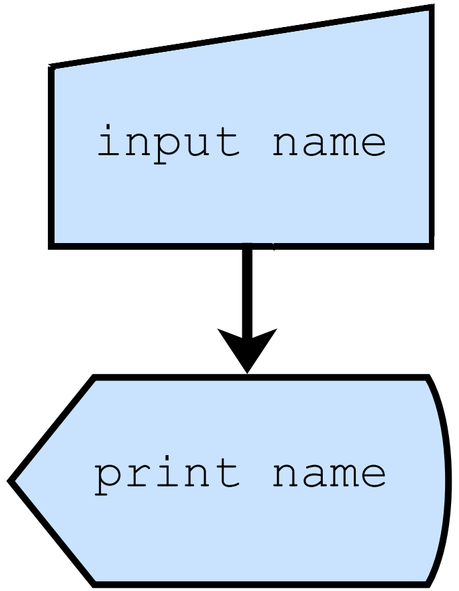
**F** 6

# โครงสร้างการควบคุมโปรแกรม (Program Control Structures)

## 1. Sequential Statement (คำสั่งแบบลำดับ)

- คำสั่งแบบลำดับเป็นรูปแบบพื้นฐานที่สุดของโครงสร้างการควบคุมโปรแกรม
- โปรแกรมจะทำงานเรียงตามลำดับของคำสั่งที่เขียนไว้ในโปรแกรม เรียกว่า Top-Down
- ไม่มีการตรวจสอบเงื่อนไขหรือการทำซ้ำ
- ทุกคำสั่งจะถูกทำงานหนึ่งครั้งตามลำดับที่ปรากฏ

```
# Sequential Statement
# Code 1
name = input("Please enter your name:")
print("Hello, "name)
```



# โครงสร้างการควบคุมโปรแกรม (Program Control Structures)

## 2. Conditional Statement (คำสั่งเงื่อนไข)

คำสั่งเงื่อนไขช่วยให้โปรแกรมสามารถเลือกทำงานทางเลือกต่าง ๆ ตามเงื่อนไขที่กำหนด
โครงสร้างนี้ประกอบด้วยคำสั่ง **if**, **elif**, และ **else** ในภาษา Python

ตัวอย่าง เช่น ถ้าเงื่อนไขเป็นจริง (True) บล็อกของคำสั่งใน if จะถูกทำงาน;
ถ้าเงื่อนไขเป็นเท็จ (False), โปรแกรมอาจเลือกทำงานบล็อกใน elif หรือ else

# Conditional Statement (คำสั่งเงื่อนไข) - if

```
1.# Code 2
2.sunset = True
3.if sunset:
4.    print("Good Evening")
```

**Good Evening**

```
1.# Code 3
2.sunset = True
3.if sunset:
4.    print("Good Evening")
5.else:
6.    print("Good Morning")
```

# Conditional Statement (คำสั่งเงื่อนไข) - if

```
# Code 4
height = 178
weight = 75
if(weight>60 and weight<=80):
    print("Weight: Perfect")
```

**A** Weight: Perfect

**B** Nil

# Conditional Statement (คำสั่งเงื่อนไข) - if

Set a condition to print **'You are short.'** if the height in the variable my_height is less than 150

```
# Code 5
my_height = 142
if ____________ :
    print('You are short.')
```

# Conditional Statement (คำสั่งเงื่อนไข) - if

Set a condition to print **'You are wealthy and tall.'** if the salary in my_salary is greater than 35000 and my_height is greater than 170.

```
# Code 6
my_salary = _______
my_height = _______
if ____________ :
    print('You are wealthy and tall.')
```

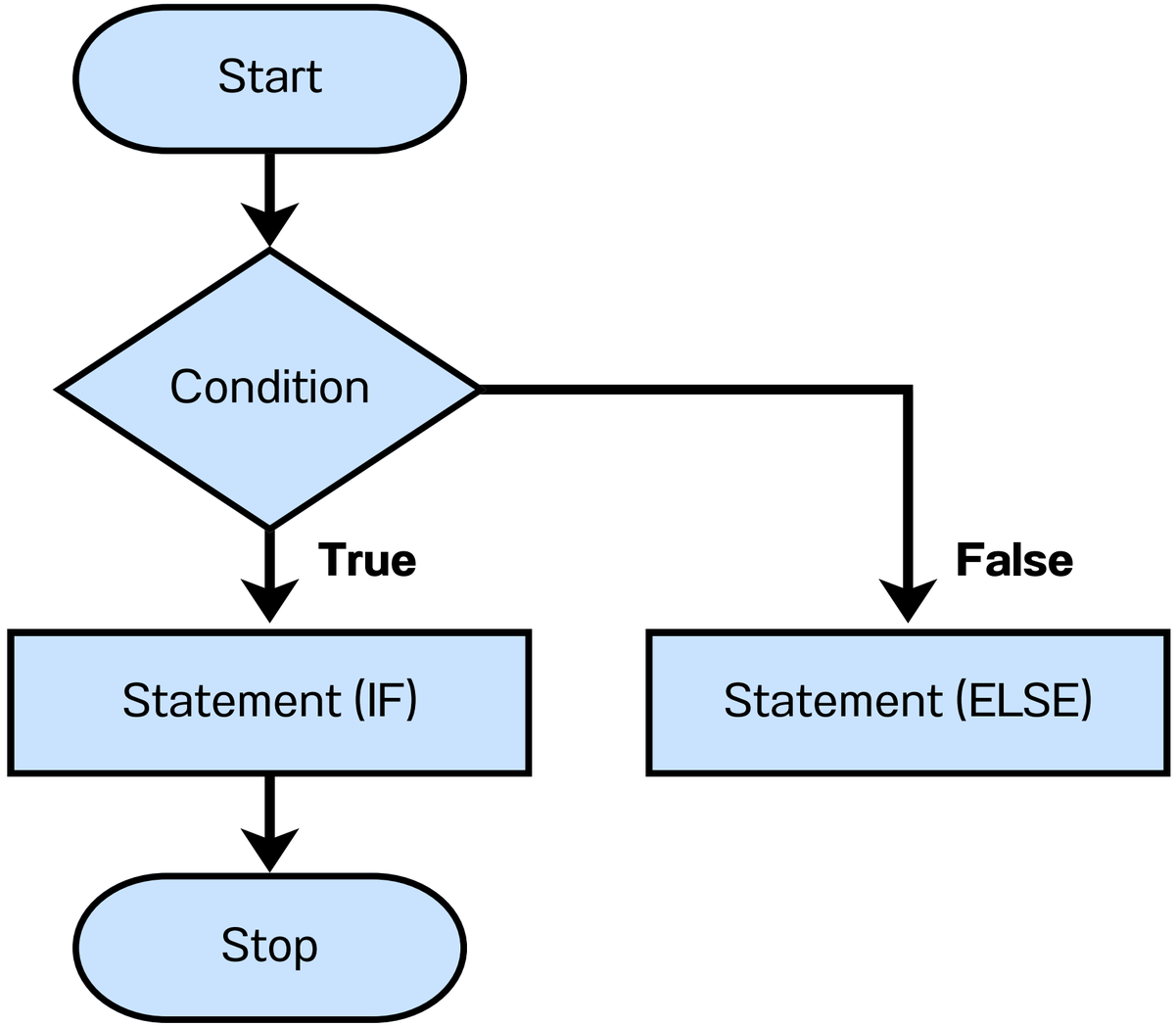
# Conditional Statement (คำสั่งเงื่อนไข) - if

```
# Code 7
object1="strawberry"
object2="pencil"
if(object1=="strawberry" or object2=="watermelon"):
    print("There is fruit in the basket")
```

**A** There is fruit in the basket

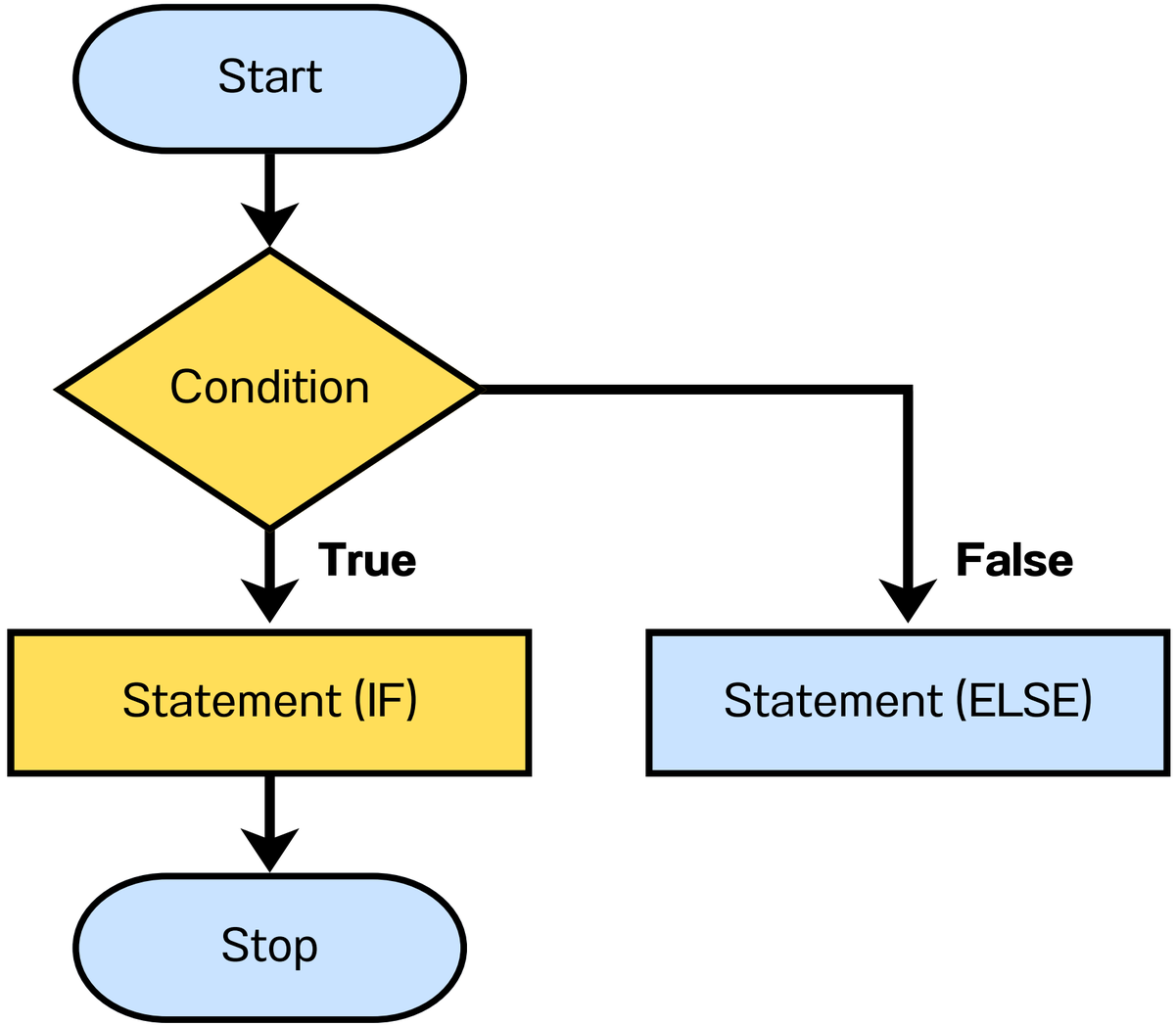
**B** Nil

# Conditional Statement (คำสั่งเงื่อนไข) – if



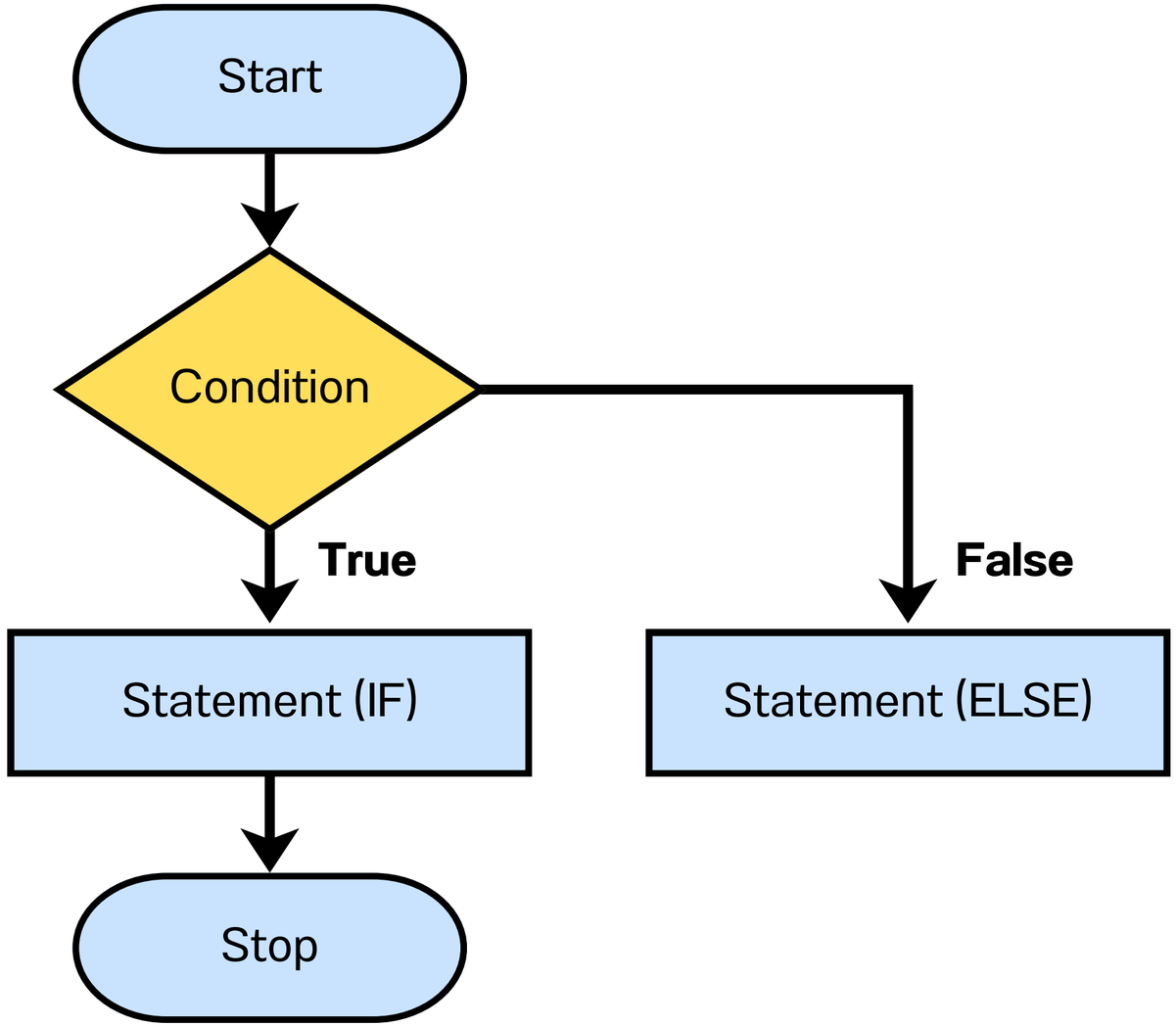

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if



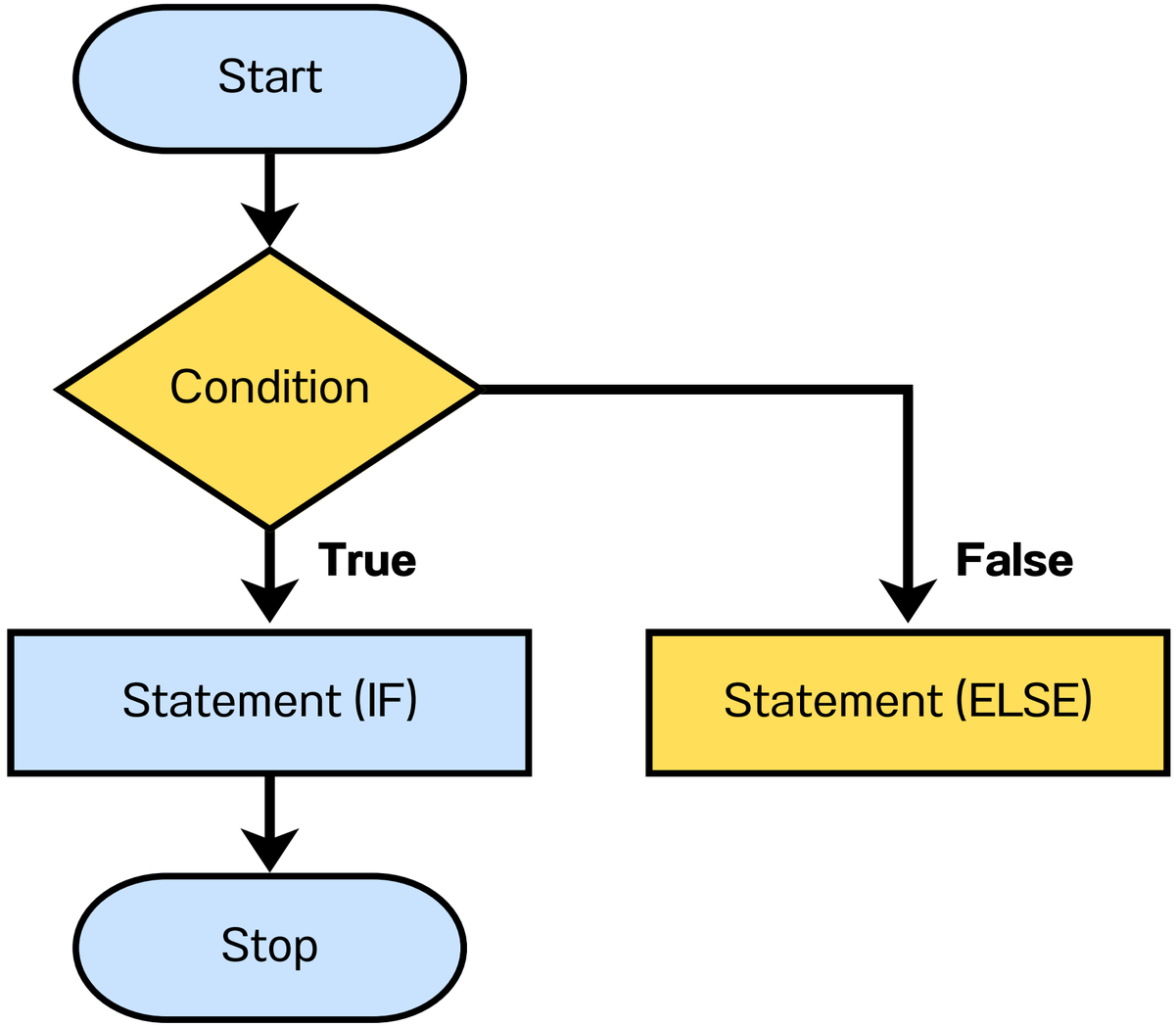

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if




```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if



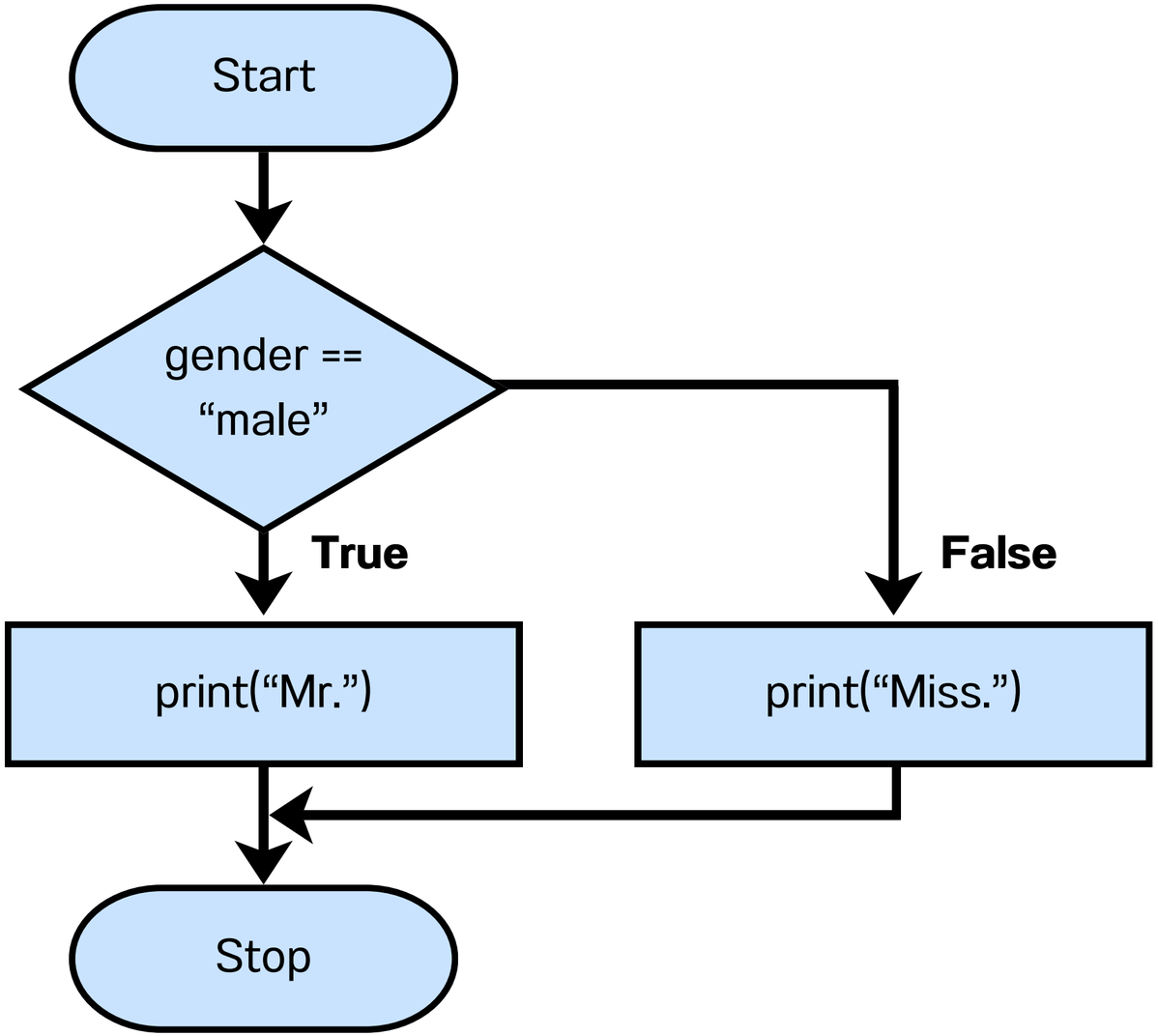

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

# ตัวดำเนินการเปรียบเทียบ (Comparison Operators)

**ตัวดำเนินการเปรียบเทียบ (Comparison Operators)** เป็นตัวดำเนินการที่ใช้ในการเปรียบเทียบค่าของตัวแปรหรือค่าที่มีอยู่ในนิพจน์ เพื่อตรวจสอบว่าสอดคล้องกับเงื่อนไขที่กำหนดหรือไม่ ผลลัพธ์ของการเปรียบเทียบจะเป็นค่าบูลีน (Boolean) ซึ่งอาจเป็น True (จริง) หรือ False (เท็จ) ตามผลการเปรียบเทียบ

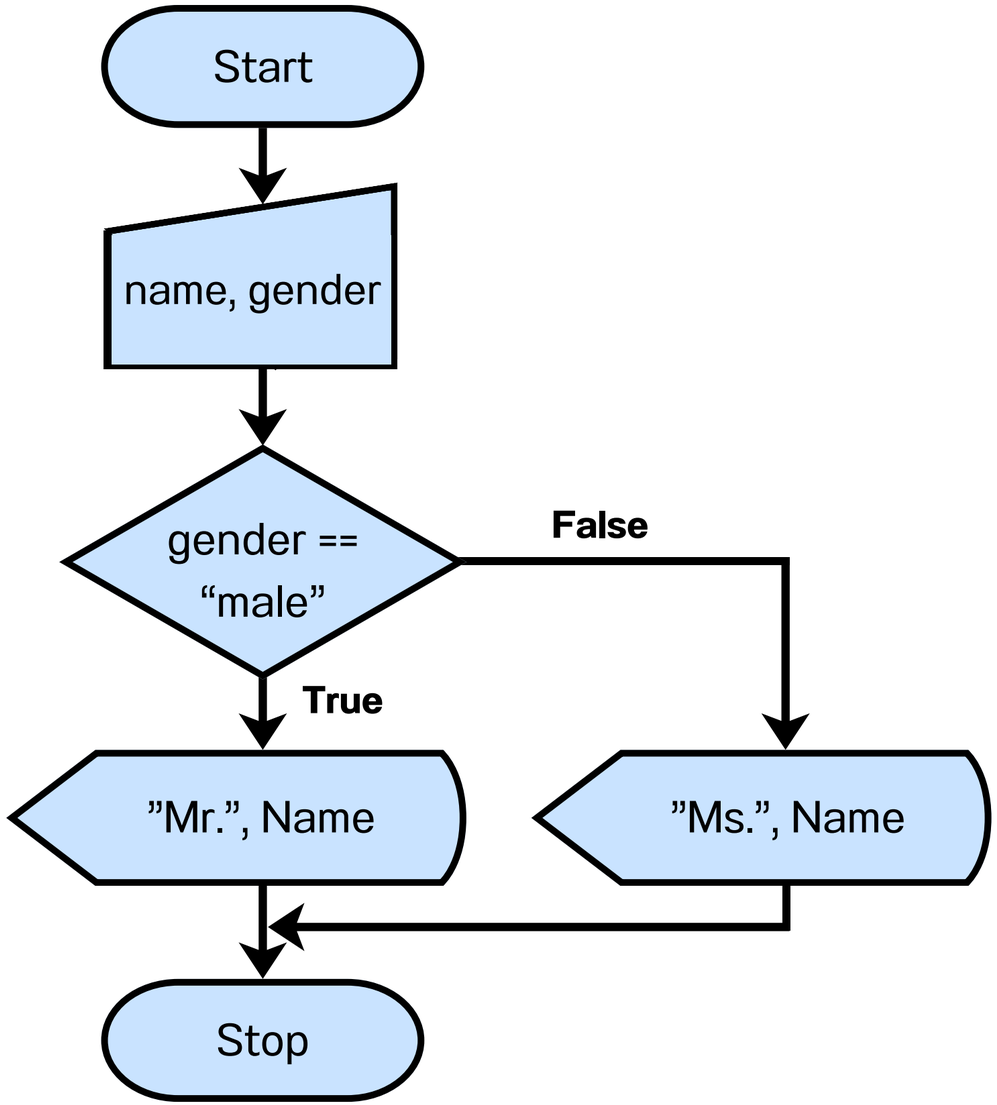
1. เท่ากับ (==): ตรวจสอบว่าค่าของสองค่าเท่ากันหรือไม่
   ตัวอย่าง: x == y จะเป็น True ถ้า x มีค่าเท่ากับ y
2. ไม่เท่ากับ (!=): ตรวจสอบว่าค่าของสองค่าไม่เท่ากัน
   ตัวอย่าง: x != y จะเป็น True ถ้า x ไม่เท่ากับ y
3. มากกว่า (>): ตรวจสอบว่าค่าด้านซ้ายมากกว่าค่าด้านขวา
   ตัวอย่าง: x > y จะเป็น True ถ้า x มากกว่า y
4. น้อยกว่า (<): ตรวจสอบว่าค่าด้านซ้ายน้อยกว่าค่าด้านขวา
   ตัวอย่าง: x < y จะเป็น True ถ้า x น้อยกว่า y
5. มากกว่าหรือเท่ากับ (>=): ตรวจสอบว่าค่าด้านซ้ายมากกว่าหรือเท่ากับค่าด้านขวา
   ตัวอย่าง: x >= y จะเป็น True ถ้า x มากกว่าหรือเท่ากับ y
6. น้อยกว่าหรือเท่ากับ (<=): ตรวจสอบว่าค่าด้านซ้ายน้อยกว่าหรือเท่ากับค่าด้านขวา
   ตัวอย่าง: x <= y จะเป็น True ถ้า x น้อยกว่าหรือเท่ากับ y

# Conditional Statement (คำสั่งเงื่อนไข) – if




```
if( gender == “male” ):
        print(“Mr.”)
else:
        print(“Miss.”)
```

# Practice 1: ให้รับข้อมูลเพศ และเติมคำนำหน้าชื่อตาม Flowchart



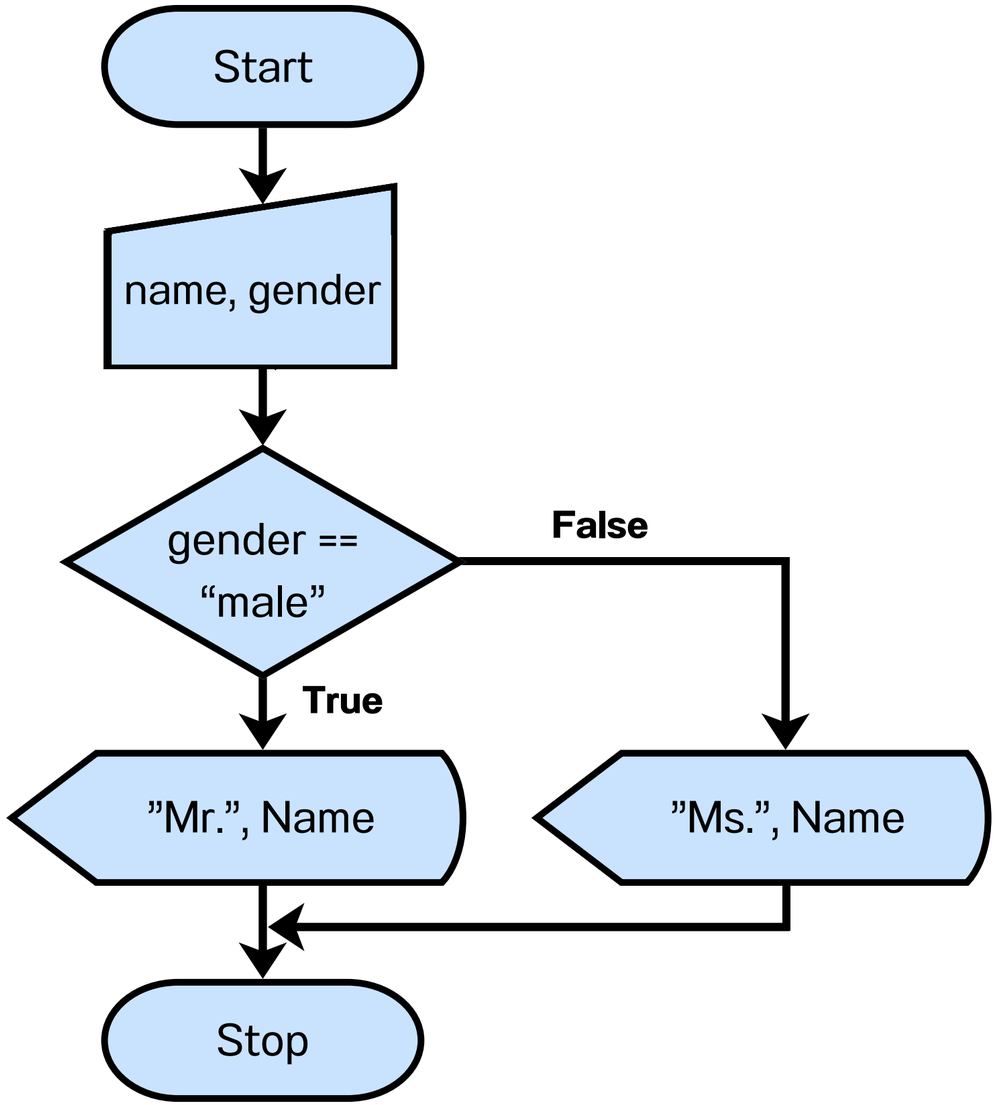

Input Name: ___Pasawut___

Input Gender:__male___

↓

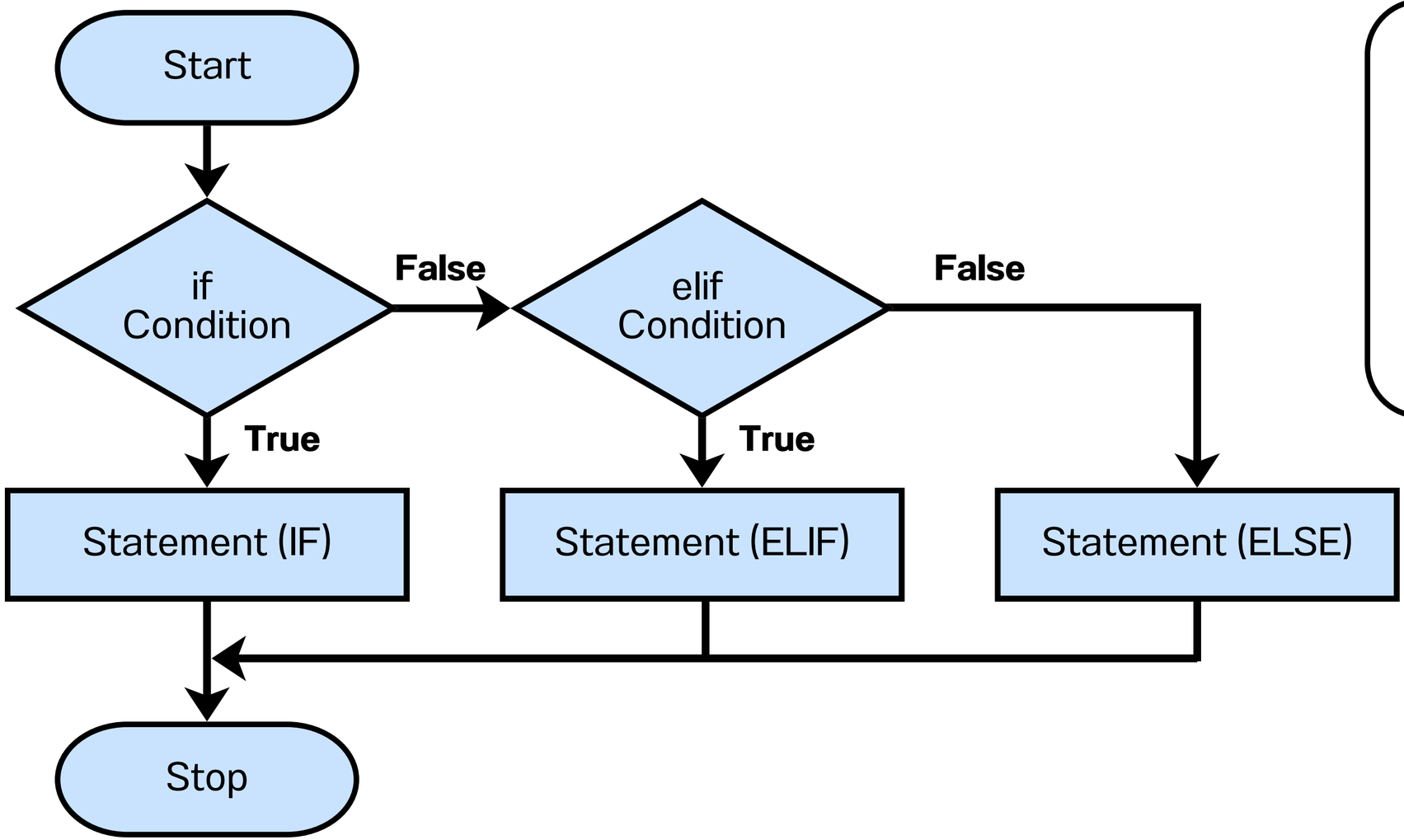
Mr.Pasawut

# Practice 1: ให้รับข้อมูลเพศ และเติมคำนำหน้าชื่อตาม Flowchart



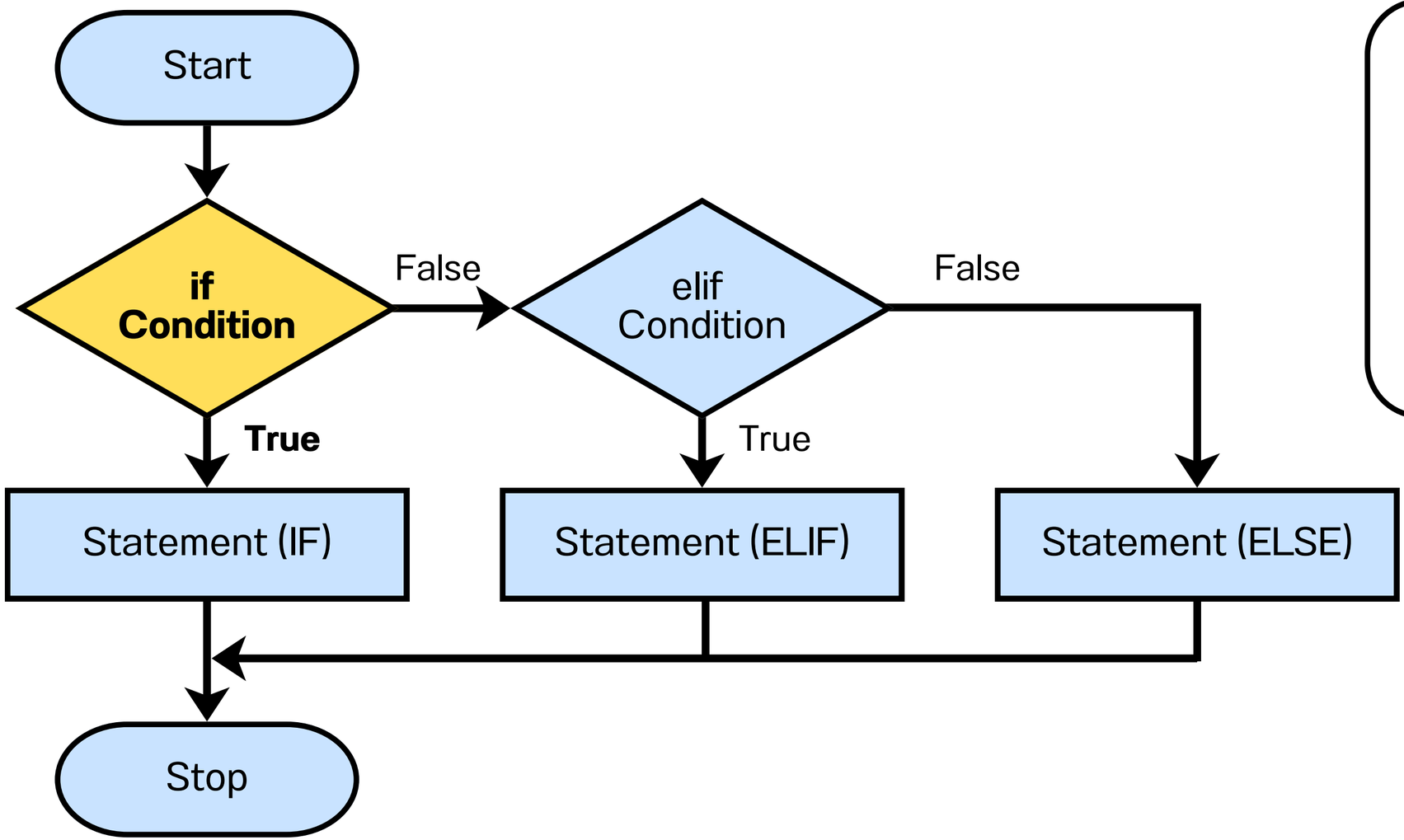

```
# Practice 1
name = input("Name: ")
gender = input("Gender: ")
if(gender == "male"):
      print("Mr. "+ name)
else:
      print("Ms. "+ name)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if elif else



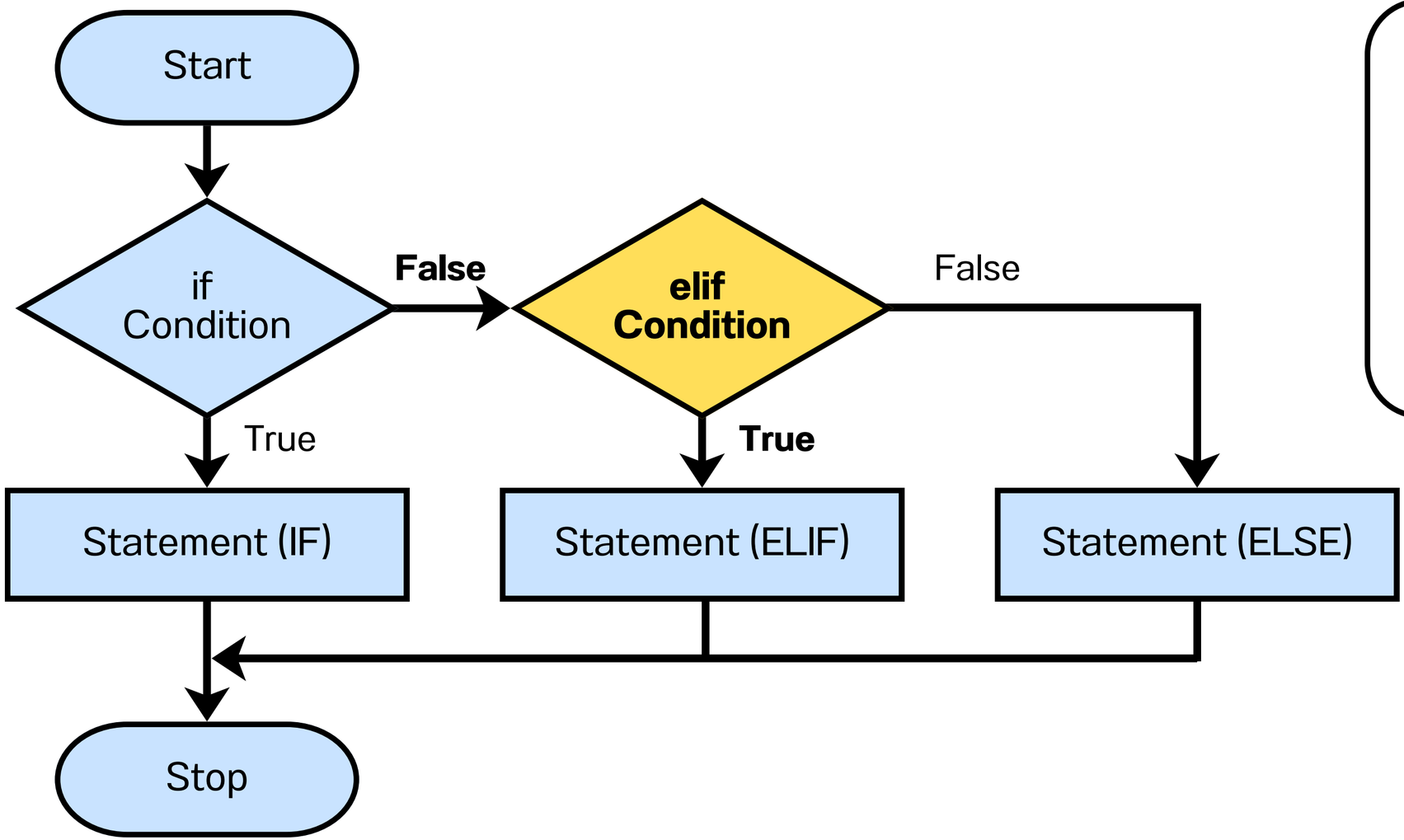

```
if( Condition ):
        Statement (IF)
elif( Condition ):
        Statement (ELIF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if elif else



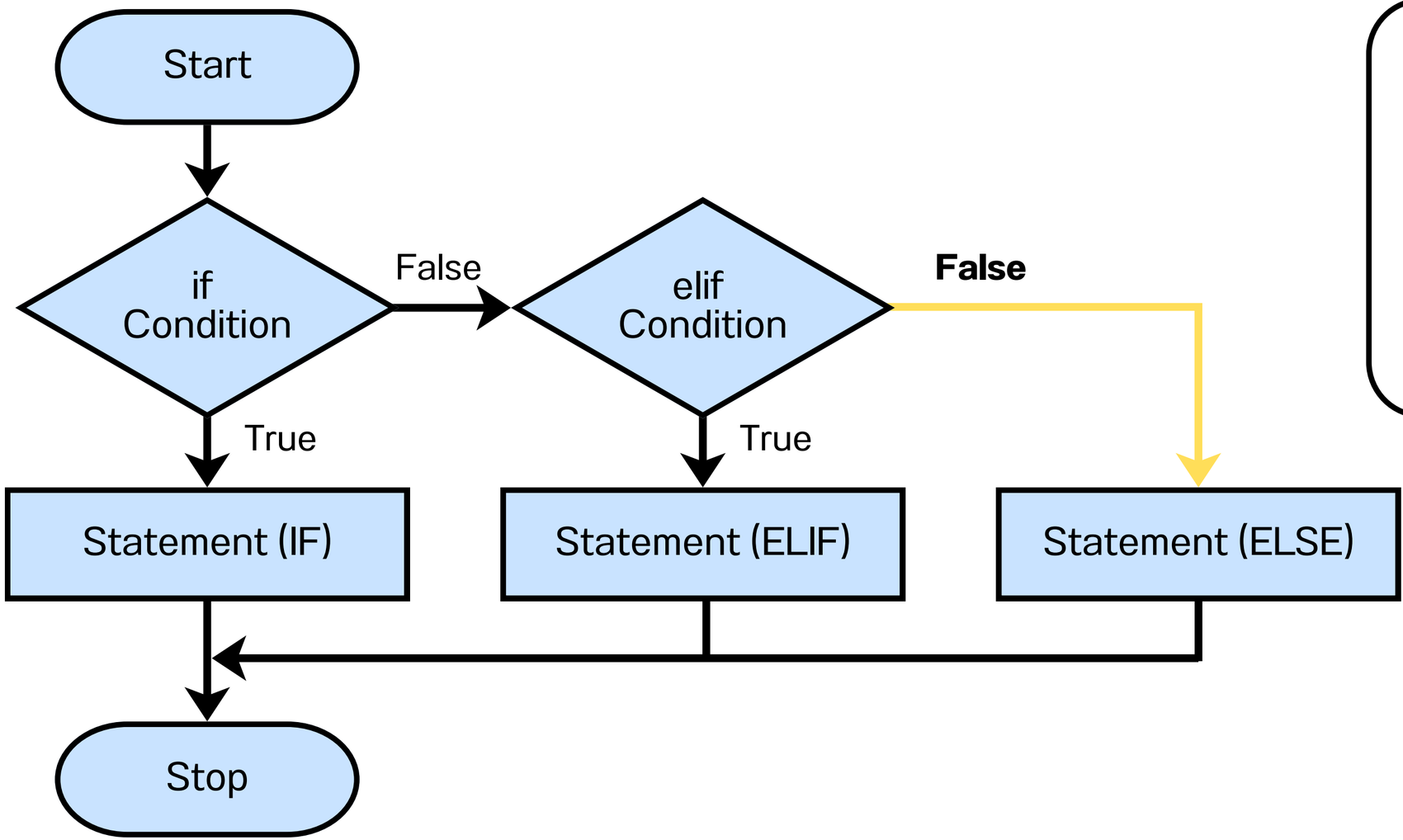

```
if( Condition ):
        Statement (IF)
elif( Condition ):
        Statement (ELIF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if elif else




```
if( Condition ):
        Statement (IF)
elif( Condition ):
        Statement (ELIF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if elif else




```
if( Condition ):
        Statement (IF)
elif( Condition ):
        Statement (ELIF)
else:
        Statement (ELSE)
```

# Conditional Statement (คำสั่งเงื่อนไข) – if elif else

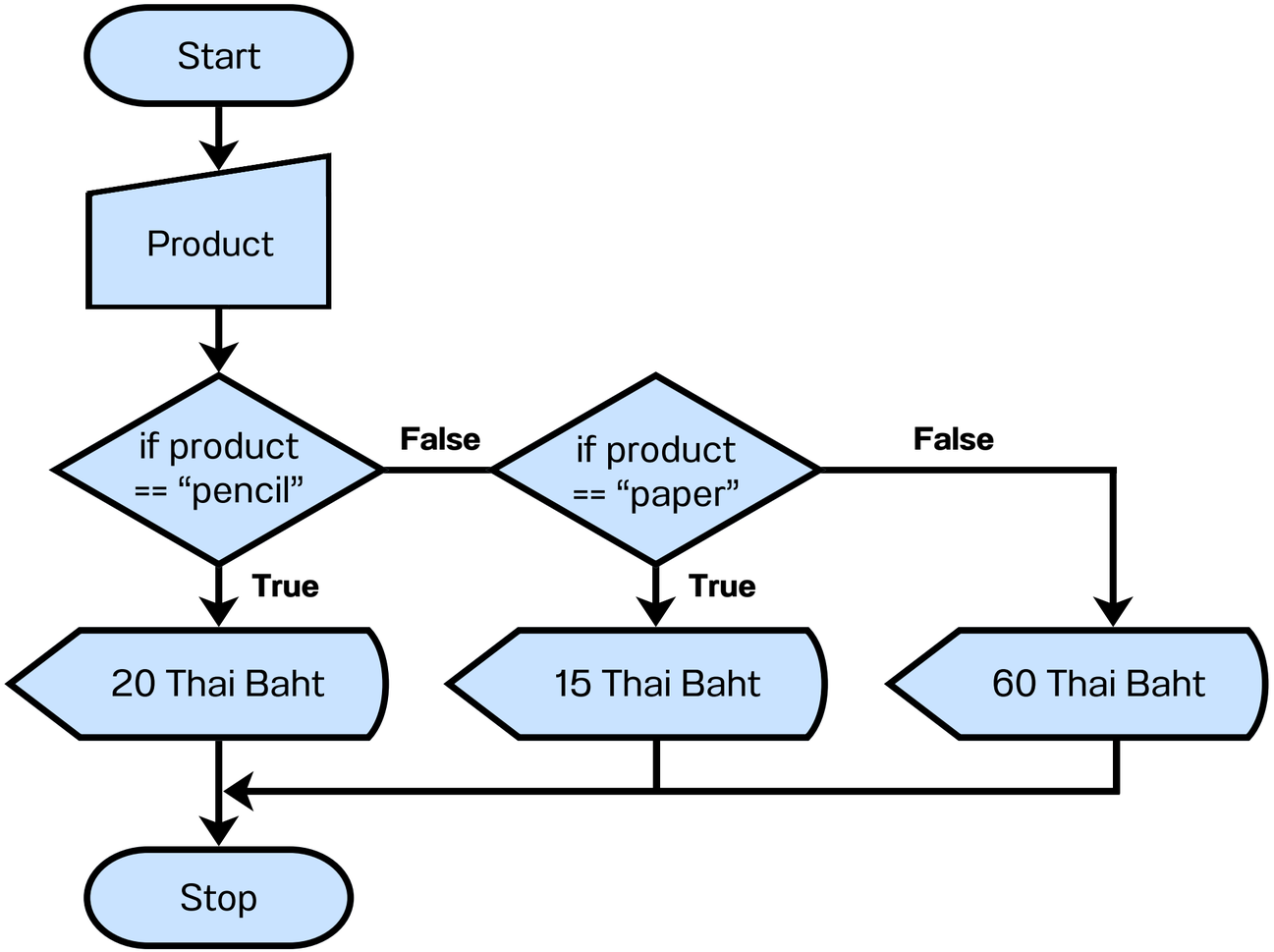
```
# Condition Statement
# Code 8
product = input("Please enter product:")
if product == "pencil":
   print(product, "20 Thai Baht")
elif product == "paper":
   print(product, "15 Thai Baht")
else:
   print(product, "60 Thai Baht")
```

**คำอธิบาย Flowchart**

1. เริ่ม
2. รับข้อมูล (input) ชื่อสินค้า โดยใช้คำสั่ง
   product = input("Please enter product:")
3. ตรวจสอบเงื่อนไขแรก:
   ถ้า product == "pencil" เป็นจริง ให้พิมพ์ข้อความ product, "20 Thai Baht”
4. ตรวจสอบเงื่อนไขที่สอง (ถ้าเงื่อนไขแรกไม่เป็นจริง):
   ถ้า product == "paper" เป็นจริง ให้พิมพ์ข้อความ product, "15 Thai Baht”
5. ถ้าเงื่อนไขแรกและเงื่อนไขที่สองไม่เป็นจริง ให้ไปที่ else:
   พิมพ์ข้อความ product, "60 Thai Baht”
6. จบการทำงาน

# Conditional Statement (คำสั่งเงื่อนไข) - if elif else

```
# Condition Statement
# Code 8
product = input("Please enter product:")
if product == "pencil":
   print(product, "20 Thai Baht")
elif product == "paper":
   print(product, "15 Thai Baht")
else:
   print(product, "60 Thai Baht")
```



## modulus (หรือเครื่องหมาย %)

การใช้ **modulus** (หรือเครื่องหมาย %) เป็นการหารเลขและคืนเศษของการหาร โดยการใช้ modulus คุณจะได้ผลลัพธ์เป็นเศษที่เหลือจากการหารของเลข 2 จำนวน โดยทั่วไปการใช้ modulus จะมีรูปแบบดังนี้

**a % b**

a คือตัวเลขที่จะนำไปหาร
b คือตัวเลขที่ใช้ในการหาร a
ผลลัพธ์ของ a % b คือเศษที่เหลือจากการหาร a ด้วย b.

## modulus (หรือเครื่องหมาย %)

**10 % 3** จะได้ผลลัพธ์เป็น 1 เพราะ 10 หาร 3 แลว้ เหลือเศษ 1

**20 % 4** จะได้ผลลัพธ์เป็น 0 เพราะ 20 หาร 4 ลงตัว ไม่มีเศษ

**15 % 7** จะได้ผลลัพธ์เป็น 1 เพราะ 15 หาร 7 แล้ว เหลือเศษ 1

**4 % 2 ได้?** **6 % 2 ได้?** **8 % 2 ได้?** **100 % 2 ได้?**

# modulus (หรือเครื่องหมาย %)

```
# Code 9
number = 77
if number % 2 == 0:
    print(number,"เป็นเลขคู่")
else:
    print(number,"เป็นเลขคี่")
```

**77 เป็นเลขคี่**

# modulus (หรือเครื่องหมาย %)

#ตรวจสอบเลขเหล่านี้เป็นเลขคู่ หรือ คี่

```
num1 = 204
num2 = -8
num3 = 22.22
```

# modulus (หรือเครื่องหมาย %)

#ตรวจสอบเลขเหล่านี้หาร 3 ลงตัวไหม

```
num1 = 27
num2 = 21
num3 = 972
```

# modulus (หรือเครื่องหมาย %)

#ตรวจสอบเลขเหล่านี้หาร 5 ลงตัวไหม

```
num1 = 225
num2 = 1225
num3 = 100.105
```

# modulus (หรือเครื่องหมาย %)

การใช้ modulus มีประโยชน์ในหลายสถานการณ์ เช่น

- ตรวจสอบว่าเลขคู่หรือเลขคี่: ในการตรวจสอบว่าเลขหนึ่งเป็นเลขคู่หรือเลขคี่ โดยสามารถใช้ **n % 2** โดยถ้าผลลัพธ์เป็น 0 แสดงว่าเป็นเลขคู่ และถ้าผลลัพธ์ไม่เท่ากับ 0 แสดงว่าเป็นเลขคี่
- การสร้างรอบ (loop): การใช้ modulus เป็นวิธีที่ดีในการสร้างรอบหรือลูปที่ทำงานซ้ำเมื่อมีจำนวนตัวเลขที่ต้องการประมวลผลเท่าๆ กัน เช่นในการแสดงตัวเลขคู่ทั้งหมดหรือเลขคี่ทั้งหมดจาก 1 ถึง n
- การใช้ modulus เป็นเครื่องมือที่มีประโยชน์ในการทำงานกับตัวเลขและคำนวณต่างๆ ในการเขยี นโปรแกรมและการแก้ปัญหาทางคณิตศาสตร์

# การสุ่มจำนวน (Random numbers)

- **การสุ่มจำนวน (random number)** คือ กระบวนการที่ใช้เลือกหรือสร้างตัวเลขที่มีค่าสุ่มจากช่วงหรือชุดข้อมูลที่กำหนด โดยความสุ่มนี้ทำให้ตัวเลขที่ได้มีค่าไม่แน่นอนและไม่สามารถทำนายล่วงหน้าได้ สิ่งที่สำคัญในการสุ่มคือความคล้ายกันในการเกิดตัวเลขที่มีค่าสุ่มอย่างเท่าเทียมกันซึ่งไม่มีลำดับหรือรูปแบบที่ซ้ำกัน
- การสุ่มจำนวนมักถูกนำมาใช้ในหลายสถานการณ์ เช่น การเล่นเกม, การทำการทดลองทางวิทยาศาสตร์, การสร้างข้อมูลทดสอบ (testing data), และอื่น ๆ อีกมากมาย
- ในโปรแกรมคอมพิวเตอร์การสุ่มจำนวนจะใช้ฟังก์ชันสุ่ม (random function) เพื่อสร้างตัวเลขที่มีค่าสุ่ม ในภาษา Python

```
1.  # Code 10
2.  import random

3.  # สุ่มจำนวนเต็มในช่วง 1 ถึง 10
4.  random_int = random.randint(1, 10)
5.  print(random_int)

6.  # สุ่มจำนวนเต็มในระยะ 1 ถึง 9
7.  random_range = random.randrange(1, 10)
8.  print(random_range)

9.  # สุ่มจำนวนทศนิยมในช่วง 0.0 ถึง 1.0
10. random_float = random.random()
11. print(random_float)
```

# Practice 2: สุ่มและบอกว่าค่านั้นเป็นเลขคู่ หรือ เลขคี่

1. ให้ทำการเขียนโปรแกรมเพื่อสุ่มเลข ระหว่าง 1 ถึง 100
2. นำเลขสุ่มที่ได้ ไปพิจารณาหาว่า ค่านั้นเป็นเลขคู่ หรือ เลขคี่
3. แจ้งออกทางหน้าจอว่าสุ่มได้เลขอะไร และเลขนั้นเป็นเลขคู่ (Even) หรือ เลขคี่ (Odd)

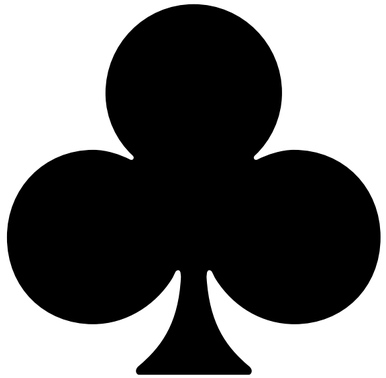
# Practice 2: สุ่มและบอกว่าค่านั้นเป็นเลขคู่ หรือ เลขคี่

1. ให้ทำการเขียนโปรแกรมเพื่อสุ่มเลขระหว่าง 1 ถึง 100
2. นำเลขสุ่มที่ได้ ไปพิจารณาหาว่าค่านั้นเป็น เลขคู่ หรือ เลขคี่
3. แจ้งออกทางหน้าจอว่าสุ่มได้เลขอะไร และเลขนั้นเป็นเลขคู่ (Even) หรือ เลขคี่ (Odd)

```
# Practice 2
1. import random
2. # สุ่มเลขในช่วง 1 ถึง 100
3. random_number = random.randint(1, 100)

4. # ตรวจสอบว่าเลขนี้เป็นเลขคู่หรือเลขคี่
5. if random_number % 2 == 0:
6.     result = "เลขคู่ (Even)"
7. else:
8.     result = "เลขคี่ (Odd)”

9. # แสดงผลลัพธ์
10.print("เลขที่สุ่มได้:", random_number)
11.print("ผลลัพธ์: ", result)
```

# print( ) and Unicode

```
print("\u2663")
```



```
1 print("\u2660")
2 print("\u2665")
3 print("\u2666")
4 print("\u2663")
```

♠
♥
♦
♣

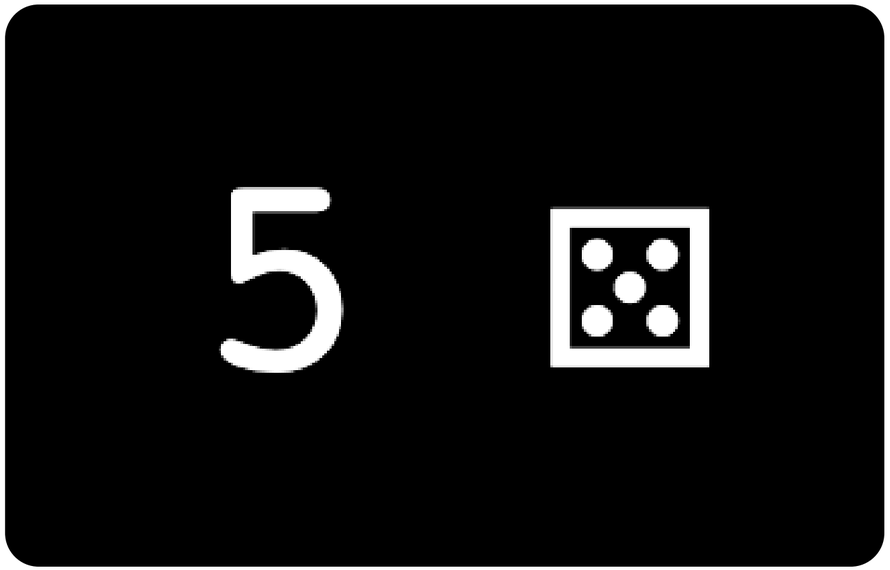
# print( ) and Unicode

```
# Code 11
print("\u2680")
print("\u2681")
print("\u2682")
print("\u2683")
print("\u2684")
print("\u2685")
```

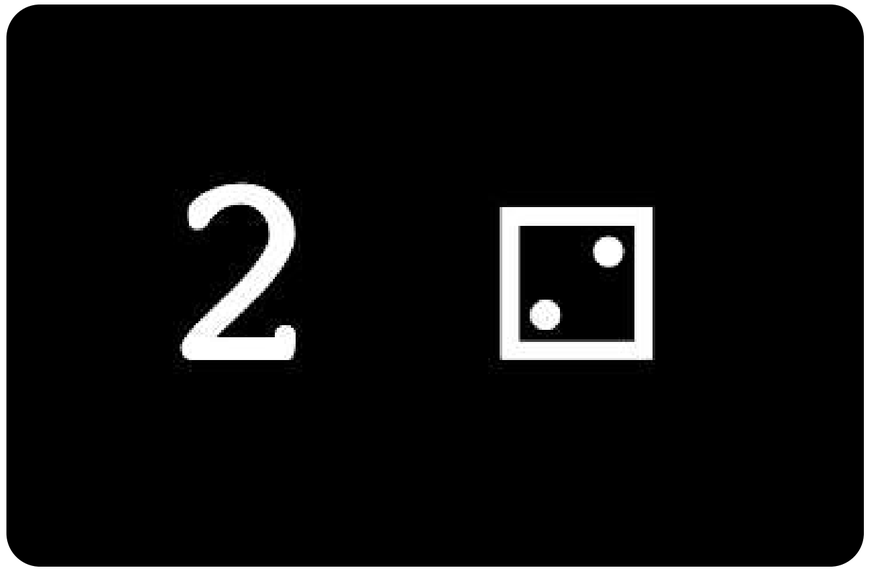
## Practice 3 : Randomize numbers and convert them to the sides of the dice

ให้เขียนโปรแกรมเพื่อสุ่มตัวเลขและแปลงให้เป็นด้านของลูกเต๋า

Randomize numbers



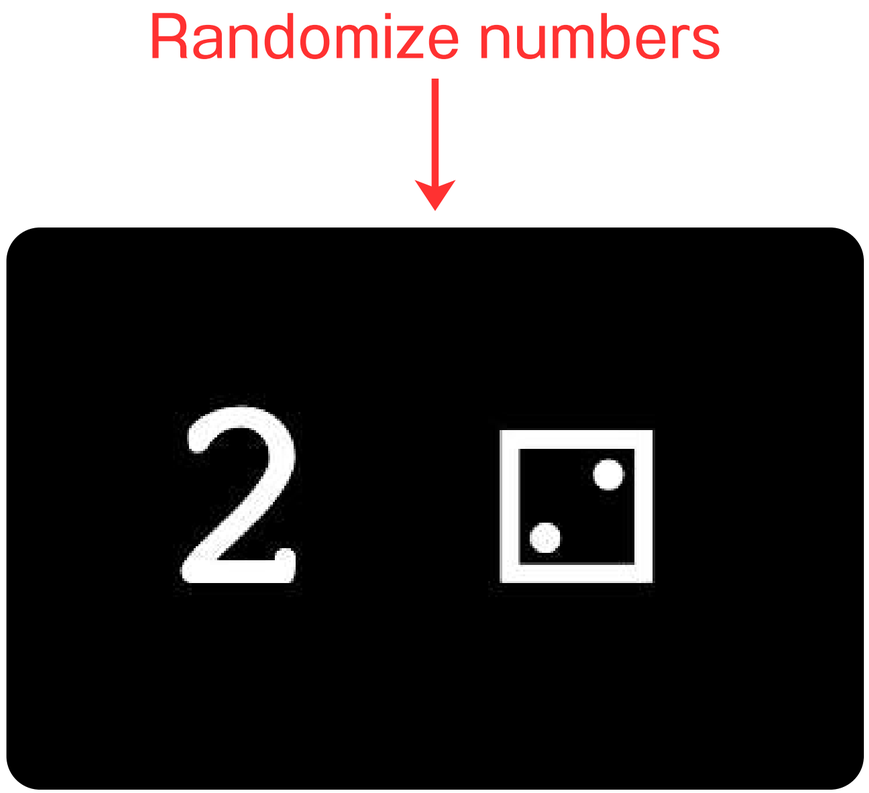

Randomize numbers



# Practice 3 : Randomize numbers and convert them to the sides of the dice

ให้เขียนโปรแกรมเพื่อสุ่มตัวเลขและแปลงให้เป็นด้านของลูกเต๋า

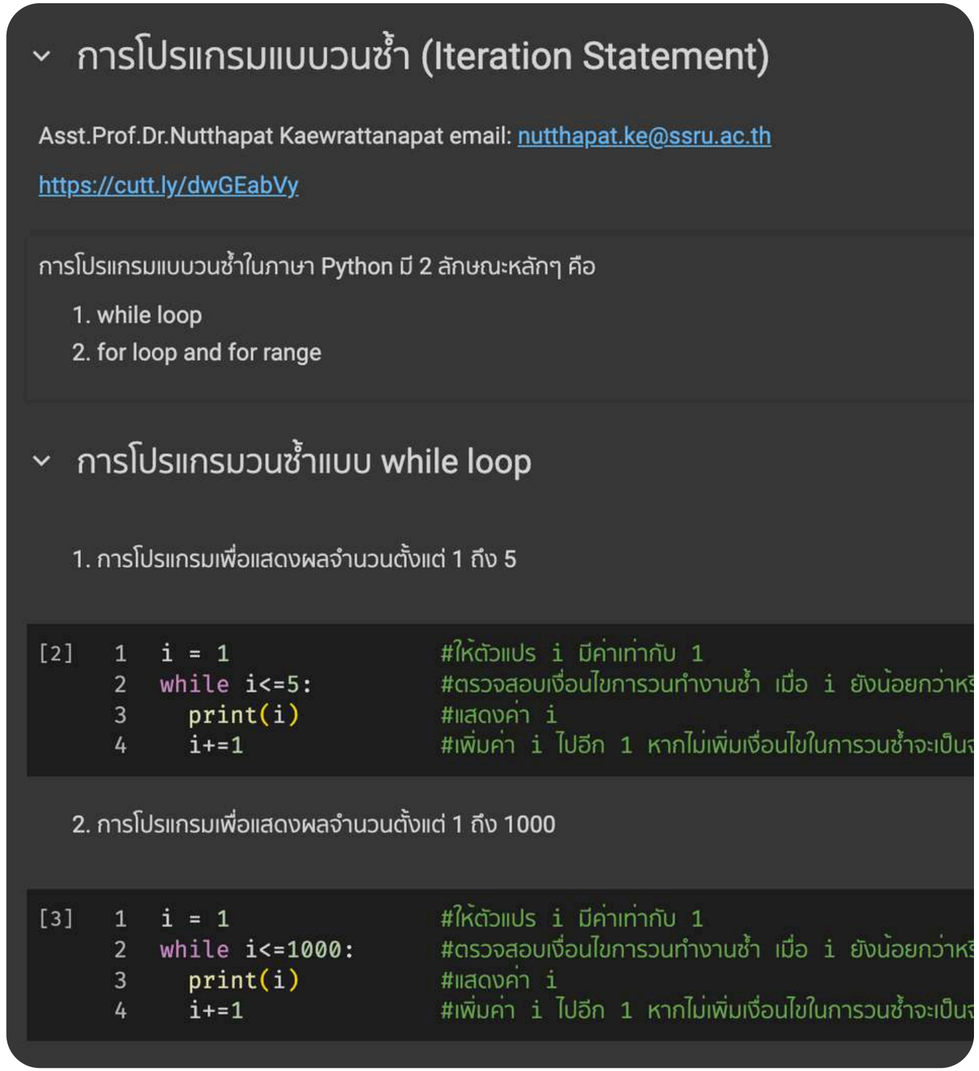
```
# Practice 3
import random
dice = random.randrange(1, 7)
if(dice==1):
   ucode = "\u2680"
elif(dice==2):
   ucode = "\u2681"
elif(dice==3):
   ucode = "\u2682"
elif(dice==4):
   ucode = "\u2683"
elif(dice==5):
   ucode = "\u2684"
elif(dice==6):
   ucode = "\u2685"
print(dice,ucode)
```



# Learning Materials – References

## ตัวอย่าง Code

https://cutt.ly/dwGEabVy



การโปรแกรมแบบวนซ้ำ (Iteration Statement)

Asst.Prof.Dr.Nutthapat Kaewrattanapat email: nutthapat.ke@ssru.ac.th

https://cutt.ly/dwGEabVy

การโปรแกรมแบบวนซ้ำในภาษา Python มี 2 ลักษณะหลักๆ คือ

1. while loop
2. for loop and for range

การโปรแกรมวนซ้ำแบบ while loop

1. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 5

```
[2]  1  i = 1                #ให้ตัวแปร i มีค่าเท่ากับ 1
     2  while i<=5:          #ตรวจสอบเงื่อนไขการวนทำงานซ้ำ เมื่อ i ยังน้อยกว่าหรื
     3    print(i)           #แสดงค่า i
     4    i+=1               #เพิ่มค่า i ไปอีก 1 หากไม่เพิ่มเงื่อนไขในการวนซ้ำจะเป็นจ
```

2. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 1000

```
[3]  1  i = 1                #ให้ตัวแปร i มีค่าเท่ากับ 1
     2  while i<=1000:       #ตรวจสอบเงื่อนไขการวนทำงานซ้ำ เมื่อ i ยังน้อยกว่าหรื
     3    print(i)           #แสดงค่า i
     4    i+=1               #เพิ่มค่า i ไปอีก 1 หากไม่เพิ่มเงื่อนไขในการวนซ้ำจะเป็นจ
```

# โครงสร้างการควบคุมโปรแกรม (Program Control Structures)

โครงสร้างการควบคุมโปรแกรม (Program Control Structures) ในการเขียนโปรแกรม มี 3 ลักษณะหลัก ได้แก่

- **การโปรแกรมแบบลำดับ (Sequential Statement)**
- **การโปรแกรมแบบเงื่อนไข (Conditional Statement)**
- **การโปรแกรมแบบทำซ้ำ (Iteration Statement)**

แต่ละลักษณะมีความสำคัญ และทำหน้าที่ที่แตกต่างกันในการควบคุมการทำงานของโปรแกรม

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

หากต้องการ**แสดงผลคำว่า “Hello, Name” ทั้งห้อง** จะต้องเขียนโปรแกรมกี่บรรทัด

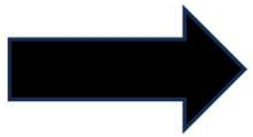
```
print("Hello, Pasawut")
print("Hello, Nutthapat")
print("Hello, Thanadol")
print("Hello, Natthapol")
print("Hello, Vilasinee")
print("Hello, Wichuda")
print("Hello, Chinchira")
print("Hello, Phachaya")
print("Hello, Manop")
```

```
Hello, Pasawut
Hello, Nutthapat
Hello, Thanadol
Hello, Natthapol
Hello, Vilasinee
Hello, Wichuda
Hello, Chinchira
Hello, Phachaya
Hello, Manop
```

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

หากต้องการ**เปลี่ยนคำว่า ”Hello” เป็นคำว่า “สวัสดี,” ทั้งห้อง** จะต้องแก้โปรแกรมกี่บรรทัด

```
print("Hello, Pasawut")
print("Hello, Nutthapat")
print("Hello, Thanadol")
print("Hello, Natthapol")
print("Hello, Vilasinee")
print("Hello, Wichuda")
print("Hello, Chinchira")
print("Hello, Phachaya")
print("Hello, Manop")
```



```
1. print("สวัสดี, Pasawut")
2. print("สวัสดี, Nutthapat")
3. print("สวัสดี, Thanadol")
4. print("สวัสดี, Natthapol")
5. print("สวัสดี, Vilasinee")
6. print("สวัสดี, Wichuda")
7. print("สวัสดี, Chinchira")
8. print("สวัสดี, Phachaya")
9. print("สวัสดี, Manop")
```

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

หากต้องการ**แสดงจำนวนตั้งแต่ 1 ถึง 10,000** จะต้องเขียนโปรแกรมกี่บรรทัด

```
1. print("1")
2. print("2")
3. print("3")
4. print("4")
5. print("5")
6. print("6")
7. print("7")
8. print("8")
9. print("9")
10.print("10")
11.print("11")
12.print("12")
13.print("13")
14.print("14")
15.print("15")
16.print("16")
17.print("17")
18.print("18")
19.print("19")
20.print("20")
21.print("21")
22.print("22")
23.print("23")
24.print("24")
25.print("25")
26.print("26")
9999.print("9999")
10000.print("10000")
```

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

การโปรแกรมแบบทำซ้ำ (Iteration Statement) เป็นเทคนิคในการเขียนโปรแกรมให้ทำงานซ้ำๆ กันหลายรอบ มีประโยชน์ในการเขียนโปรแกรมที่ต้องทำซ้ำ เช่น

1. การพิมพ์ข้อมูลทีละรายการ
2. การคำนวณจำนวนที่มีหลายค่า
3. การวนซ้ำผ่านสมาชิกของชุดข้อมูล
4. การวนซ้ำทำงานตามเงื่อนไขที่กำหนดเพื่อหาข้อมูล หรือแก้ปัญหาที่ต้องมีการไล่เรียง

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

```
while loop syntax:

while condition:
    # คำสั่งที่ต้องการทำงานเมื่อเงื่อนไขเป็นจริง
```

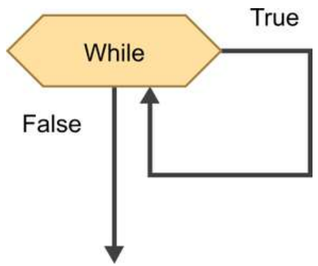
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็นเท็จ (False)

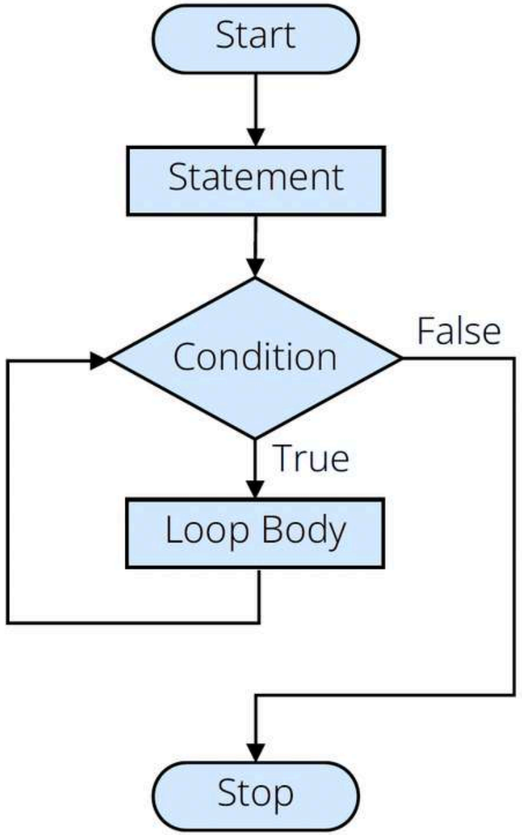
```
for loop syntax:

for item in collection:
    # คำสั่งที่ต้องการทำงานกับ item
```

การใช้ **for loop** สำหรับการวนลูปผ่านคอลเลคชันข้อมูล เช่น รายการ, สตริง , คอลเลคชันอื่นๆ และทำงานกับแต่ละรายการในคอลเลคชัน

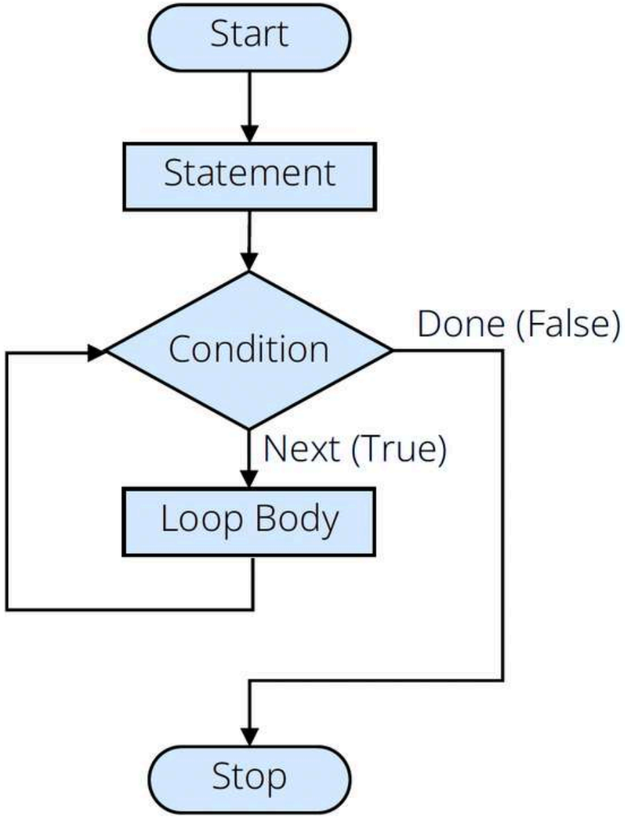
# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

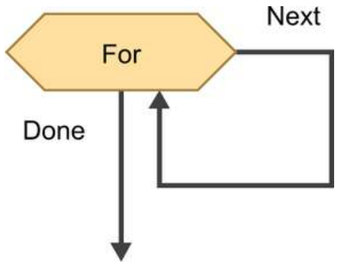
## โครงสร้างการควบคุมทำซ้ำแบบ while
## While Statement



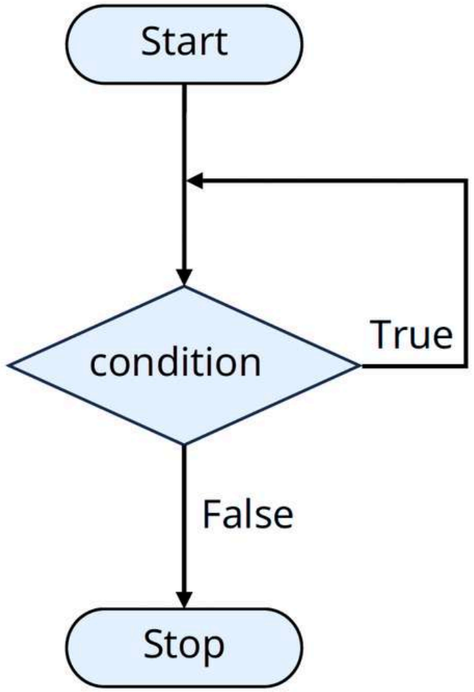

## โครงสร้างการควบคุมทำซ้ำแบบ for
## For Statement

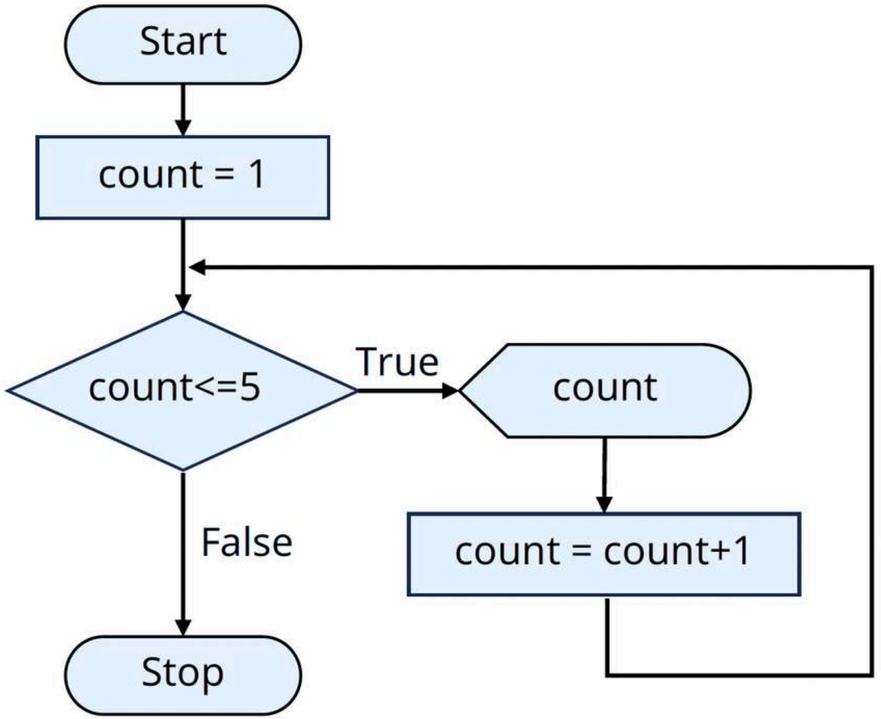


# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



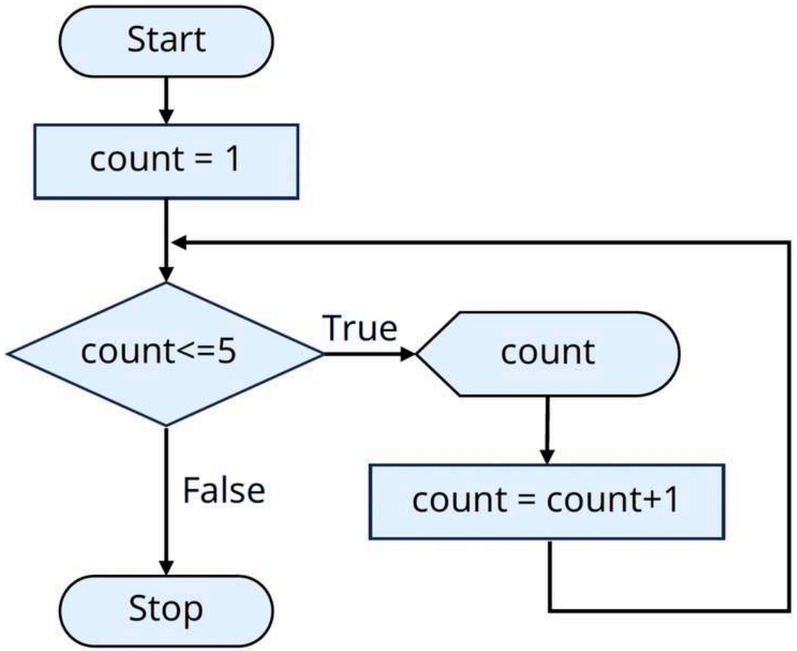




Python

```
1. count = 1
2. while count<=5:
3.    print(count)
4.    count+=1
```

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

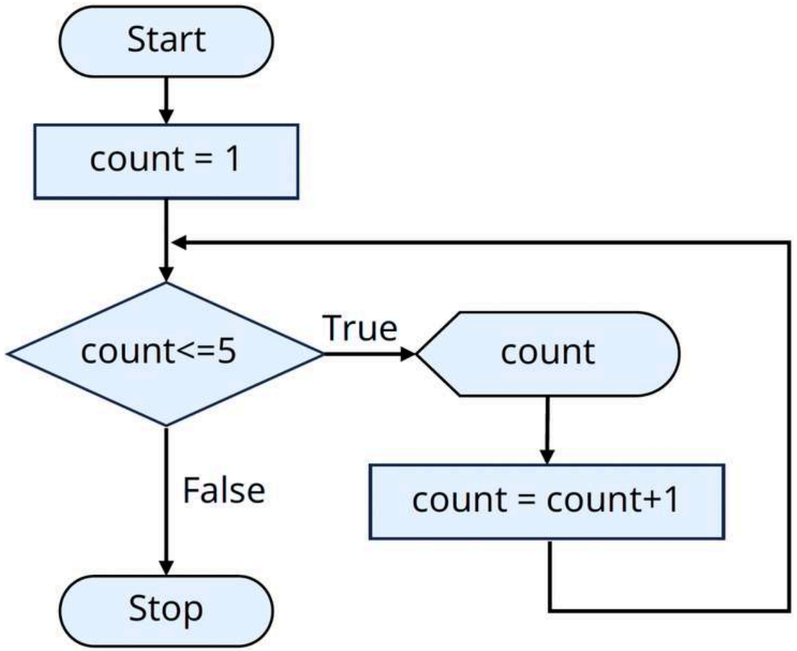
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

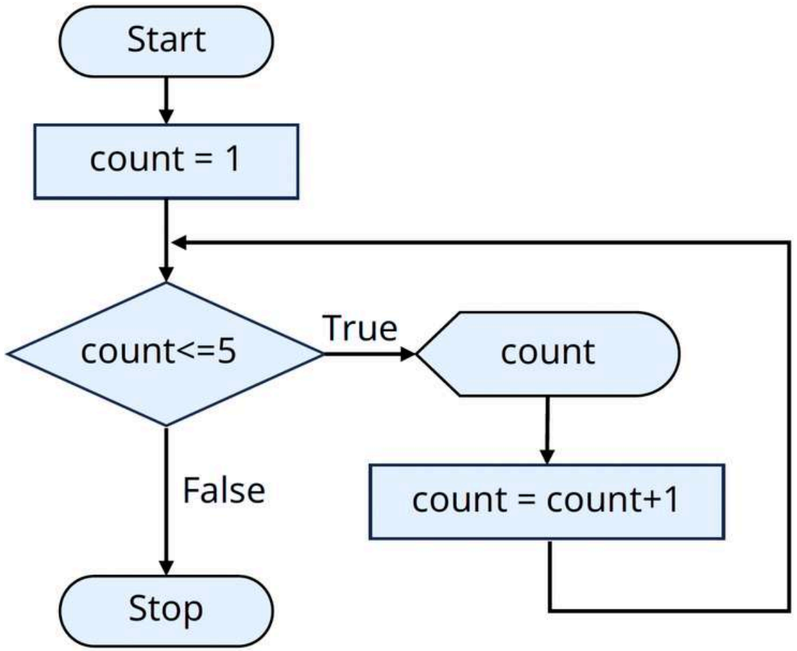
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| 2 | 2 | TRUE | 2 |
| | | | |
| | | | |
| | | | |
| | | | |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

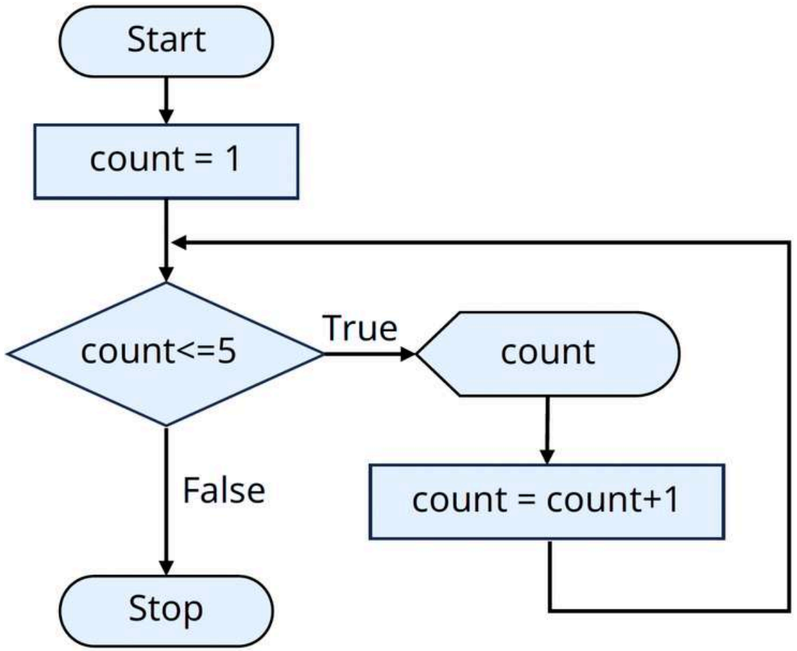
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| 2 | 2 | TRUE | 2 |
| 3 | 3 | TRUE | 3 |
| | | | |
| | | | |
| | | | |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

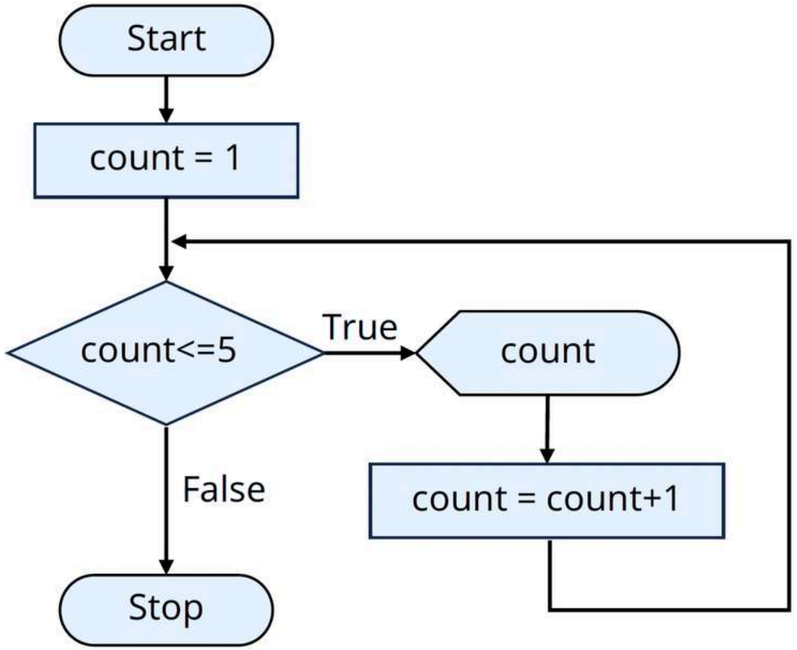
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| 2 | 2 | TRUE | 2 |
| 3 | 3 | TRUE | 3 |
| 4 | 4 | TRUE | 4 |
| | | | |
| | | | |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

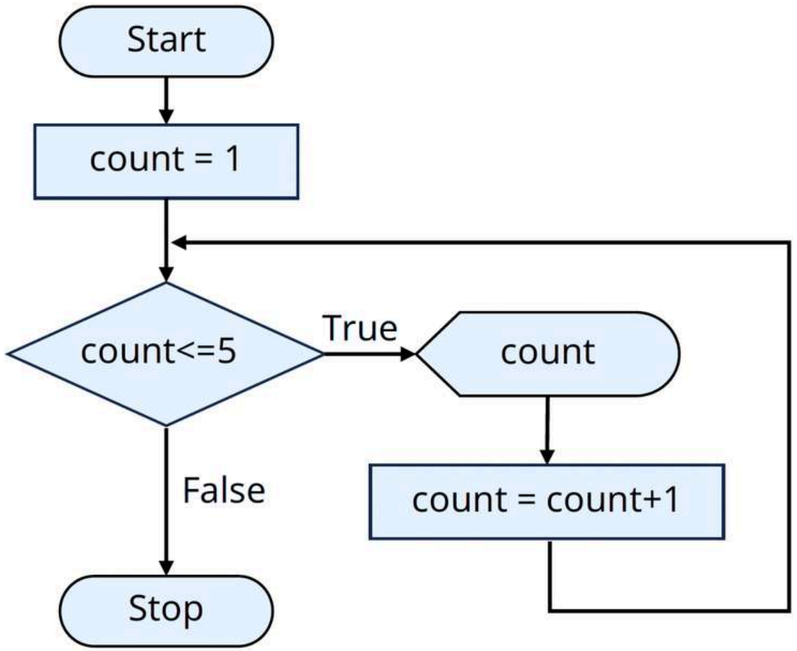
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| 2 | 2 | TRUE | 2 |
| 3 | 3 | TRUE | 3 |
| 4 | 4 | TRUE | 4 |
| 5 | 5 | TRUE | 5 |
| | | | |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

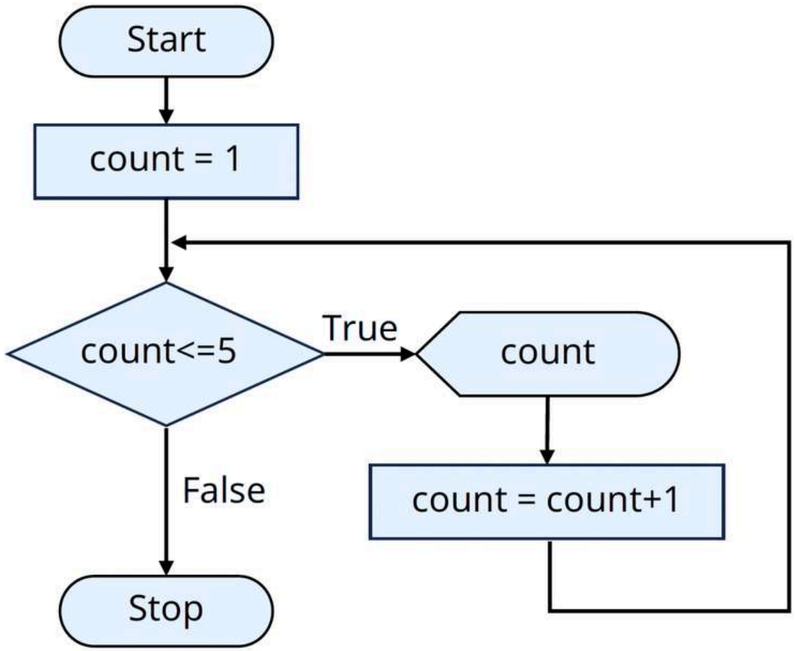
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| 2 | 2 | TRUE | 2 |
| 3 | 3 | TRUE | 3 |
| 4 | 4 | TRUE | 4 |
| 5 | 5 | TRUE | 5 |
| 6 | 6 | FALSE | - |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

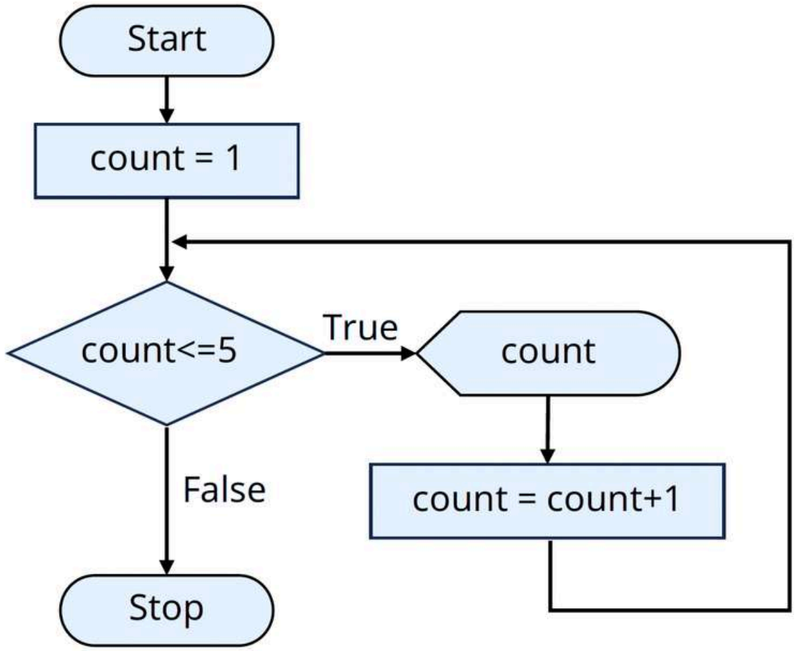
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**




| รอบที่ | ค่าของตัวแปร count | เงื่อนไข (True, False) | แสดงผล |
|---|---|---|---|
| 1 | 1 | TRUE | 1 |
| 2 | 2 | TRUE | 2 |
| 3 | 3 | TRUE | 3 |
| 4 | 4 | TRUE | 4 |
| 5 | 5 | TRUE | 5 |
| ~~6~~ | 6 | ~~FALSE~~ | - |

# การโปรแกรมแบบทำซ้ำ (Iteration Statement)

1. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 5



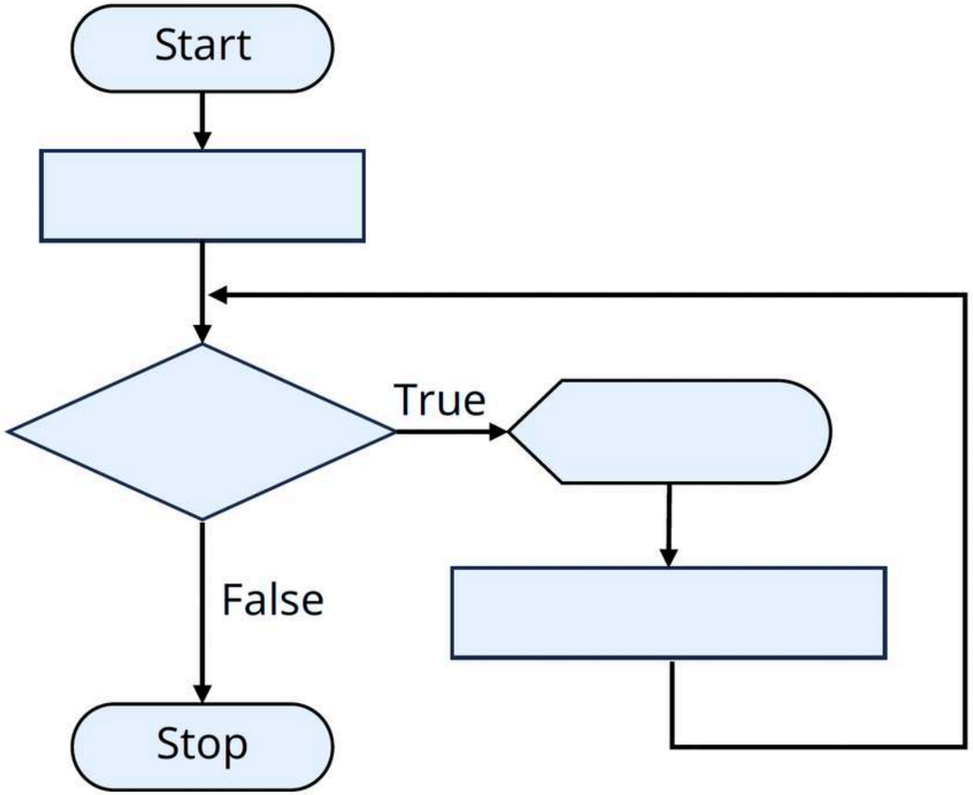

```
1. count = 1
2. while count<=5:
3.    print(count)
4.    count+=1
```

**Python Tutor**

https://cutt.ly/DwGxR4R2

# Iteration Statement – while

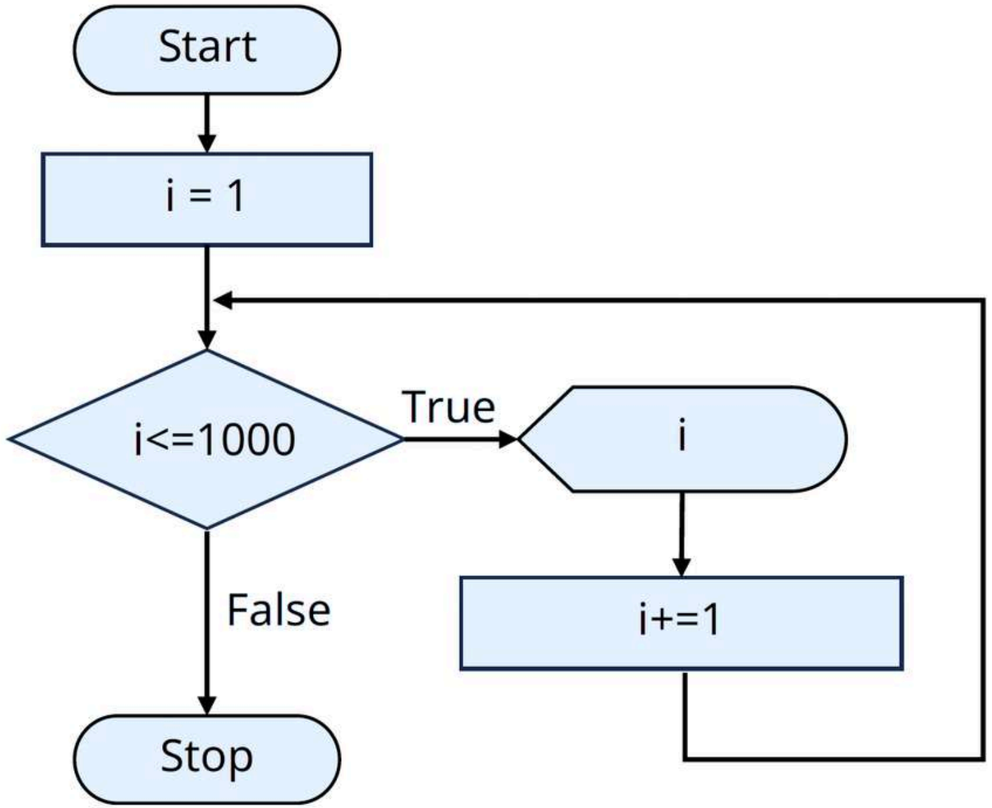
2. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 1000




```
1.          =
2.   while          :
3.     print(       )
4.             +=1
```

# Iteration Statement – while

2. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 1000




```
1. i = 1
2. while i<=1000:
3.     print(i)
4.     i+=1
```

## Iteration Statement – while

3. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 5 ถึง 10 ให้เติมโปรแกรมให้ถูกต้อง

```
1. i =
2. while i     :
3.    print( )
4.    i+=
```

## Iteration Statement – while

3. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 5 ถึง 10 ให้เติมโปรแกรมให้ถูกต้อง

```
1. i = 5
2. while i<=10 :
3.    print(i)
4.    i+=1
```

## Iteration Statement – while

4. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 2, 4, 6, ... ถึง 20 ให้เติมโปรแกรมให้ถูกต้อง

```
1. i =
2. while i     :
3.    print( )
4.    i+=
```

## Iteration Statement – while

4. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 2, 4, 6, ... ถึง 20 ให้เติมโปรแกรมให้ถูกต้อง

```
1. i = 2
2. while i<=20:
3.     print(i)
4.     i+=2
```

## Iteration Statement – while

5. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1, 3, 5, 7, ... ถึง 19 ให้เติมโปรแกรมให้ถูกต้อง

```
1. i =
2. while       :
3.    print(i)
4.    i
```

## Iteration Statement – while

5. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1, 3, 5, 7, ... ถึง 19 ให้เติมโปรแกรมให้ถูกต้อง

```
1. i = 1
2. while i<=19:
3.     print(i)
4.     i+=2
```

## Iteration Statement – while

6. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 50, 49, 48, ... 0

```
1. i =
2. while       :
3.    print(i)
4.    i
```

## Iteration Statement – while

6. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 50, 49, 48, ... 0

```
1. i = 50
2. while i>=0:
3.    print(i)
4.    i-=1
```

## Iteration Statement – while

7. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

```
1. i =
2. while i<=50:
3.     if i == 0:
4.         print(i)
5.     i
```

## Iteration Statement – while

7. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

```
1. i = 1
2. while i<=50:
3.     if i%2 == 0:
4.         print(i)
5.     i+=1
```

# Iteration Statement – while

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

```
1. i =
2. count =
3. while i :
4.    if == 0:
5.       print(i)
6.       count
7.    i
8. print("มีทั้งหมด", count, "จำนวน")
```

# Iteration Statement – while

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

```
1. i = 1
2. count = 0
3. while i<=50:
4.     if i%2 == 0:
5.         print(i)
6.         count+=1
7.     i+=1
8. print("มีทั้งหมด", count, "จำนวน")
```

# Iteration Statement – while

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนที่หาร 5 ลงตัว ตั้งแต่ 1 ถึง 100 และสรุปว่ามีทั้งหมดกี่จำนวน

```
1. i =
2. count = 0
3. while i     :
4.   if        :
5.     print( )
6.     count
7.   i
8. print("มีทั้งหมด", count, "จำนวน")
```

## Iteration Statement – while

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนที่หาร 5 ลงตัว ตั้งแต่ 1 ถึง 100 และสรุปว่ามีทั้งหมดกี่จำนวน

```
1. i = 1
2. count = 0
3. while i<=100:
4.    if i%5==0:
5.       print(i)
6.       count+=1
7.    i+=1
8. print("มีทั้งหมด", count, "จำนวน")
```

# Iteration Statement – while

10. การโปรแกรมเพื่อหาผลรวมของจำนวนตั้งแต่ 1, 2, 3, ... ถึง 10

```
1. i =
2. n =
3. summation =
4. while        :
5.   summation
6.   i
7. print(summation)
```

# Iteration Statement - while

10. การโปรแกรมเพื่อหาผลรวมของจำนวนตั้งแต่ 1, 2, 3, ... ถึง 10

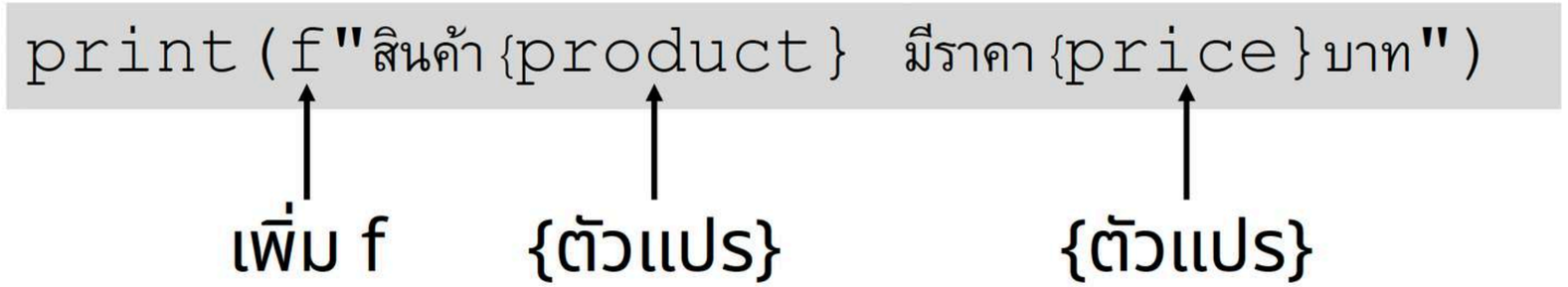
```
1. i = 1
2. n = 10
3. summation = 0
4. while i<=n:
5.    summation += i
6.    i+=1
7. print(summation)
```

# Iteration Statement – while

11. การโปรแกรมเพื่อหาค่าเฉลี่ย (average) โดยค่าเฉลี่ยมาจาก ผลรวมของจำนวนทั้งหมด หารด้วยจำนวนทั้งหมด หรือ average = summation/count

```python
# กำหนดค่าเริ่มต้นของตัวแปร
i = 1
n = 10
summation = 0
count = 0
average = 0

# ใช้ลูป while เพื่อหาผลรวมของตัวเลข 1 ถึง 10
while i <= n:
    summation += i  # เพิ่มค่า i เข้าไปใน summation
    i += 1          # เพิ่มค่า i ไป 1
    count += 1      # เพิ่มค่า count ไป 1 เพื่อนับจำนวนตัวเลขที่ถูกบวกเข้ารวม

# คำนวณค่าเฉลี่ย
average = summation / count

# แสดงผลรวมของตัวเลข, จำนวนตัวเลขที่ถูกบวกเข้ารวม, และค่าเฉลี่ย
print(f"ผลรวมของตัวเลข 1 ถึง {n} คือ {summation}")
print(f"จำนวนตัวเลขทั้งหมดคือ {count}")
print(f"ค่าเฉลี่ยของตัวเลข 1 ถึง {n} คือ {average}")
```

# การใช้ f string เบื้องต้น




```
product = "Apple"
price = 50
print(f"สินค้า {product} มีราคา {price}บาท")
```

# Iteration Statement - while

12. การโปรแกรมเพื่อแสดงจำนวน * เท่ากับจำนวนของตัวเลขตั้งแต่ 1 ถึง 10

```
1. i = 1
2. star = 1
3. while i<=10:
4.     while star<=i:
5.         print("*", end="")
6.         star+=1
7.     i+=1
8.     star = 1
9.     print()
```

```
*
**
***
****
*****
******
*******
********
*********
**********
```

# Iteration Statement – while

## 12. การโปรแกรมเพื่อแสดงจำนวน * เท่ากับจำนวนของตัวเลขตั้งแต่ 1 ถึง 10

```
1. i = 1
2. star = 1
3. while i<=10:
4.     while star<=i:
5.         print("*", end="")
6.         star+=1
7.     i+=1
8.     star = 1
9.     print()
```

| iteration: i | star<=i | print |
|---|---|---|
| 1 | **1**<=1 | * |
| 2 | **1**<=2 | ** |
| 3 | **1**<=3 | *** |
| 4 | **1**<=4 | **** |
| 5 | **1**<=5 | ***** |
| 6 | **1**<=6 | ****** |
| 7 | **1**<=7 | ******* |
| 8 | **1**<=8 | ******** |
| 9 | **1**<=9 | ********* |
| 10 | **1**<=10 | ********** |

# Iteration Statement – while

## 12. การโปรแกรมเพื่อแสดงจำนวน * เท่ากับจำนวนของตัวเลขตั้งแต่ 1 ถึง 10

```
1.    # กำหนดค่าเริ่มต้นของตัวแปร i และ star
2.    i = 1              # ตัวแปร i ใช้ในลูปเพื่อควบคุมจำนวนบรรทัด
3.    star = 1           # ตัวแปร star ใช้ในลูปภายในเพื่อควบคุมจำนวนดาว

4.    # ใช้ลูป while เพื่อสร้างรูปทรงสามเหลี่ยมดาว
5.    while i <= 10:
6.        while star <= i:
7.            print("*", end="")   # แสดงดาวและไม่ขึ้นบรรทัดใหม่ด้วย end=""
8.            star += 1   # เพิ่มค่า star ไป 1 เพื่อเพิ่มจำนวนดาวในแต่ละบรรทัด
9.        i += 1      # เพิ่มค่า i ไป 1 เพื่อเลื่อนจาก 1 ไปเรื่อยๆ ถึง 10
10.       star = 1    # กำหนดค่าเริ่มต้นของตัวแปร star ใหม่
11.       print()     # ขึ้นบรรทัดใหม่
```

```
*
**
***
****
*****
******
*******
********
*********
**********
```

## คำสั่งทำซ้ำ string

13. ในภาษา Python มีความพิเศษเกี่ยวกับคำสั่งที่ใช้ในการเพิ่มจำนวนข้อความ

```
print("*"*2)
print("*"*5)
print("*"*10)
print("*"*15)
```

```
**
*****
**********
***************
```

# Iteration Statement – while

14. การโปรแกรมเพื่อแสดงจำนวน * เท่ากับจำนวนของตัวเลขตั้งแต่ 1 ถึง 10 โดยใช้การคูณข้อความ

```
1. i = 1
2. while i<=10:
3.     print("*" *i)
4.     i+=1
```

| iteration: i | star |
|---|---|
| 1 | print("*" * 1) |
| 2 | print("*" * 2) |
| 3 | print("*" * 3) |
| 4 | print("*" * 4) |
| 5 | print("*" * 5) |
| 6 | print("*" * 6) |
| 7 | print("*" * 7) |
| 8 | print("*" * 8) |
| 9 | print("*" * 9) |
| 10 | print("*" * 10) |

```
*
**
***
****
*****
******
*******
********
*********
**********
```

## Iteration Statement – while

15. การโปรแกรมเพื่อหาค่า Factorial 3! โดยที่ Factorial คือการหาผลคูณของจำนวนที่ต้องการ เช่น

- **3! มีค่าเท่ากับ 3x2x1 ได้ผลลัพธ์เท่ากับ 6**
- **5! มีค่าเท่ากับ 5x4x3x2x1 ได้ผลลัพธ์เท่ากับ 120**

```python
# กำหนดค่าตัวแปร fac เพื่อระบุตัวเลขที่ต้องการหาค่า Factorial
fac = 3

# กำหนดค่าเริ่มต้นของตัวแปร i และ result
i = 1          # ตัวแปร i ใช้ในการวนลูปเพื่อคูณตัวเลขต่อเนื่อง
result = 1     # ตัวแปร result ใช้เก็บผลคูณของตัวเลขต่อเนื่อง

# ใช้ลูป while เพื่อคำนวณ Factorial ของตัวเลข fac
while i <= fac:
    result = result * i  # คูณตัวแปร result ด้วยตัวแปร i เพื่อคำนวณ Factorial
    i += 1               # เพิ่มค่า i ไป 1 เพื่อคูณตัวเลขถัดไป

# แสดงผลลัพธ์ Factorial
print(f"{fac}! = {result}")
```

# Iteration Statement – while

16. การโปรแกรมเพื่อแสดงจำนวนที่ได้จาก Factorial 3! เช่น

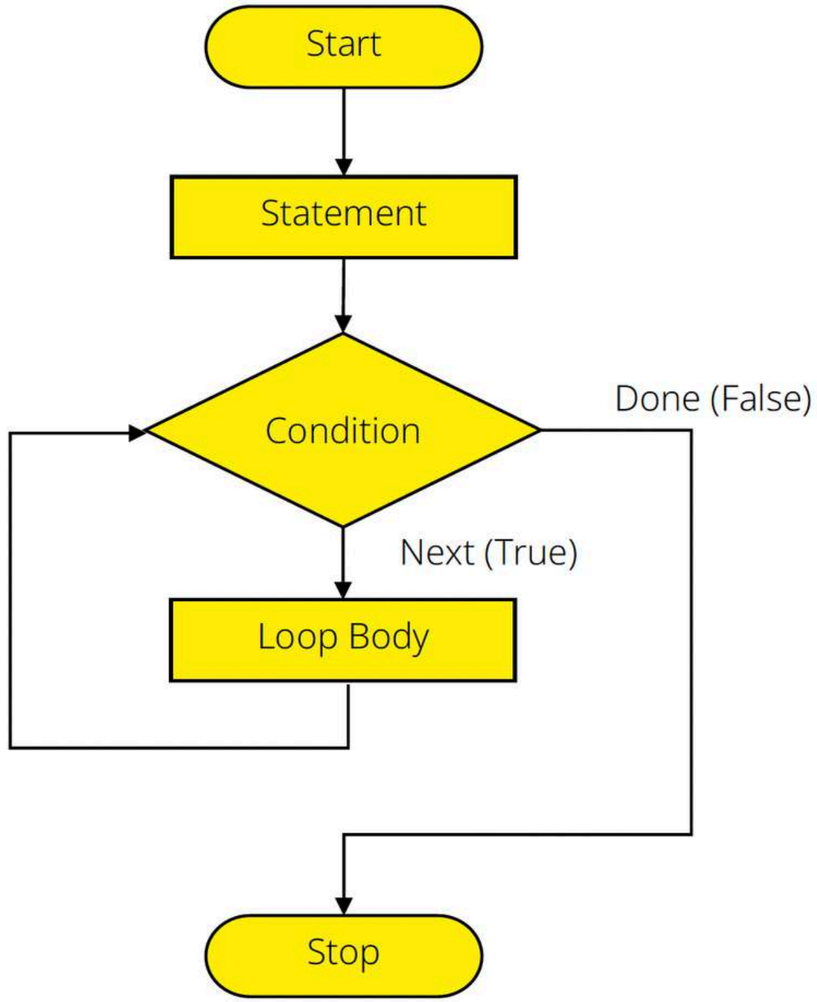
- 3! แสดงผล **3! = 3x2x1 = 6**

```
# กำหนดค่าตัวแปร fac และ result เริ่มต้น
fac = 3
result = 1

# แสดงข้อความและตัวแปร fac และตัวแปร result ที่เริ่มต้น
print(fac, "! = ", sep="", end="")

# ใช้ลูป while เพื่อคำนวณ Factorial
while fac > 0:
    if fac != 1:
        # แสดงตัวเลขและสัญลักษณ์คูณถ้าไม่ใช่ตัวสุดท้าย
        print(fac, "x", sep="", end="")
    else:
        # แสดงตัวสุดท้ายโดยไม่มีสัญลักษณ์คูณ
        print(fac, end="")

    # คำนวณผลลัพธ์ Factorial
    result = result * fac

    # ลดค่า fac ลง 1 ในทุกรอบ
    fac -= 1

# แสดงผลลัพธ์ของ Factorial
print(" = ", result, sep="")
```

# Iteration Statement – for

การใช้ **for loop** สำหรับการวนลูปผ่านคอลเลคชันข้อมูล เช่น ระยะ, รายการ, สตริง , คอลเลคชันอื่นๆ และทำงานกับแต่ละรายการในคอลเลคชัน

**โครงสร้างการควบคุมทำซ้ำแบบ for**
**For Statement**



## Iteration Statement – for

range() เป็นฟังก์ชันในภาษา Python เพื่อสร้างชุดของตัวเลขแบบติดต่อกัน (sequence of numbers) โดยมักใช้ในการสร้างลูป (loop) เพื่อทำงานกับข้อมูลที่มีลำดับหรือเป็นตัวเลขที่เรียงต่อกัน

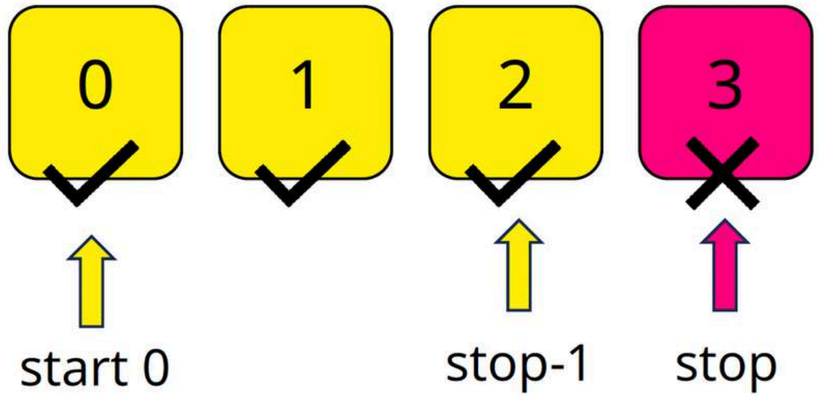
**range() มีรูปแบบการใช้งาน 3 ลักษณะ ดังนี้**

1. **range(stop):** สร้างชุดตัวเลขตั้งแต่ 0 ถึง stop - 1
2. **range(start, stop):** สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1
3. **range(start, stop, step):** สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1 โดยที่ลำดับของตัวเลขมีการเพิ่มขึ้นที่ละ step

# ฟังก์ชั่น range( )

**1. range(stop):** สร้างชุดตัวเลขตั้งแต่ 0 ถึง stop - 1

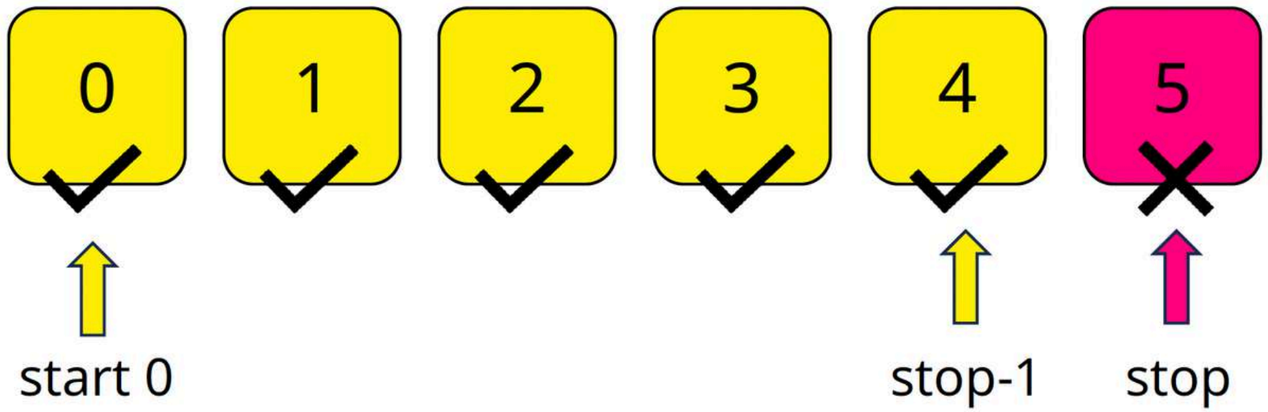
```
range(3)
```



# ฟังก์ชั่น range( )

**1. range(stop):** สร้างชุดตัวเลขตั้งแต่ 0 ถึง stop - 1

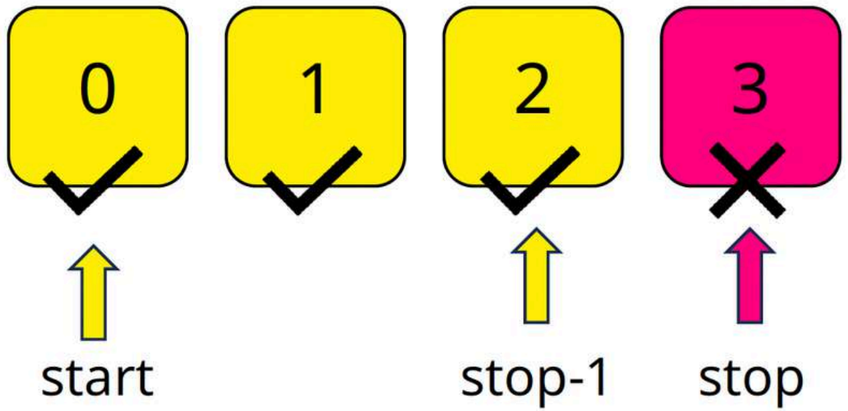
```
range(5)
```



# ฟังก์ชั่น range( )

## 2. range(start, stop): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1

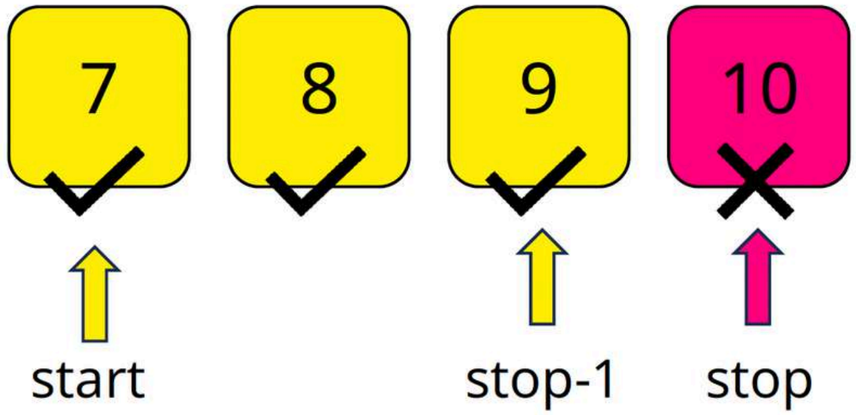
```
range(0,3)
```



# ฟังก์ชั่น range( )

## 2. range(start, stop): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1

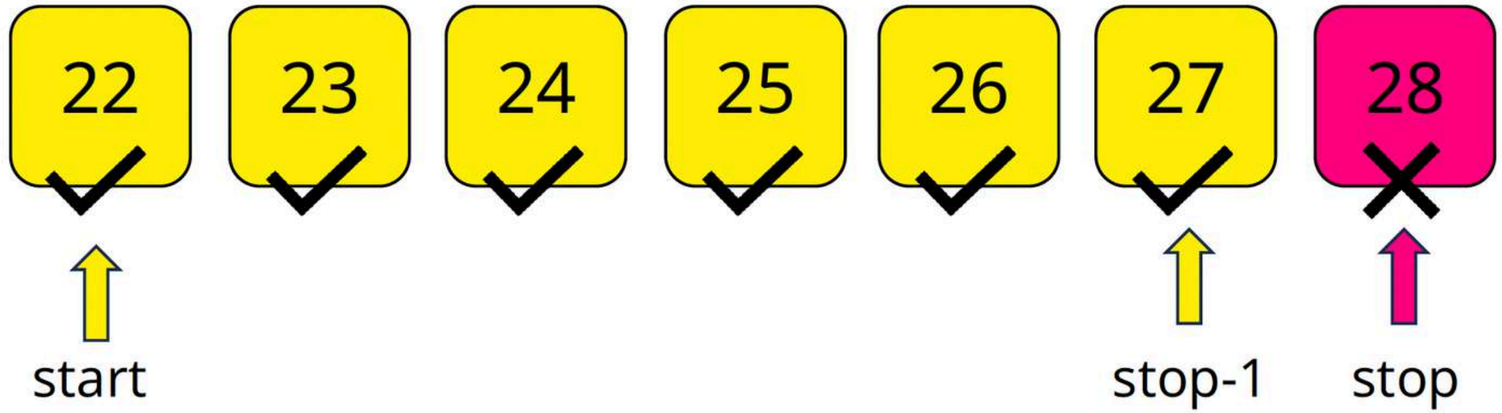
```
range(7,10)
```



# ฟังก์ชั่น range( )

## 2. range(start, stop): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1

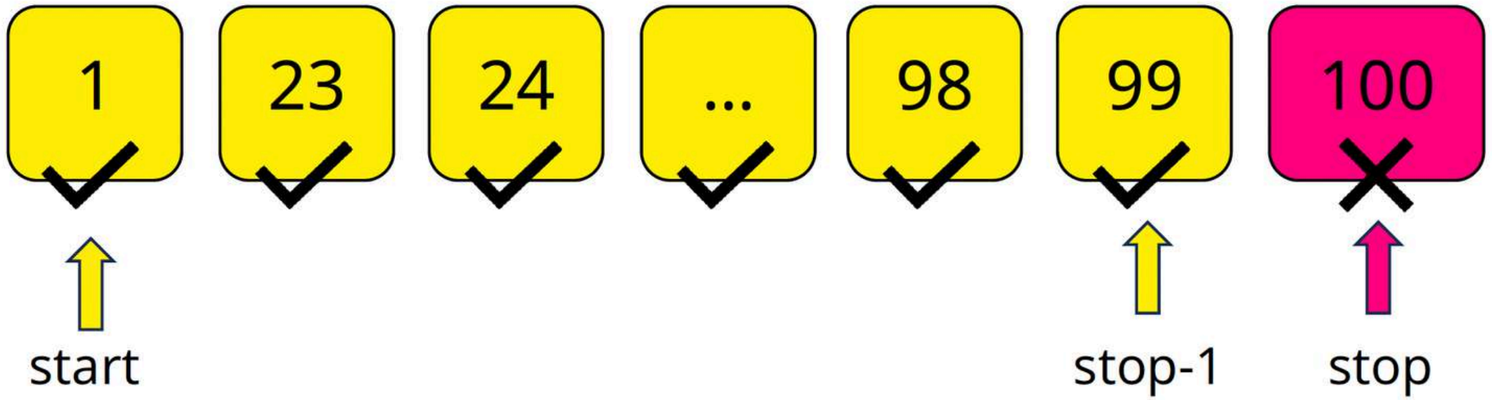
```
range(22,28)
```



# ฟังก์ชั่น range( )

## 2. range(start, stop): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1

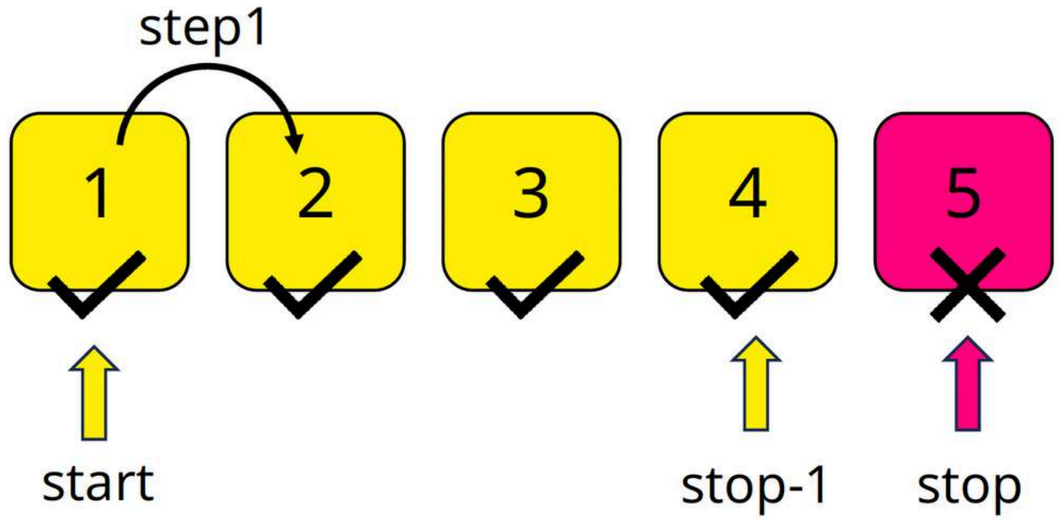
```
range(1,100)
```



# ฟังก์ชั่น range( )

**3. range(start, stop, step): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1 โดยที่ลำดับของตัวเลขมีการเพิ่มขึ้นที่ละ step**

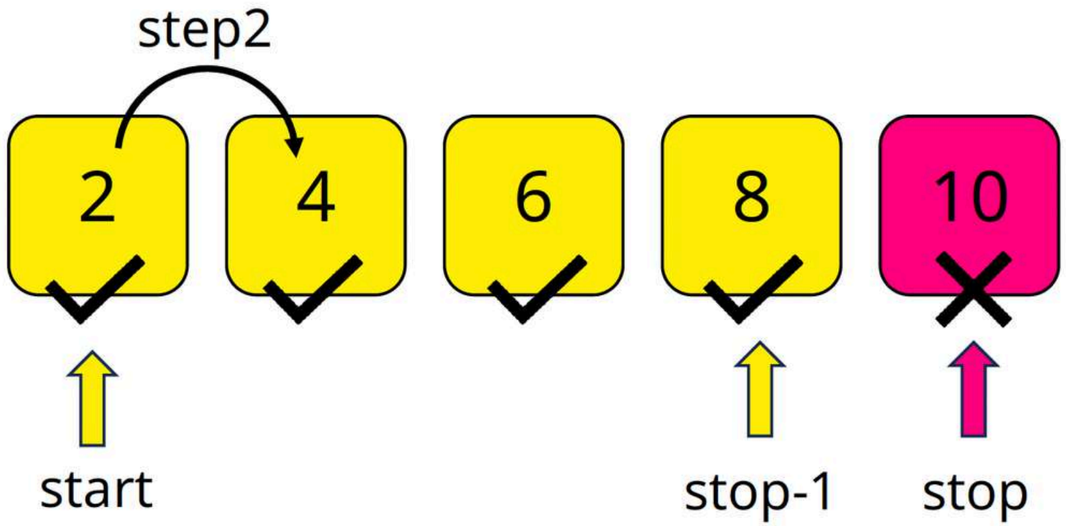
```
range(1,5,1)
```



# ฟังก์ชั่น range( )

**3. range(start, stop, step): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1 โดยที่ลำดับของตัวเลขมีการเพิ่มขึ้นที่ละ step**

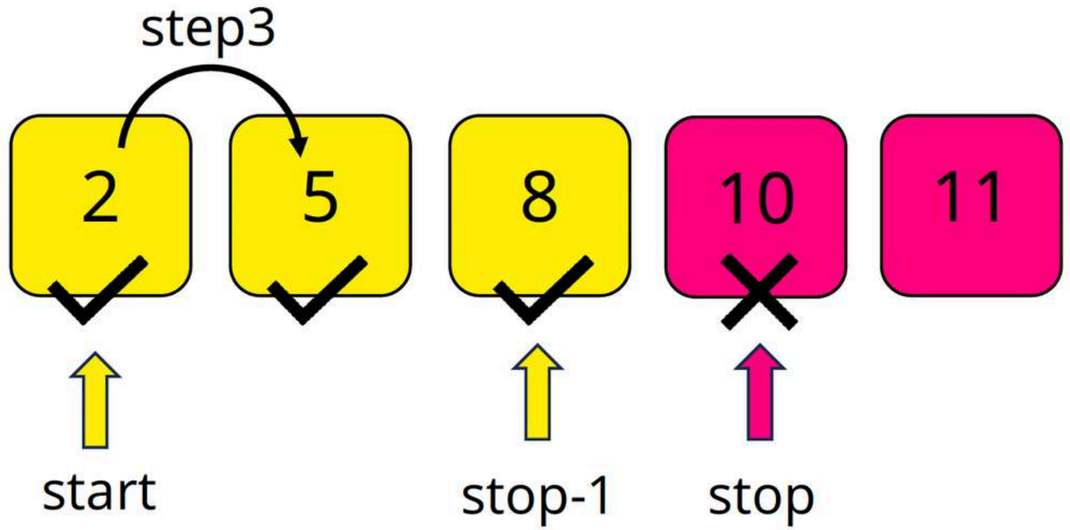
```
range(2,10,2)
```



# ฟังก์ชั่น range( )

**3. range(start, stop, step): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1 โดยที่ลำดับของตัวเลขมีการเพิ่มขึ้นที่ละ step**

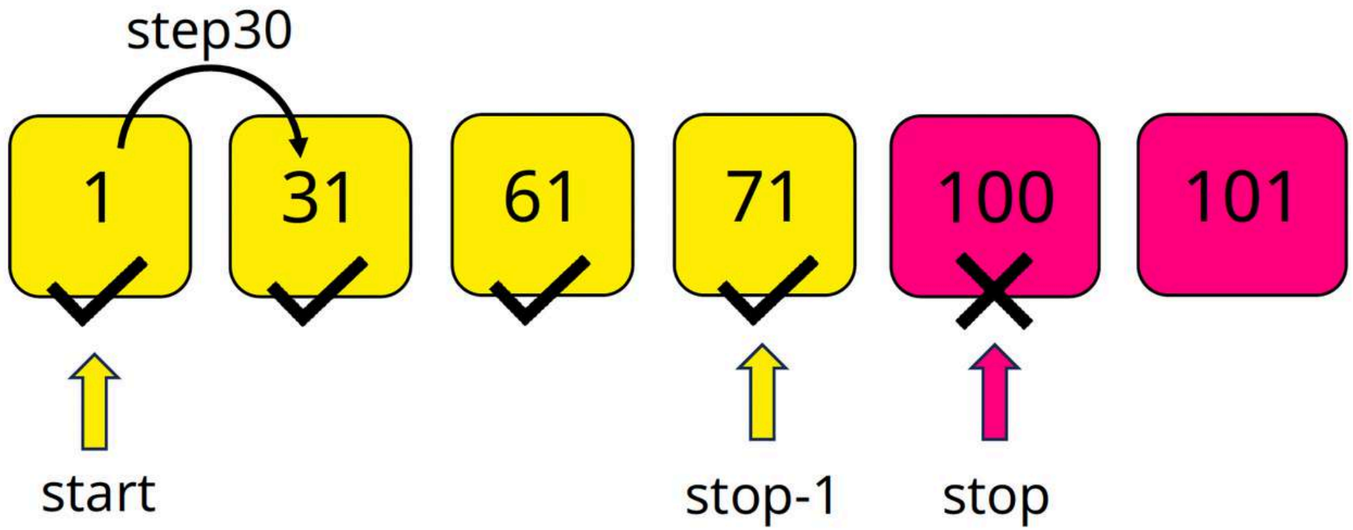
```
range(2,10,3)
```



# ฟังก์ชั่น range( )

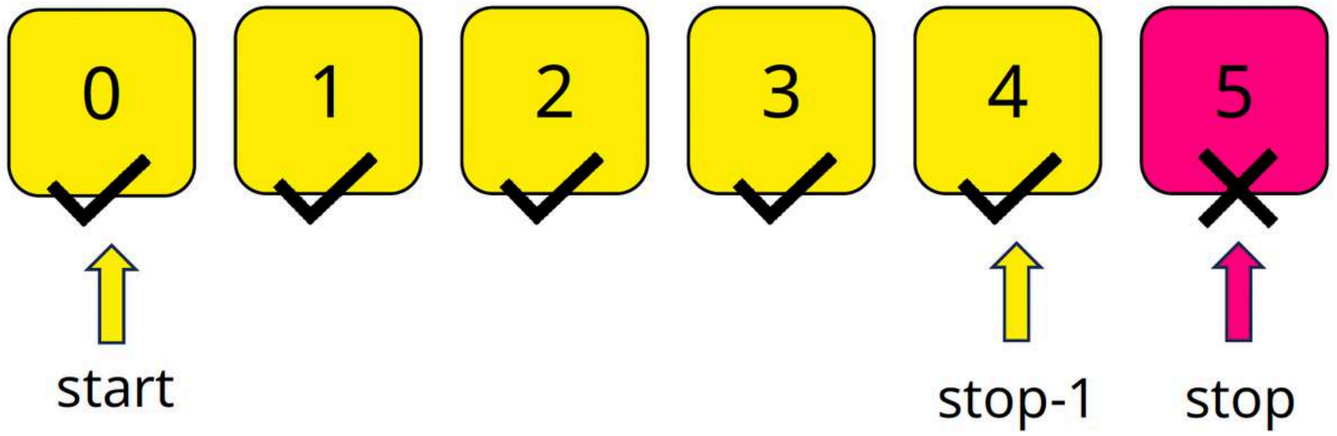
**3. range(start, stop, step): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1 โดยที่ลำดับของตัวเลขมีการเพิ่มขึ้นที่ละ step**

```
range(1,100,30)
```



# Iteration Statement – for

1. ต้องการแสดงค่าข้อมูลใน range( ) จาก 0 ถึง 4



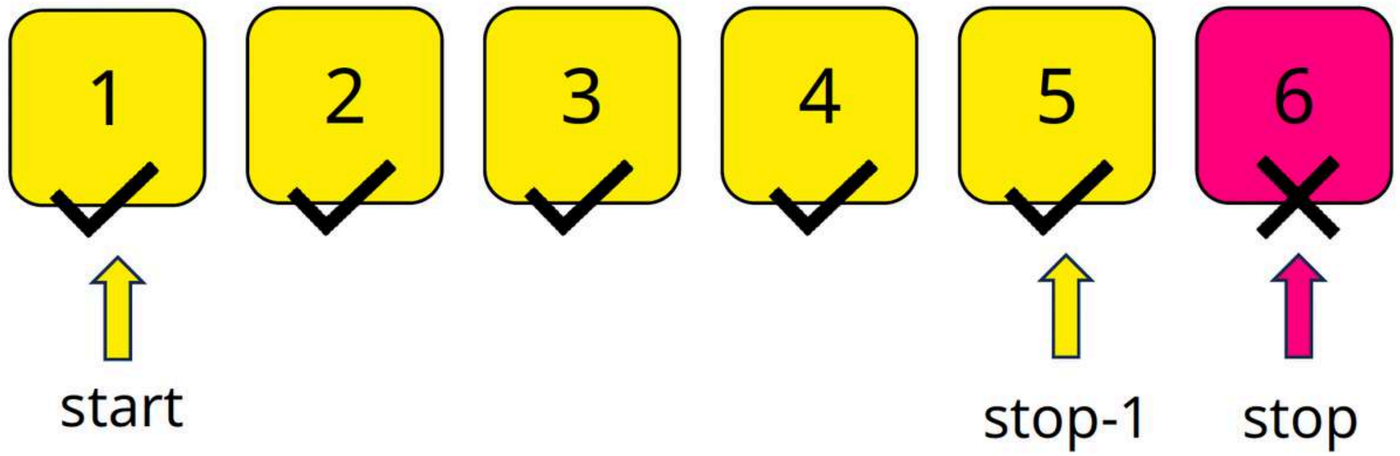

```
for i in range(5):
  print(i)
```

```
1  for i in range(5):
2    print(i)
```

```
0
1
2
3
4
```

# Iteration Statement – for

2. ต้องการแสดงค่าข้อมูลใน range( ) จาก 1 ถึง 5



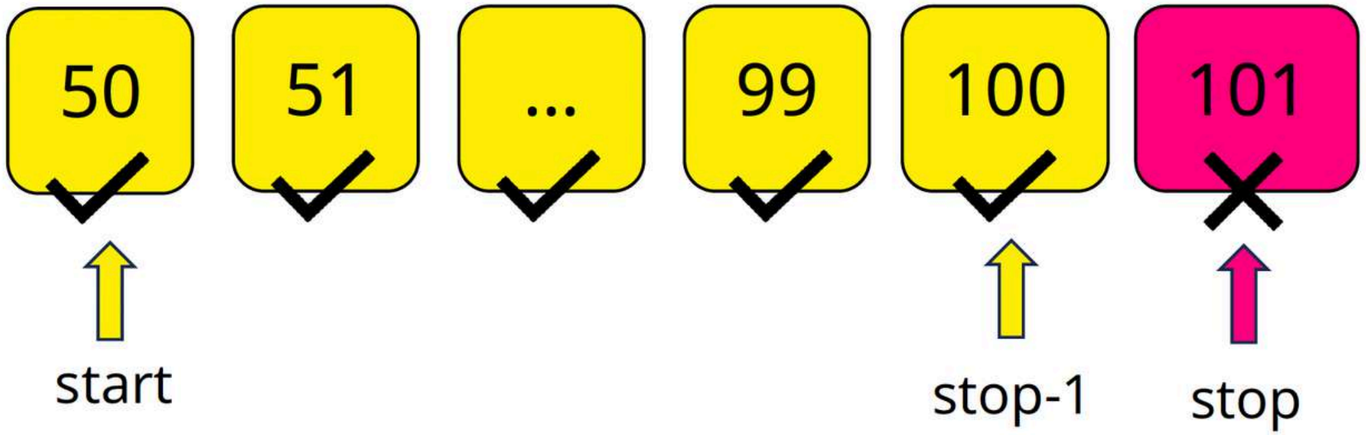

```
for i in range(1,6):
  print(i)
```

```
1  for i in range(1,6):
2    print(i)
```

```
1
2
3
4
5
```

# Iteration Statement – for

3. ต้องการแสดงค่าข้อมูลใน range( ) จาก 50 ถึง 100




```
for i in range(50,_____):
  print(i)
```

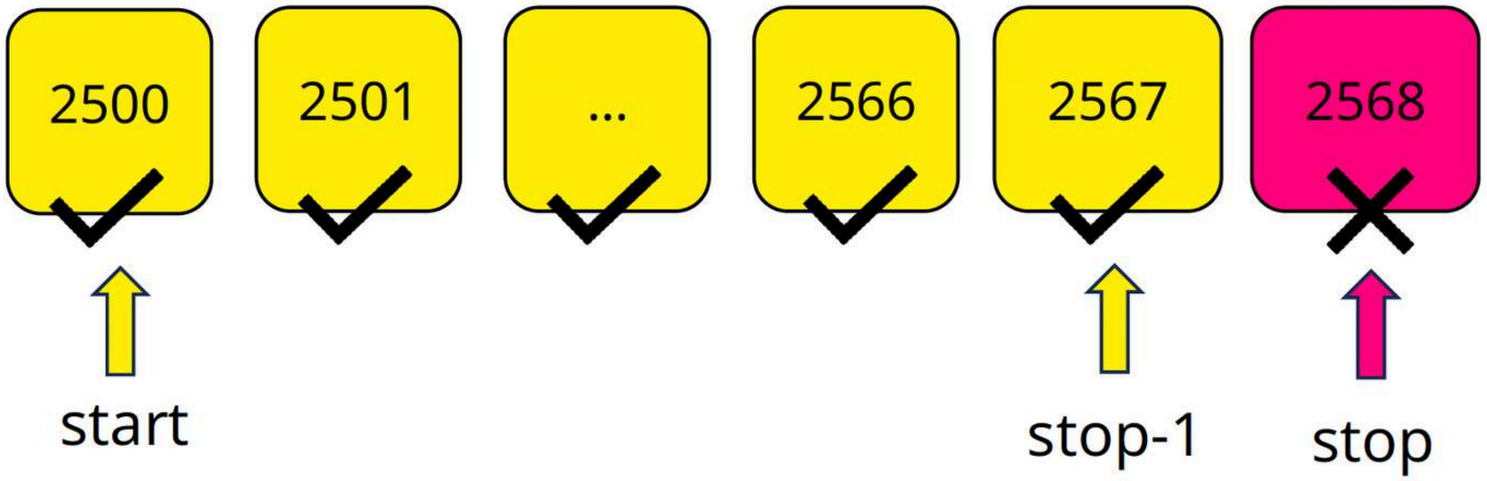
```
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
```

```
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

# Iteration Statement – for

4. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ปี พ.ศ.2500 ถึง พ.ศ.2567

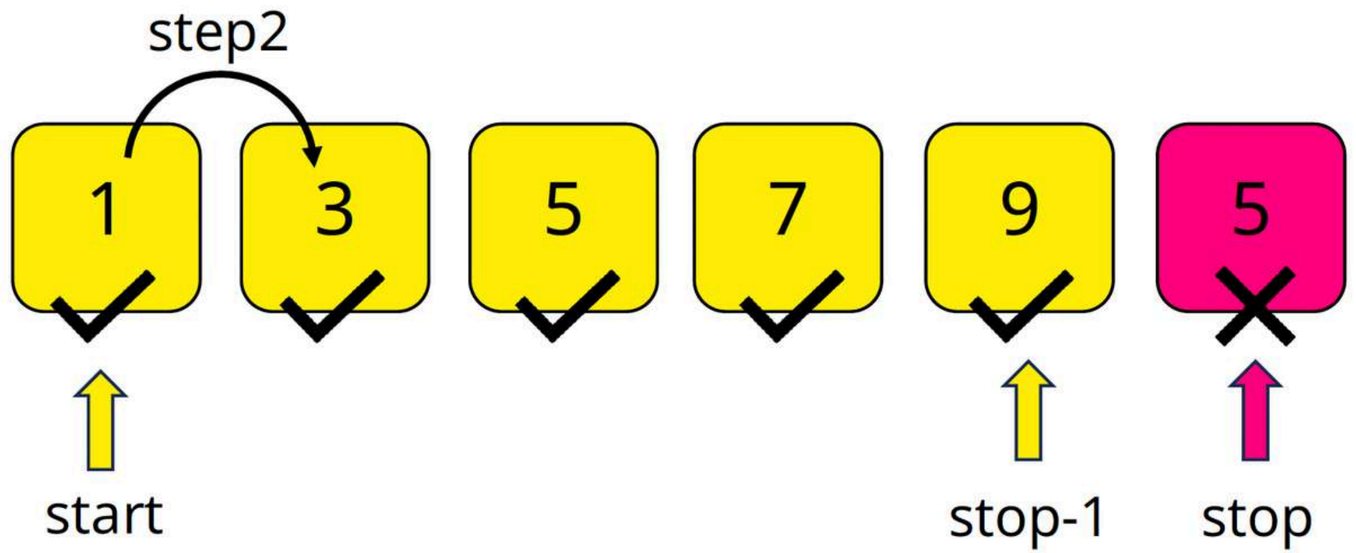
```
for i in range(_____,_____):
  print(i)
```



# Iteration Statement – for

5. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 10 เพิ่มทีละ 2

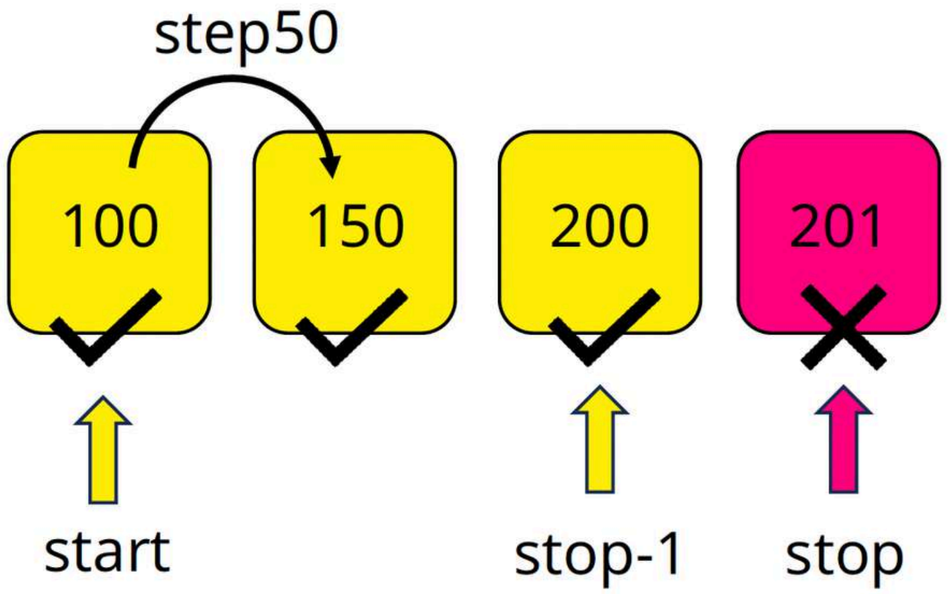
```
for i in range(_________,_____________,_________):
  print(i)
```



# Iteration Statement – for

6. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 100 ถึง 200 เพิ่มทีละ 50

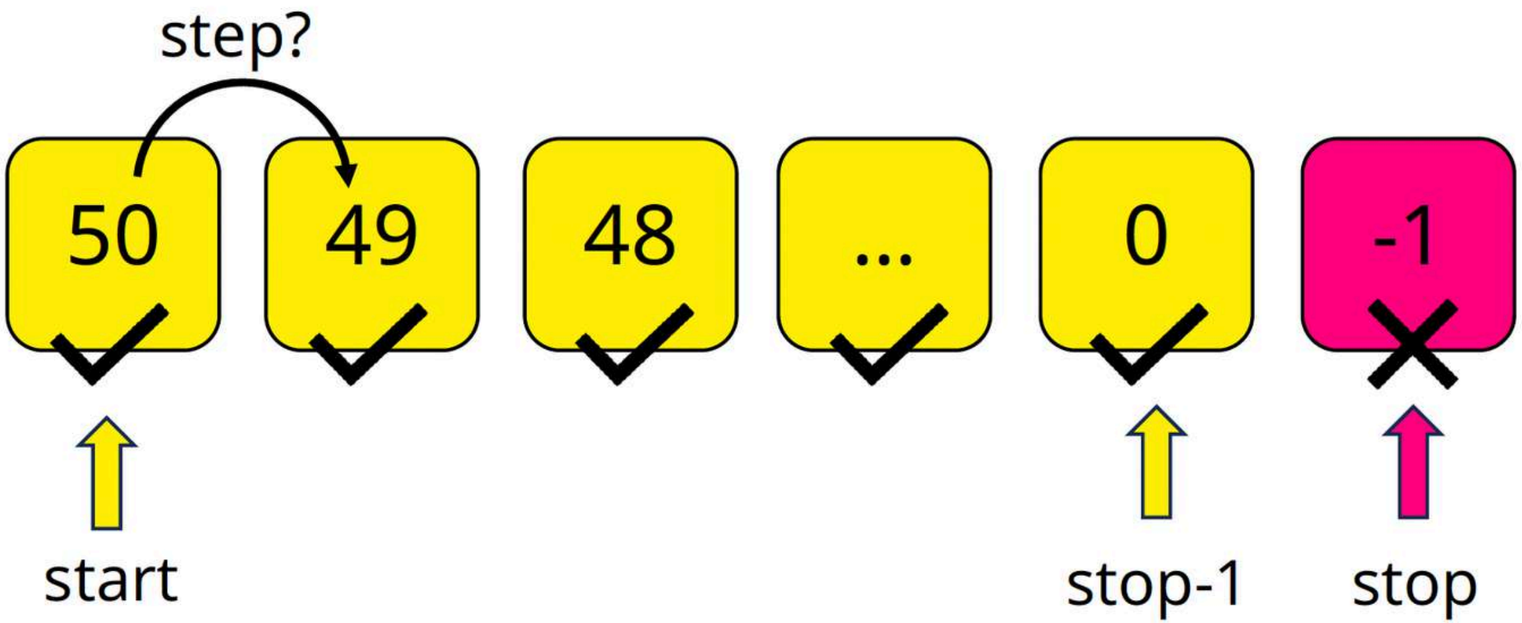
```
for i in range(_____,_____,_____):
  print(i)
```



# Iteration Statement – for

7. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 50, 49, 48, ... 0

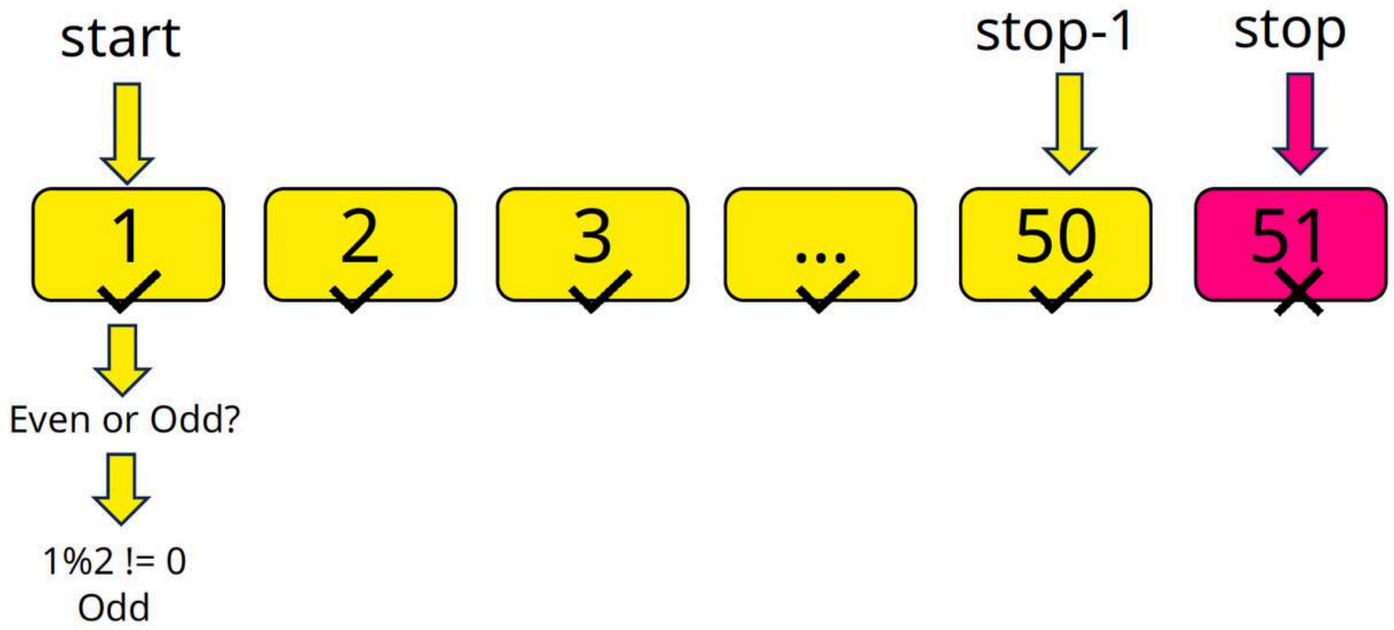
```
for i in range(_____,_____,_____):
  print(i)
```



# Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

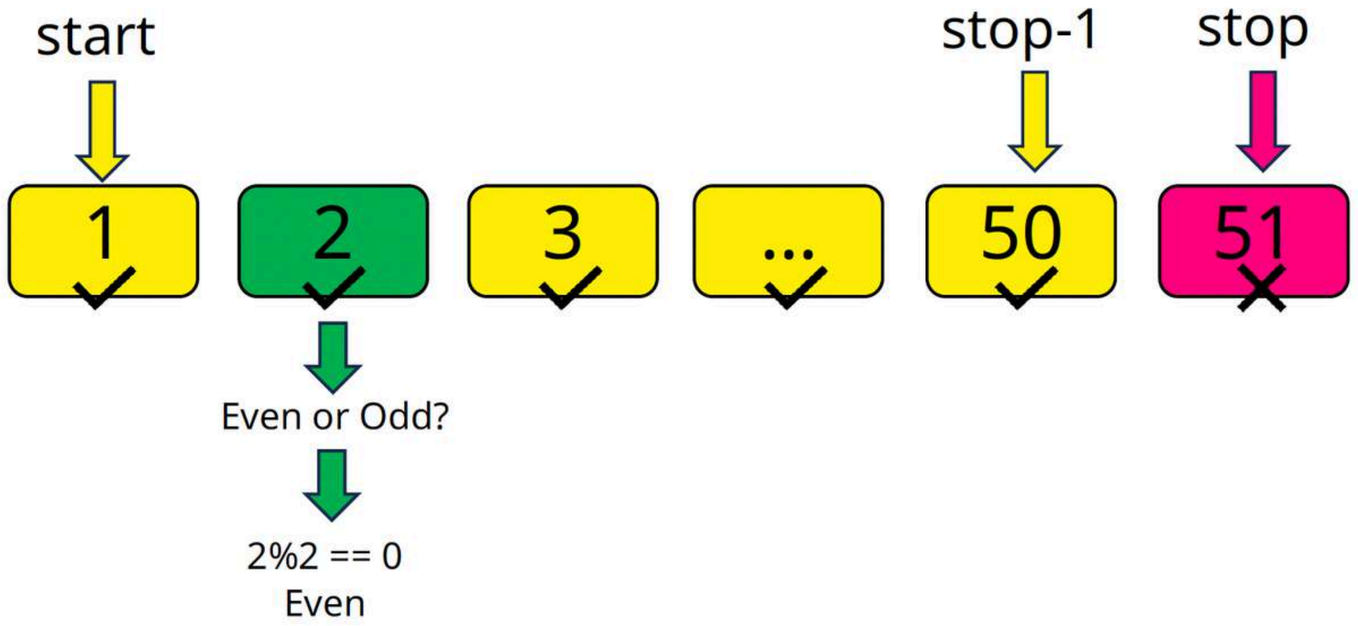
```
for i in range(________,________):
  if ___________:
    print(i)
```



# Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

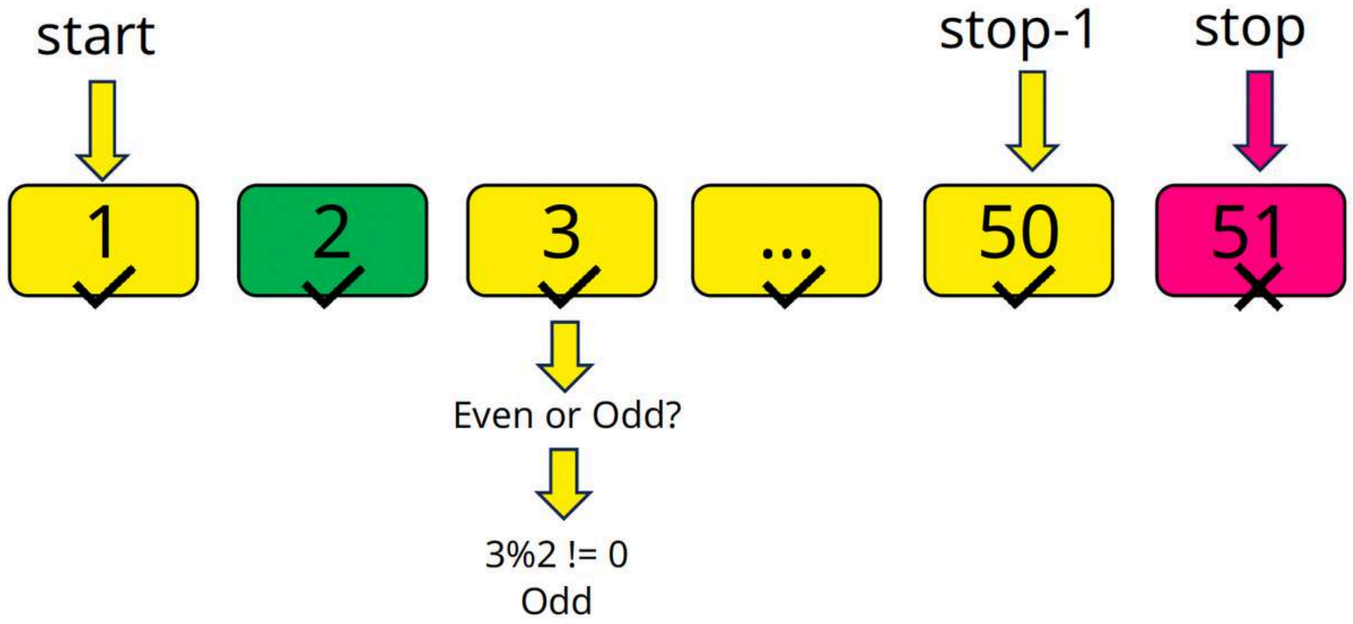
```
for i in range(________,________):
  if ___________:
    print(i)
```



# Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

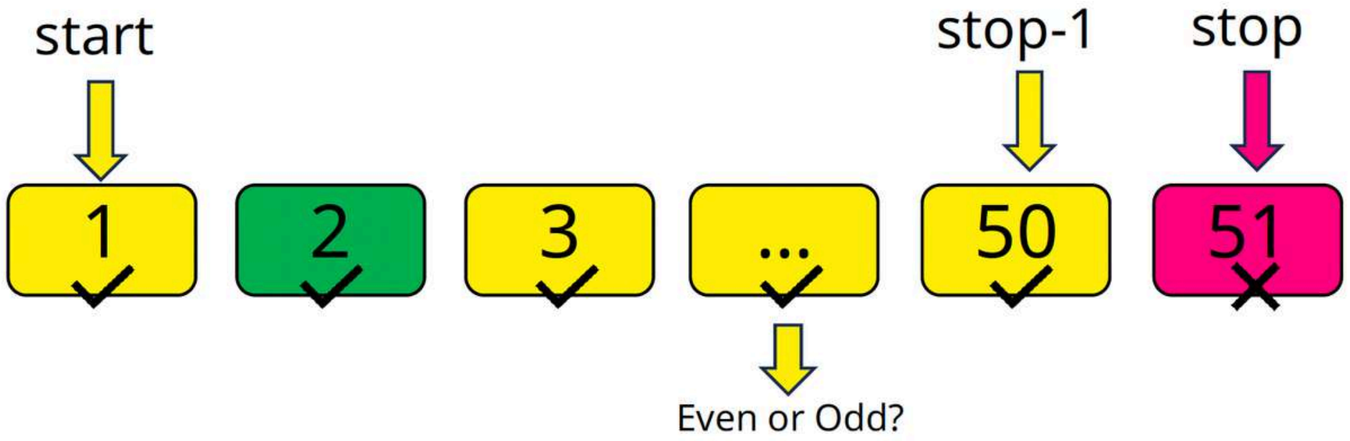
```
for i in range(________,________):
  if ___________:
    print(i)
```



# Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

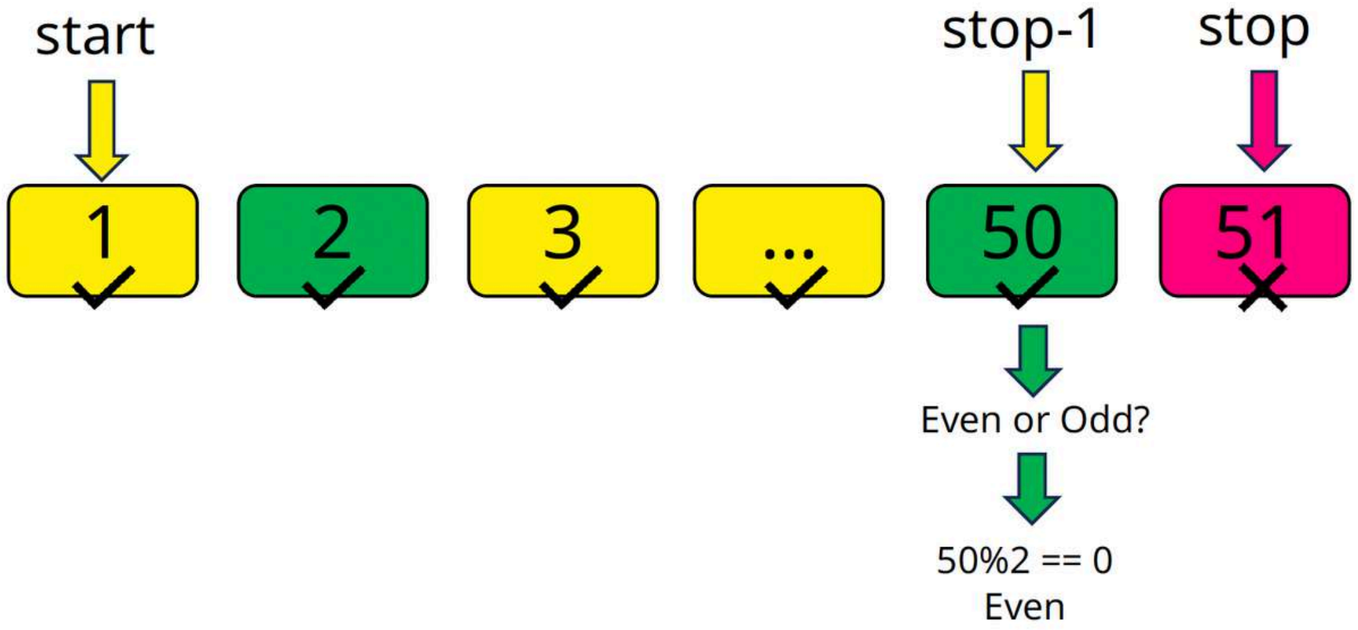
```
for i in range(________,________):
  if ___________:
    print(i)
```



# Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

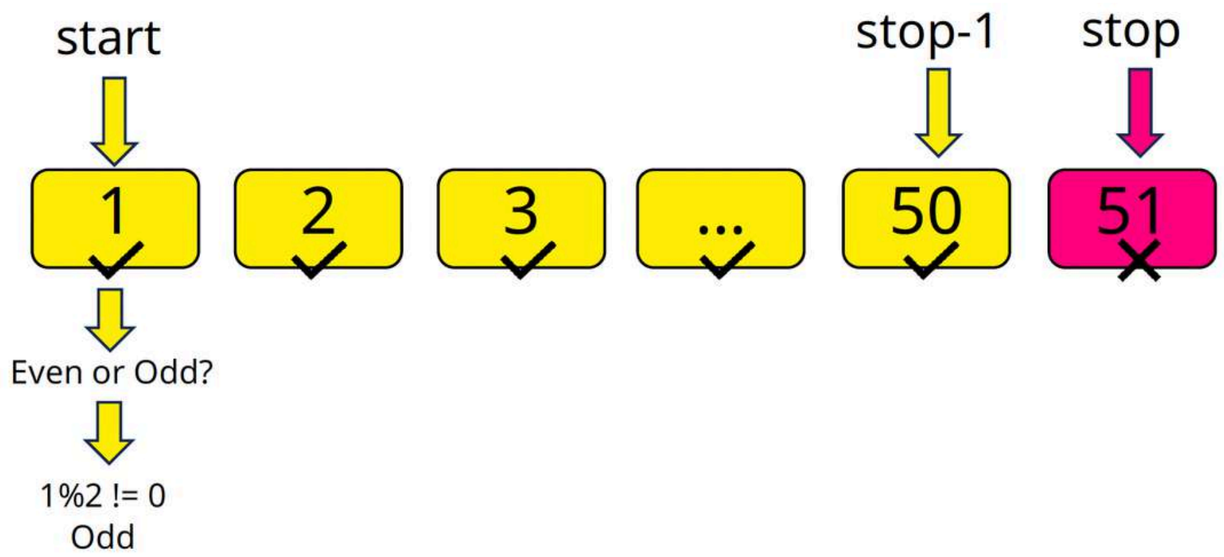
```
for i in range(________,________):
  if ___________:
    print(i)
```



# Iteration Statement – for

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

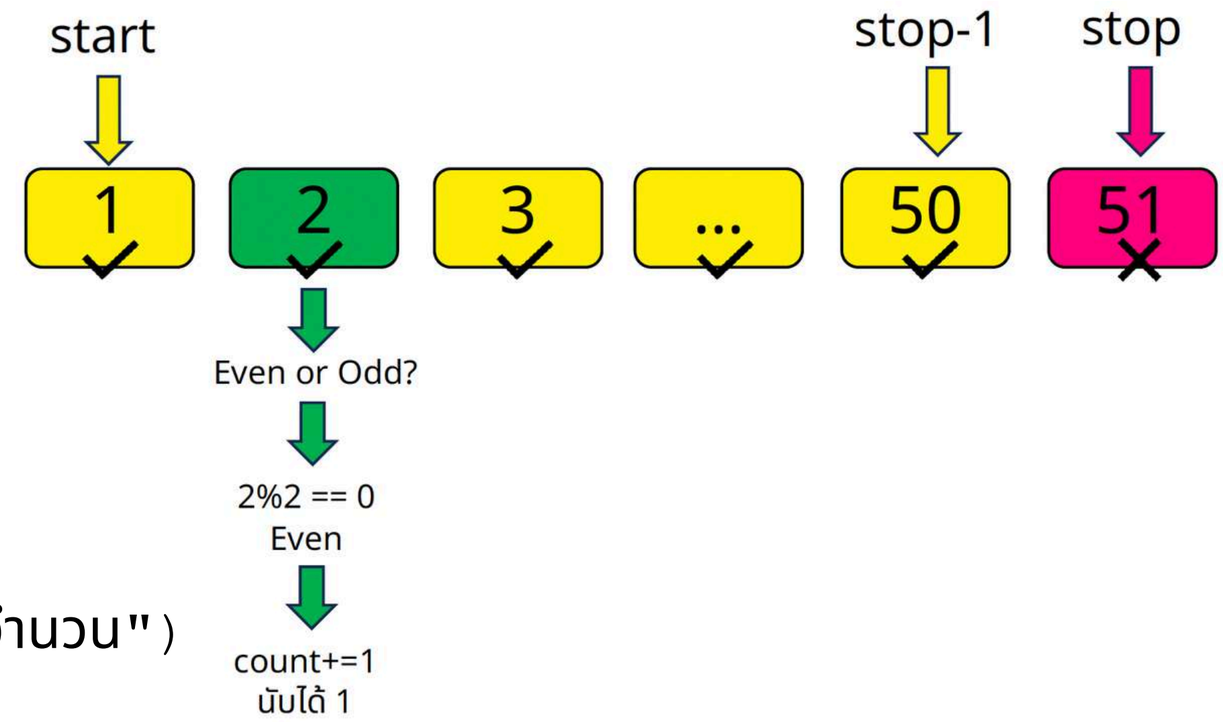
```
_______ = _______
for i in range(1,50+1):
  if i%2==0:
    print(i)
    _______ = _______
print(f"มีจำนวนคู่ทั้งหมด: {count} จำนวน")
```



# Iteration Statement – for

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

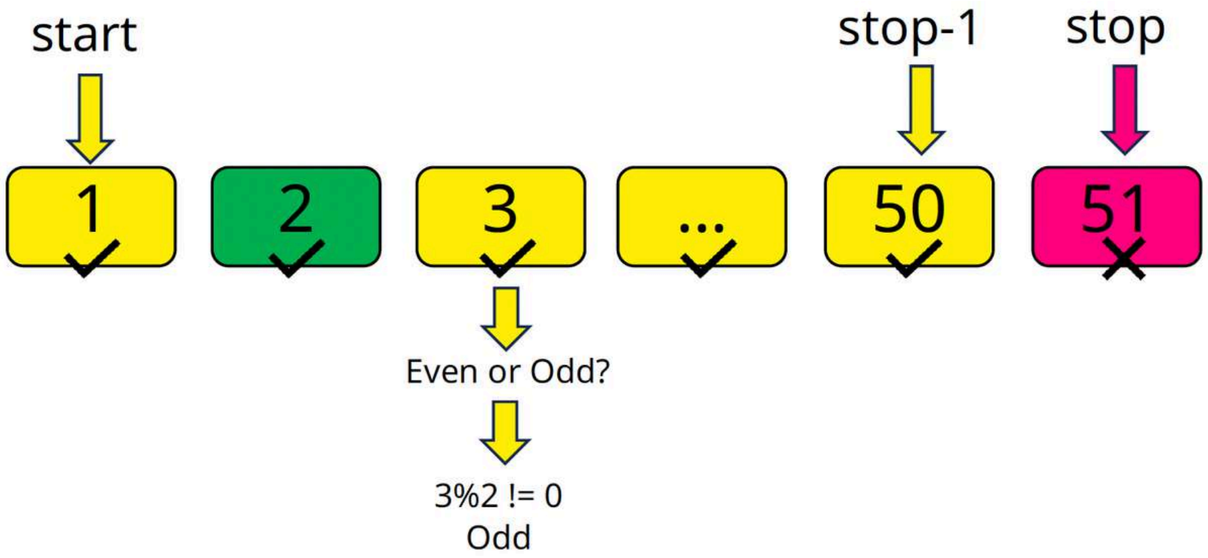
```
_______ = _______
for i in range(1,50+1):
  if i%2==0:
    print(i)
    _______ = _______

print(f"มีจำนวนคู่ทั้งหมด: {count} จำนวน")
```



# Iteration Statement – for

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

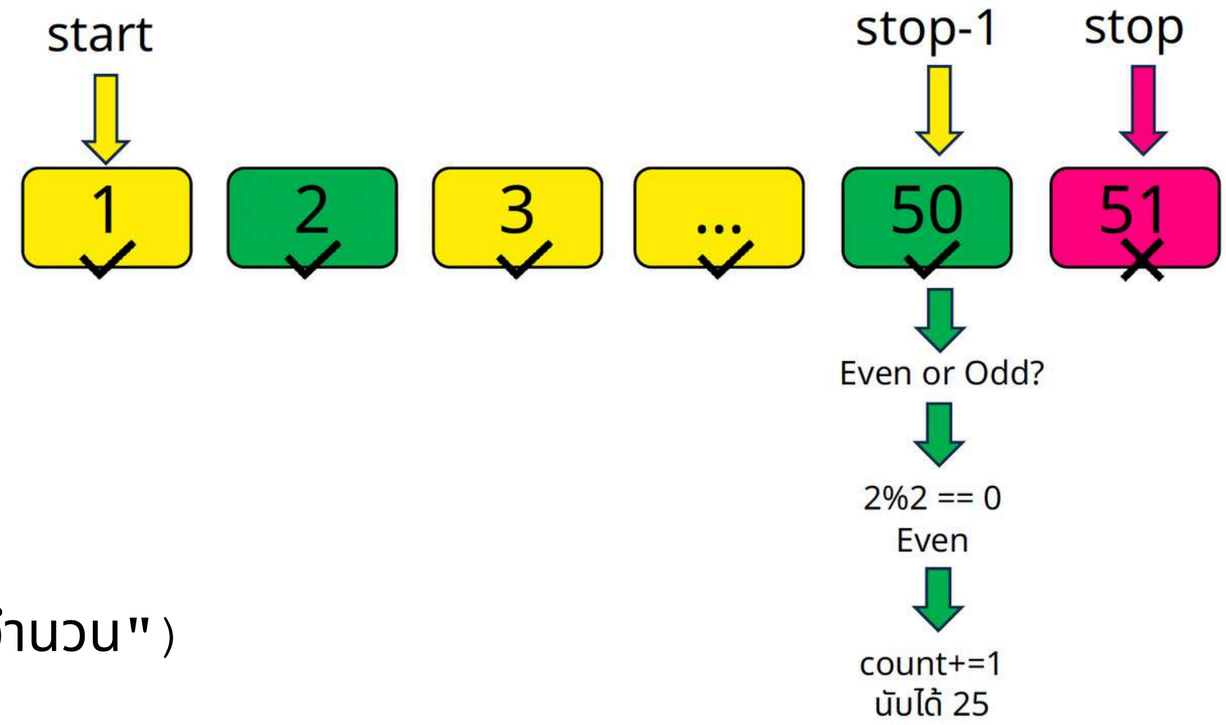
```
_______ = _______
for i in range(1,50+1):
  if i%2==0:
    print(i)
    _______ = _______

print(f"มีจำนวนคู่ทั้งหมด: {count} จำนวน")
```



# Iteration Statement – for

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

```
_______ = _______
for i in range(1,50+1):
  if i%2==0:
    print(i)
    _______ = _______

print(f"มีจำนวนคู่ทั้งหมด: {count} จำนวน")
```



# Iteration Statement – for

10. การโปรแกรมเพื่อหาผลรวมของจำนวนตั้งแต่ 1, 2, 3, ... ถึง 10

```
summation = 0
for i in range(1,10+1):
  summation = summation+i

print(f"Summation of 1 to 10 : {summation}")
```

```
Summation of 1 to 10 : 55
```

# Iteration Statement – for

11. การโปรแกรมเพื่อหาผลรวมของจำนวนตั้งแต่ 2,4,6,..., 18, 20

```
summation = _____
for i in range(_______, _________, _______):
  summation = _____________

print(f"Summation of 2,4,6,...,18, 20 : {summation}")
```

```
Summation of 2,4,6, ... ,18, 20 : 110
```

# Iteration Statement – for

11. การโปรแกรมเพื่อหาผลรวมของจำนวนตั้งแต่ 2,4,6,..., 18, 20

```
summation = 0
for i in range(2, 20+1, 2):
  summation = summation+i

print(f"Summation of 2,4,6,...,18, 20 : {summation}")
```

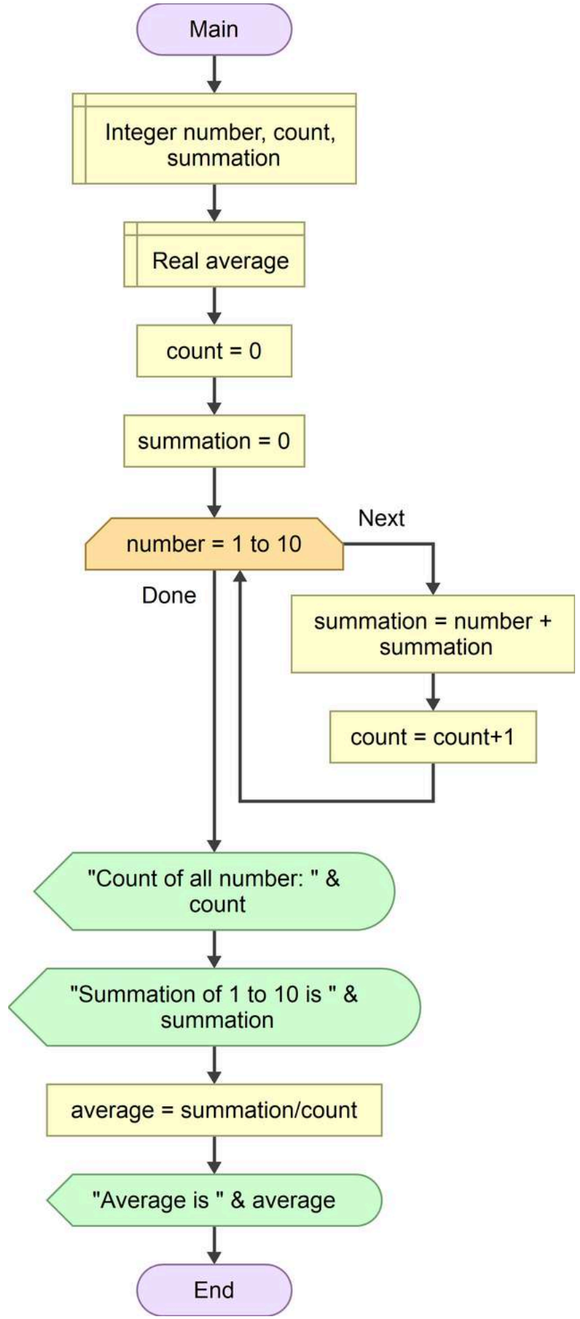
```
Summation of 2,4,6, ... ,18, 20 : 110
```

# Iteration Statement – for

12. การโปรแกรมเพื่อหาค่าเฉลี่ย (average) ของจำนวนตั้งแต่ 1 ถึง 10 โดยค่าเฉลี่ยมาจาก ผลรวมของจำนวนทั้งหมด หารด้วย จำนวนทั้งหมด หรือ

**average = summation/count**

```
summation = _____
_______ = _____
_______ = _____
for i in range(___,_____):
  summation = ___________________
  count = ___________________

average = ___________________
print(f"Average 1 to 10 : {___________}")
```

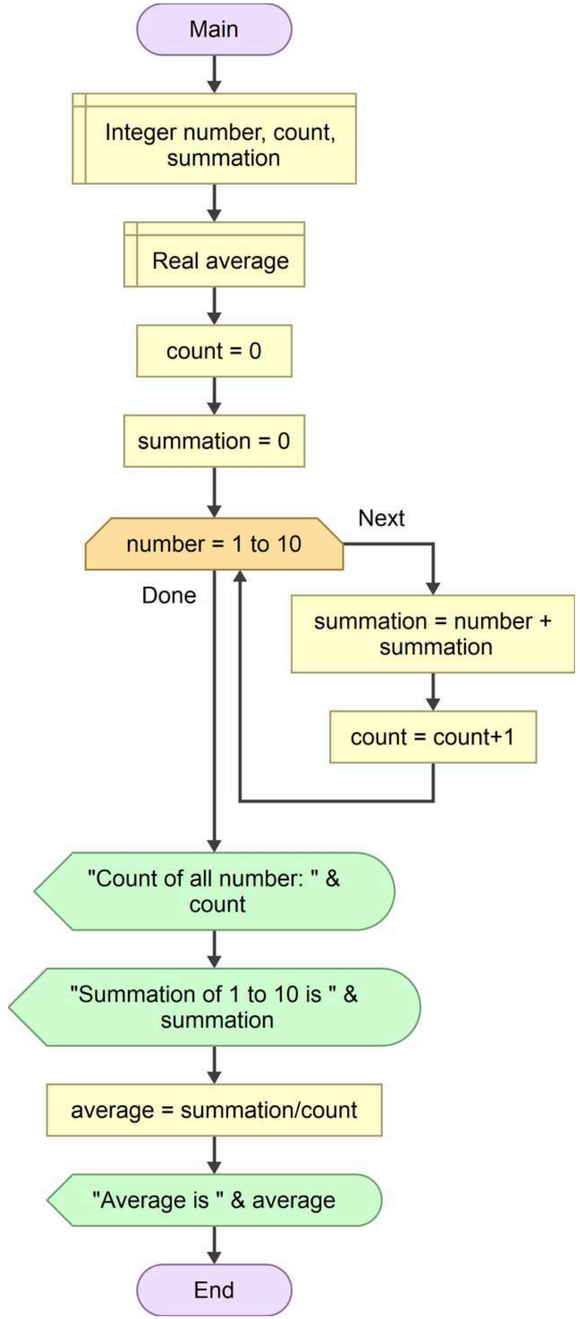


# Iteration Statement – for

12. การโปรแกรมเพื่อหาค่าเฉลี่ย (average) ของจำนวนตั้งแต่ 1 ถึง 10 โดยค่าเฉลี่ยมาจาก ผลรวมของจำนวนทั้งหมด หารด้วย จำนวนทั้งหมด หรือ

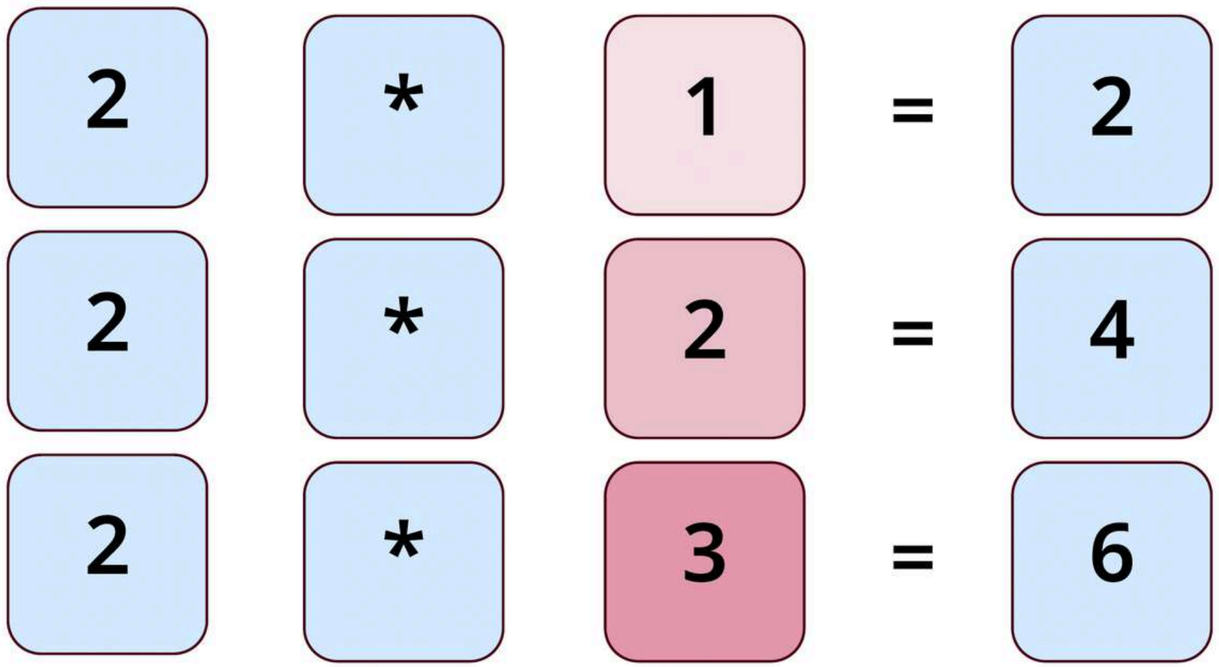
**average = summation/count**

```
summation = 0
count = 0
average = 0
for i in range(1,10+1):
    summation = summation+i
    count = count+1

average = summation/count
print(f"Average 1 to 10 : {average}")
```



# Iteration Statement – for

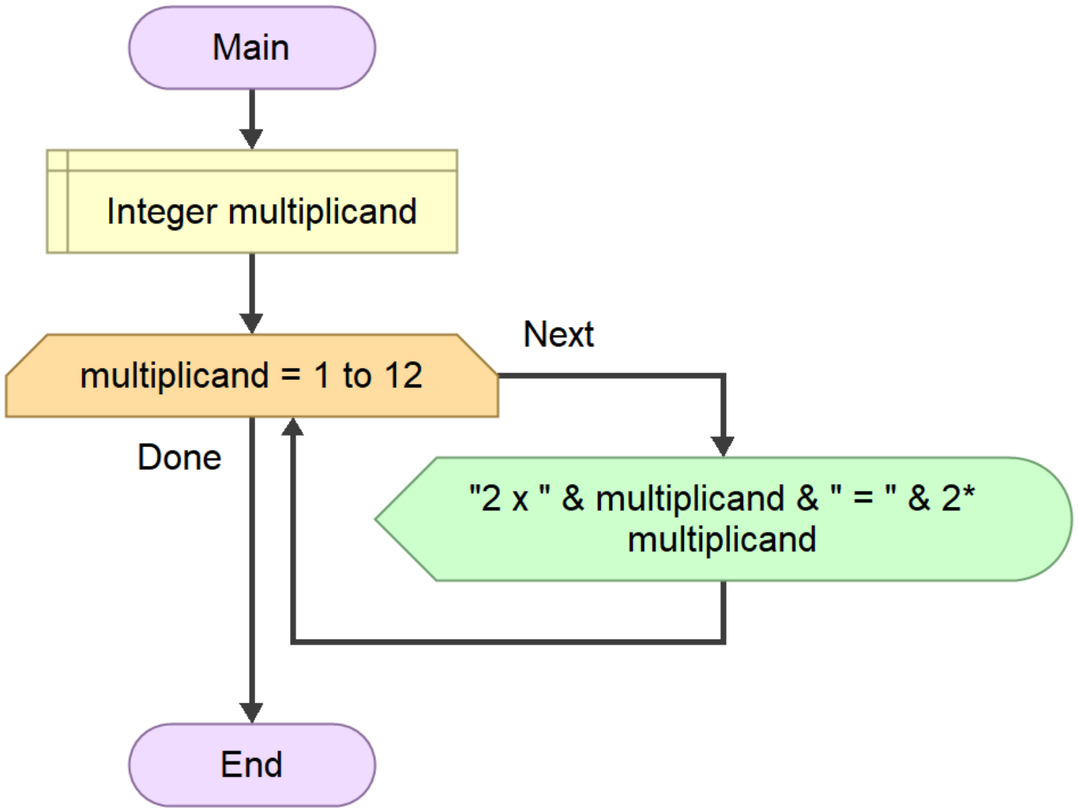
13. แสดงสูตรคูณแม่ 2 โดยเริ่มตั้งแต่ 2x1=2 ถึง 2x12=24 ด้วย for loop




```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
2 x 11 = 22
2 x 12 = 24
```

# Iteration Statement – for

13. แสดงสูตรคูณแม่ 2 โดยเริ่มตั้งแต่ 2x1=2 ถึง 2x12=24 ด้วย for loop




```
for i in range(1,12+1):
  print(f"2 x {i} = {2*i}")
```

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
2 x 11 = 22
2 x 12 = 24
```

# Iteration Statement – for

สูตรคูณ แม่ 2 ถึงแม่ 5

```
for multiplier in range(2,5+1):
    print(f"สูตรคูณแม่ {multiplier}")
    for multiplicand in range(1,12+1):
        print(f"{multiplier}x{multiplicand}={multiplier*multiplicand}")
print("-----------------")
```

```
สูตรคูณแม่ 2
2x1=2
2×2=4
2×3=6
2×4=8
2×5=10
2×6=12
2×7=14
2×8=16
2×9=18
2×10=20
2×11=22
2×12=24
_________________
```

```
สูตรคูณแม่ 3
3x1=3
3×2=6
3×3=9
3×4=12
3×5=15
3×6=18
3×7=21
3×8=24
3×9=27
3×10=30
3×11=33
3×12=36
_________________
```

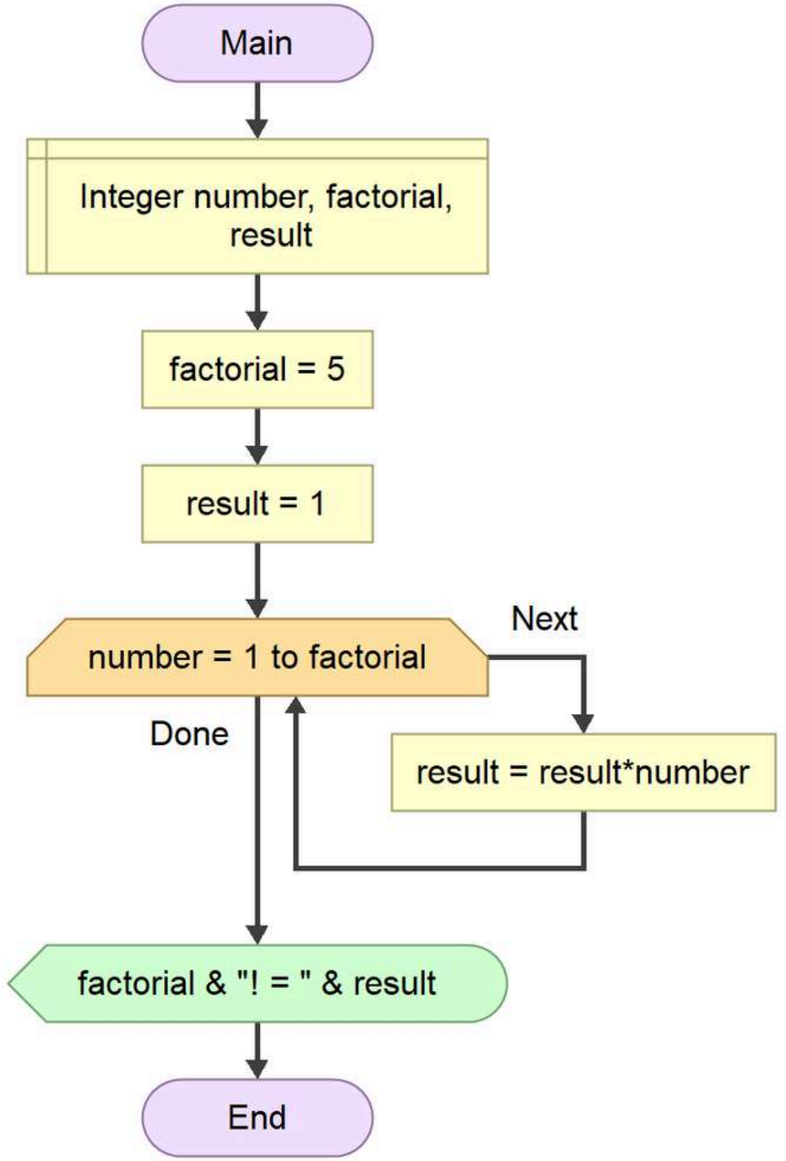
```
สูตรคูณแม่ 4
4x1=4
4×2=8
4×3=12
4×4=16
4×5=20
4×6=24
4×7=28
4×8=32
4×9=36
4×10=40
4×11=44
4×12=48
_________________
```

```
สูตรคูณแม่ 5
5x1=5
5×2=10
5×3=15
5×4=20
5×5=25
5×6=30
5×7=35
5×8=40
5×9=45
5×10=50
5×11=55
5×12=60
_________________
```

# Iteration Statement – for

14. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5!
Factorial (n!) คือ ผลคูณของจำนวนเต็มบวกตั้งแต่ 1 ถึง n เช่น

- 1! = 1
- 2! = 2 x 1 = 2
- 3! = 3 x 2 x 1 = 6
- 4! = 4 x 3 x 2 x 1 = 24
- 5! = 5 x 4 x 3 x 2 x 1 = 120



# Iteration Statement – for

14. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5!
Factorial (n!) คือ ผลคูณของจำนวนเต็มบวกตั้งแต่ 1 ถึง n
เช่น

- 1! = 1
- 2! = 2 x 1 = 2
- 3! = 3 x 2 x 1 = 6
- 4! = 4 x 3 x 2 x 1 = 24
- 5! = 5 x 4 x 3 x 2 x 1 = 120

```
result = 1
for i in range(5,0,-1):
  result = result*i

print(f"5! = {result}")
```

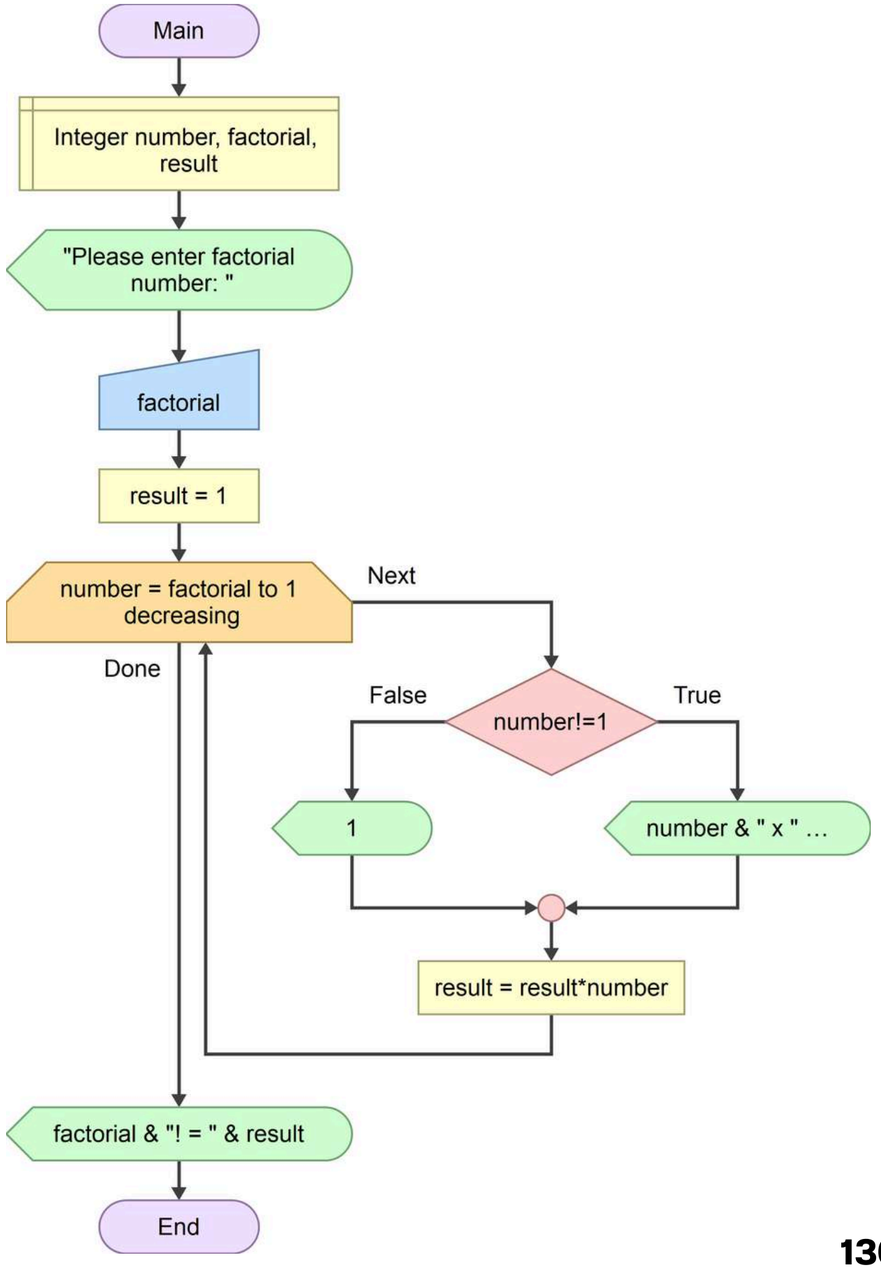
```
5! = 120
```

# Iteration Statement – for

15. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5!

**โดยแสดงเป็น**

**5! = 5 x 4 x 3 x 2 x 1 = 120**



# Iteration Statement – for

15. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5!

**โดยแสดงเป็น**

**5! = 5 x 4 x 3 x 2 x 1 = 120**

```
result = _______
print("5! = ", end="")
for i in range(____,____,____):
  if i!=1:
    print(f"{_____}x", end="")
  else:
    print(”_____", end="")
  result = _______________

print(f" = {_______}")
```

# Iteration Statement – for

15. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5!

**โดยแสดงเป็น**

**5! = 5 x 4 x 3 x 2 x 1 = 120**

```
result = 1
print("5! = ", end="")
for i in range(5,0,-1):
  if i!=1:
    print(f"{i}x", end="")
  else:
    print("1", end="")
  result = result*i

print(f" = {result}")
```

# Iteration Statement – for

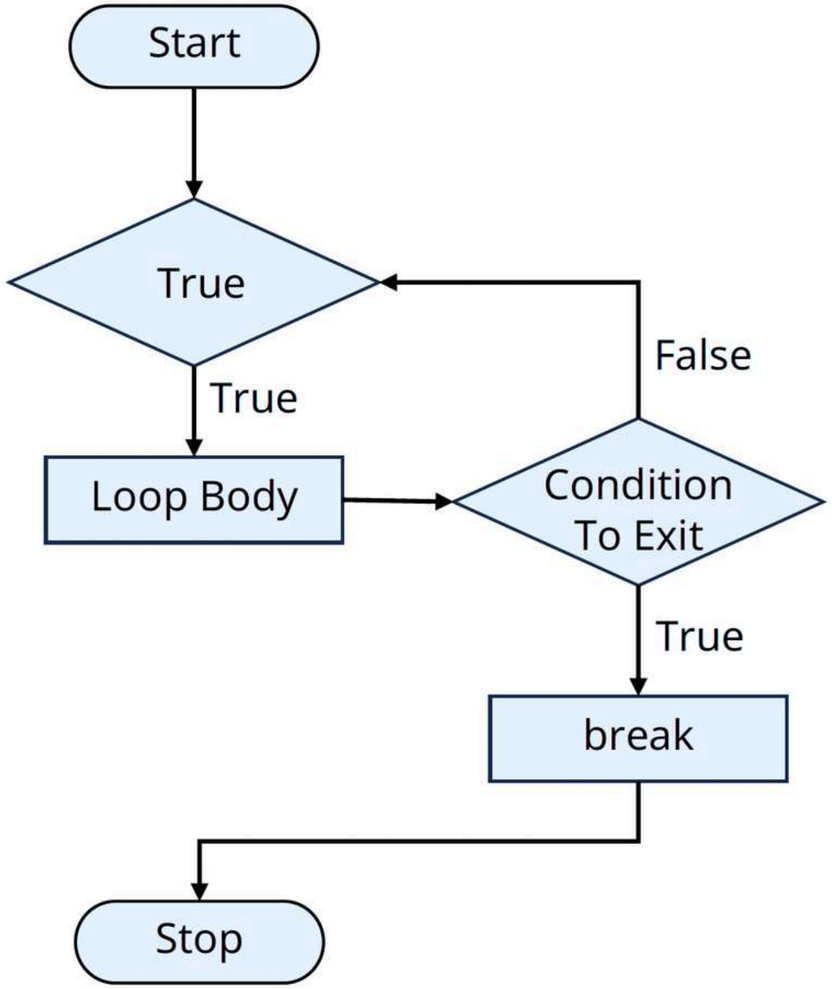
15. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5! **โดยแสดงเป็น 5! = 5 x 4 x 3 x 2 x 1 = 120**

```
# กำหนดค่าเริ่มต้นของตัวแปร result เพื่อเก็บผลคูณของตัวเลข
result = 1
# แสดงข้อความเริ่มต้นของการแสดงผลลัพธ์
print("5! = ", end="")
# ใช้ลูป for เพื่อคำนวณ Factorial ของ 5 และแสดงผลลัพธ์
for i in range(5, 0, -1):
    if i != 1:
        # แสดงตัวเลขและสัญลักษณ์คูณถ้าไม่ใช่ตัวสุดท้าย
        print(f"{i}x", end="")
    else:
        # แสดงตัวสุดท้ายโดยไม่มีสัญลักษณ์คูณ
        print("1", end="")
    result = result * i  # คำนวณผลคูณของตัวเลขทุกตัว
# แสดงผลลัพธ์ของ Factorial
print(f" = {result}")
```

# Iteration Statement – do-while

ในภาษา Python ไม่มี Syntax do-while เหมือนกับภาษาอื่น ๆ เช่น ภาษา C, C++, Java แต่ก็ยังสามารถประยุกต์ใช้ while loop ในการสร้าง do-while ได้เช่นกัน ดังนี้

```
while True:
        if condition_to_exit:
            break   # Exit the loop when the condition is match
```



# Iteration Statement – do-while

ต้องการรับรหัสผ่านเข้าประตู (Digital Door Lock) กรณีกรอกผิดให้ทำการถามใหม่วนซ้ำไปจนกว่าจะตอบถูก กรณีตอบถูกขึ้นคำว่า “Correct”

```
while True:
    userinput = input("Please input password: ")
    if userinput=="1234":
        print("Correct")
        break
```

```
Please input password: 2233333
Please input password: 22211
Please input password: 4456
Please input password: 65432
Please input password: 199[
Please input password: 1234
Correct
```

# # Assignment 7.2 - มอบหมายงานครั้งที่ 7.2

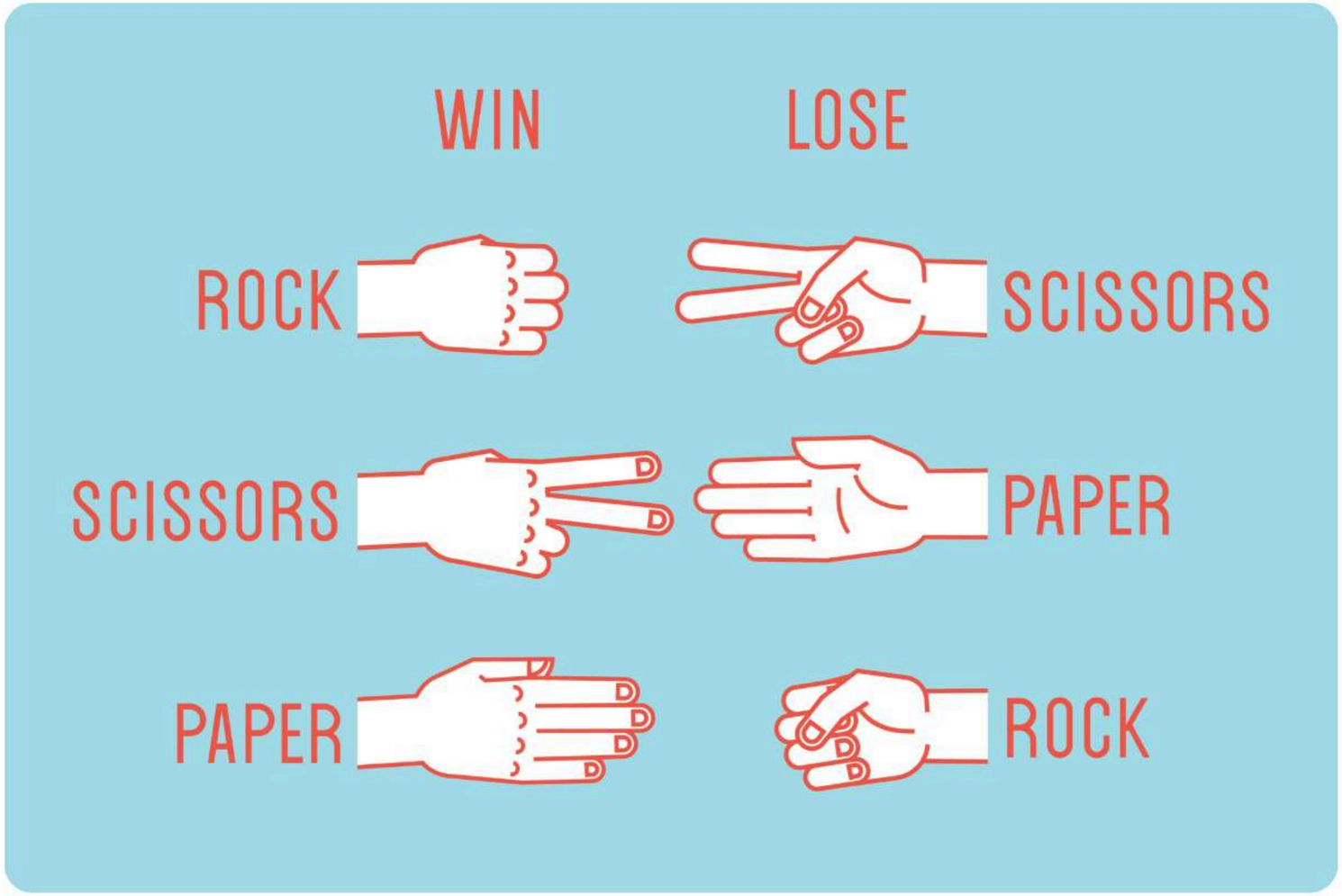
ให้เขียนโปรแกรมเพื่อเป่ายิงฉุบ กับ โปรแกรม โดยมีขั้นตอนดังนี้

1. ให้รับค่า input จากมนุษย์
   - 1 คือ Paper
   - 2 คือ Rock
   - 3 คือ Scissor
2. ให้โปรแกรมสุ่มตัวเลข 1 ถึง 3 มาต่อสู้
   - 1 หมายถึง Paper
   - 2 หมายถึง Rock
   - 3 หมายถึง Scissor

```
Rock-Paper-Scissors by Pasawut
1 = Rock
2 = Scissors
3 = Paper
Your turn:
```

```
Rock-Paper-Scissors by Pasawut
1 = Rock
2 = Scissors
3 = Paper
Your turn: 1
Your choice: Rock and Bot: Scissors
 You Win!
```

# # Assignment 7.2 - มอบหมายงานครั้งที่ 7.2



# # Assignment 7.2 - มอบหมายงานครั้งที่ 7.2

```python
import random
print( "Rock-Paper-Scissors by Pasawut")
print("1 = Rock\n2 = Scissors\n3 = Paper")
user = int(input("Your turn: "))
bot = random.randrange(1,3)
if(user == 1 and bot == 2):
 print("Your choice: Rock and Bot: Scissors\n You Win!")
elif(user == 2 and bot == 3):
 print("Your choice: Scissors and Bot: Paper\n You Win!")
elif(user == 3 and bot == 1):
 print("Your choice: Paper and Bot: Rock\n You Win!")
elif(user == 2 and bot == 1):
 print("Your choice: Scissors and Bot: Rock\n You Lose!")
elif(user == 3 and bot == 2):
 print("Your choice: Paper and Bot: Scissors\n You Lose!")
elif(user == 1 and bot == 3):
 print("Your choice: Rock and Bot: Paper\n You Lose!")
else:
 print("Draw!")
```

```
Rock-Paper-Scissors by Pasawut
1 = Rock
2 = Scissors
3 = Paper
Your turn: 1
Your choice: Rock and Bot: Scissors
 You Win!
```

# Post Test

**Question:**

# 1. ต้องการแสดงผลคำว่า Hello, World ต้องเติมคำสั่งใด

**A** Print(“Hello, World”)

**B** Print(Hello, World)

**C** print(“Hello, World”)

**D** print(Hello, World)

**E** ถูกทุกข้อ

**F** ไม่มีข้อถูก

**Question:**

## 2. ข้อใดไม่ใช่ Reserved Word

**A** True

**B** None

**C** break

**D** class

**E** Control

**Question:**

# 3. ข้อใดให้ผลลัพธ์ Date:11/DEC/2023

**A**

```
print('Date:')
print('11', 'DEC','2023', sep='/')
```

**B**

```
print('Date:', end='/n')
print('11'/ 'DEC'/'2023', sep='/')
```

**C**

```
print('Date:')
print(‘11/', ‘DEC/','2023')
```

**D**

```
print('Date:', end='')
print('11', 'DEC', '2023', sep='/')
```

**Question:**

# 4. ข้อใด คือ การตั้งชื่อตัวแปรแบบ Pascal Case

**A** My_Product_Price

**B** MyProductPrice

**C** myProductPrice

**D** my_product_price

**E** MYPRODUCTPRICE

**Question:**

## 5. จาก Code แสดงผลตามข้อใด

```
1.  data1 = ”50”
2.  data1 = float(data1)
3.  print(data1, 'is', type(data1))
```

**A** 50.0 is <class 'float'>

**B** 50 is <class 'float'>

**C** “50” is <class ‘str'>

**D** “50.0” is <class 'float'>

**Question:**

## 6. จาก Code ข้อใดให้ค่า Boolean เป็น True

```
1.  student1 = 50
2.  student2 = 100
```

**A** print(student1>student2)

**B** print(student1=student2)

**C** print(student1<student2)

**D** print(student1!student2)

**Question:**

## 7. จาก Code ประเภทข้อมูลที่รับเข้ามาเป็นประเภทใด

```
1.  year = input(“Birth Year:”)
Birth Year: 1983
```

**A** int

**C** string

**B** float

**D** real

**Question:**

## 8. ข้อใดไม่ใช่ Keywords ใน Conditional Statement ในภาษา Python

**A** if

**B** if elif

**C** else

**D** if else

**E** if elif else

**F** if elseif else

**Question:**

## 9. จากคำสั่ง ไม่สามารถ แสดงผลข้อใด

```
import random
num = random.randrange(5,10)
print(num)
```

**A** 5

**B** 7

**C** 10

**D** 9

**E** 8

**F** 6

**Question:**

## 10. จากคำสั่ง ไม่สามารถ แสดงผลข้อใด

```
1. count = 1
2. while count<=5:
3.    print(count)
4.    count+=1
```

**A** 1

**B** 2

**C** 3

**D** 4

**E** 5

**F** 6

Computational Science

# Thank You

**Pasawut Cheerapakorn**
Suan Sunandha Rajabhat University

**DTI1306 Computational Science**
Department of Digital Technology for Education
Faculty of Education, Suan Sunandha Rajabhat University

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

# โจทย์สำหรับชิ้นงานกลุ่ม 20 คะแนน

**ให้นักศึกษาเลือกปัญหาในชีวิตประจำวัน 1 เรื่อง เช่น**

- ระบบสั่งอาหารในโรงอาหาร
- การจัดตารางเรียน
- การยืม-คืนหนังสือในห้องสมุด
- การจองห้องประชุม

**ผลงานที่ส่ง**

- สไลด์นำเสนอ 5 หน้าขึ้นไป
- แผนผัง Flowchart หรือ Pseudocode
- นำเสนอหน้าชั้นเรียน 5-10 นาที

**จากนั้นให้แต่ละกลุ่มดำเนินการดังนี้**

- วิเคราะห์ปัญหา (Problem Analysis) ตามขั้นตอนของวิทยาการคำนวณ (Decomposition, Pattern Recognition, Abstraction, Algorithm Design)
- ออกแบบขั้นตอนการทำงานเป็น Flowchart หรือ Pseudocode
- อธิบายว่าผังงานนี้ช่วยแก้ปัญหาได้อย่างไร