CS

พื้นฐานการโปรแกรมภาษาไพทอนเบื้องต้นด้วย Colab

บทที่ 7

ตอนที่ 2 โครงสร้างการโปรแกรมแบบเงื่อนไขและโครงสร้างการโปรแกรมแบบทำซ้ำเบื้องต้น

ผู้ช่วยศาสตราจารย์ ดร.ณัฐภัทร แก้วรัตนภัทร์
Asst.Prof.Dr.Nutthapat Kaewrattanapat
Suan Sunandha Rajabhat University

รายวิชาวิทยาการคำนวณ (Computational Science) 3(2-2-5) หน่วยกิต
บรรยายนักศึกษาสาขาวิชาเทคโนโลยีดิจิทัลเพื่อการศึกษา ภาคเรียนที่ 2 ปีการศึกษา 2566
ห้องบรรยาย 1121 คณะครุศาสตร์ มหาวิทยาลัยราชภัฏสวนสุนันทา

SUSTAINABLE DEVELOPMENT GOALS
4 QUALITY EDUCATION
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE

อาจารย์บรรยาย

ผู้ช่วยศาสตราจารย์ ดร.ณัฐภัทร แก้วรัตนภัทร์

การศึกษา

- 2565 ปริญญาเอก ปรัชญาดุษฎีบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการศึกษา (GPA. 4.00)
- 2551 ปริญญาโท วิทยาศาสตรมหาบัณฑิต สาขาวิชาระบบสารสนเทศเพื่อการจัดการ (GPA. 3.58)
- 2549 ปริญญาตรี วิทยาศาสตรบัณฑิต เกียรตินิยมอันดับ 1 สาขาวิชาวิทยาการคอมพิวเตอร์ (GPA. 3.66)

คุณวุฒิวิชาชีพและประกาศนียบัตร

- คุณวุฒิวิชาชีพ จาก สถาบันคุณวุฒิวิชาชีพ (องค์การมหาชน) สาขาวิชาชีพอุตสาหกรรมดิจิทัล สาขาธุรกิจดิจิทัลและพาณิชย์อิเล็กทรอนิกส์ อาชีพนักจัดการเทคโนโลยีสารสนเทศสำหรับธุรกิจ ระดับ 6 เลขที่หนังสือรับรอง PQCN-ICT-ECM-0-251100-B-64/000029
- วิทยาศาสตร์ข้อมูลด้วยภาษาไพทอน, **มหาวิทยาลัยธรรมศาสตร์**
- การโปรแกรมสำหรับนักภาษาศาสตร์, **มหาวิทยาลัยมหิดล**
- การโปรแกรมภาษาไพทอน, **มหาวิทยาลัยเพนซิลวาเนีย, สหรัฐอเมริกา**
- การโปรแกรมสำหรับทุกคน, **มหาวิทยาลัยมิชชิแกน, สหรัฐอเมริกา**

ติดต่อ: nutthapat.ke@ssru.ac.th

Course Description

DTI1306 วิทยาการคำนวณ (Computational Science)
3(2-2-5) บรรยาย 2 ชม ปฏิบัติ 2 ชม ศึกษาด้วยตนเอง 5 ชม

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.

Reference: https://edu.ssru.ac.th/useruploads/files/20230724/1772131ed638786bc8d19918b37249af72c36be4.pdf

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

Measurement and Evaluation

การวัดและประเมินผล

1. ระหว่างการจัดการเรียนรู้

- สอบ Pre-test 0%
- การมอบหมายงาน 24%
- สอบ Post-test 12%
- การมีส่วนร่วมในชั้นเรียน 4%

2. การสอบกลางภาค (Midterm Examination)

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

3. โครงงานประจำภาคเรียน (Term Project)

- บทความ และการนำเสนอ 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

การมีส่วนร่วมในชั้นเรียน – ส่ง Lecture Note

ครั้งที่ 1 วันพุธที่ 6 ธันวาคม 2566

ชื่อ-สกุล: **รหัสนักศึกษา:**
อีเมล:

Pretest

1.	**6.**
2.	**7.**
3.	**8.**
4.	**9.**
5.	**10.**

Post-test

1.	**6.**
2.	**7.**
3.	**8.**
4.	**9.**
5.	**10.**

สรุปเนื้อหาบรรยาย

Course Outline

- บทที่ 1 – พื้นฐานทางวิทยาการคำนวณ
- บทที่ 2 – พื้นฐานทางด้านเทคโนโลยีดิจิทัล
- บทที่ 3 – พื้นฐานทางด้านการรู้เท่าทันสื่อและดิจิทัล
- บทที่ 4 – พื้นฐานการวิเคราะห์และออกแบบอัลกอริทึม
- บทที่ 5 – พื้นฐานการเขียนโปรแกรมด้วยบล็อกคำสั่ง (Scratch)
- บทที่ 6 – พื้นฐานการโปรแกรมไมโครคอนโทรลเลอร์เบื้องต้นด้วย Microbit
- **บทที่ 7 – พื้นฐานการโปรแกรมภาษาไพทอนเบื้องต้นด้วย Colab**
- บทที่ 8 – การพัฒนาโครงงานทางเทคโนโลยีดิจิทัลเพื่อการศึกษา

Measurement and Evaluation

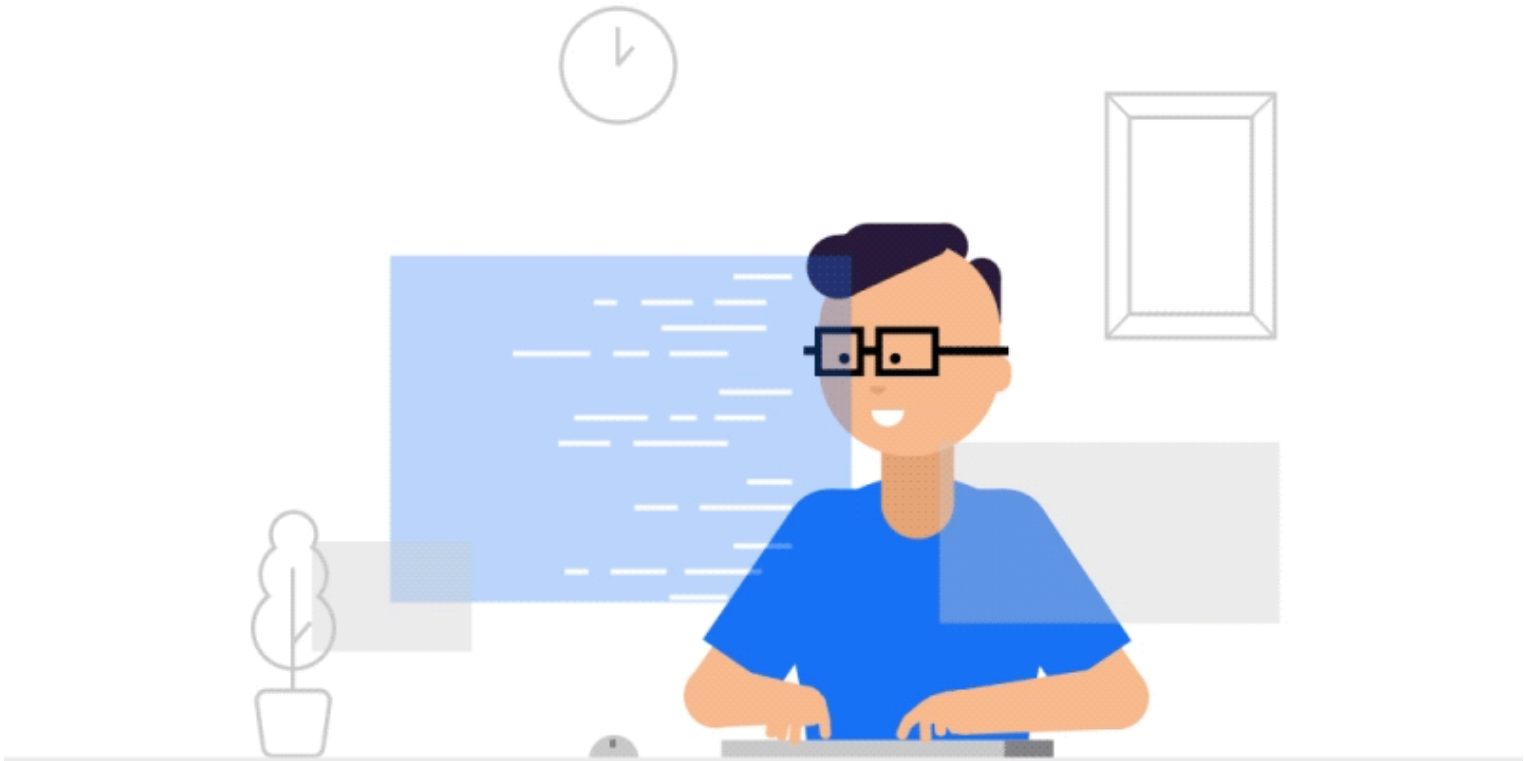
ครั้งที่/วันที่	บทเรียน/หัวข้อ	มอบหมายงาน (24%)	สอบ Post-test (12%)	การมีส่วนร่วมในชั้นเรียน (4%)
1 พุธ 6 ธค 66 Onsite	แนะนำรายวิชา การวัดและการประเมินผล หัวข้อเรียนรู้ บทที่ 1 พื้นฐานทางวิทยาการคำนวณ	-	-	ขาด/ไม่ทันเช็คชื่อ -1%
2 พุธ 13 ธค 66 Onsite	บทที่ 2 พื้นฐานทางด้านเทคโนโลยีดิจิทัล	2 คะแนน Minute Paper	2 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
3 พุธ 20 ธค 66 Onsite	บทที่ 3 พื้นฐานทางด้านการรู้เท่าทันสื่อและดิจิทัล	2 คะแนน Certificate TPQI	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
4 พุธ 27 ธค 66 Hybrid	บทที่ 4 พื้นฐานการวิเคราะห์และออกแบบอัลกอริทึม ตอนที่ 1 (Flowchart, Flowgorithm และโครงสร้างการควบคุมโปรแกรมแบบเรียงลำดับและโครงสร้างการควบคุมโปรแกรมแบบตัดสินใจ)	3 คะแนน โปรแกรม bmi	1 คะแนน สอบในระบบ	ขาด/ไม่ทันเช็คชื่อ -1%
5 พุธ 3 มค 67 Onsite	บทที่ 4 พื้นฐานการวิเคราะห์และออกแบบอัลกอริทึม ตอนที่ 2 (โครงสร้างการควบคุมโปรแกรมแบบทำซ้ำ)	3 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
6 พุธ 10 มค 67 Onsite	บทที่ 5 พื้นฐานการเขียนโปรแกรมด้วยบล็อกคำสั่ง ตอนที่ 1 (Block-based Programming ด้วย Scratch)	2 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
7 พุธ 17 มค 67 On-Demand	บทที่ 5 พื้นฐานการเขียนโปรแกรมด้วยบล็อกคำสั่ง ตอนที่ 2 (Block-based Programming และการประยุกต์)	2 คะแนน	1 คะแนน สอบในระบบ	เช็คชื่อในระบบ
8 พุธ 24 มค 67	สอบกลางภาค ปรนัย 35 ข้อ (35 คะแนน) และอัตนัย 1 ข้อ (5 คะแนน) 20%			

Measurement and Evaluation

ครั้งที่/วันที่	บทเรียน/หัวข้อ	มอบหมายงาน (24%)	สอบ Post-test (12%)	การมีส่วนร่วมในชั้นเรียน (4%)
9 พุธ 7 กพ 67 **Onsite**	บทที่ 6 การโปรแกรมไมโครคอนโทรลเลอร์เบื้องต้นด้วย Microbit	2 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
10 พุธ 14 กพ 67 **Onsite**	บทที่ 7 การโปรแกรมภาษาไพทอนเบื้องต้นด้วย Colab (Basic Input/Output, Variables, Operation)	2 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
11 พุธ 21 กพ 67 **Onsite**	บทที่ 7 การโปรแกรมภาษาไพทอนเบื้องต้นด้วย Colab (Decision and Iteration Statement)	2 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
12 พุธ 28 กพ 67 **Online**	บทที่ 8 การพัฒนาโครงงานทางเทคโนโลยีดิจิทัลเพื่อการศึกษา (การคิดเชิงออกแบบ, กระบวนการทางโครงงาน)	2 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
13 พุธ 6 มีค 67 **Online**	บทที่ 8 การพัฒนาโครงงานทางเทคโนโลยีดิจิทัลเพื่อการศึกษา (ปฏิบัติ)	2 คะแนน	1 คะแนน	ขาด/ไม่ทันเช็คชื่อ -1%
14 พุธ 13 มีค 67	สอบปลายภาค ปรนัย 35 ข้อ (35 คะแนน) และอัตนัย 1 ข้อ (5 คะแนน) 20%			
15 พุธ 20 มีค 67	ส่งบทความ (โครงงาน) และนำเสนอ 20% แผนการจัดการเรียนรู้และสื่อการเรียนรู้ด้านวิทยาการคำนวณ			

Pretest

Post-test



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

โครงสร้างการควบคุมโปรแกรม (Program Control Structures)

โครงสร้างการควบคุมโปรแกรม (Program Control Structures) ในการเขียนโปรแกรมมี 3 ลักษณะหลัก ได้แก่

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

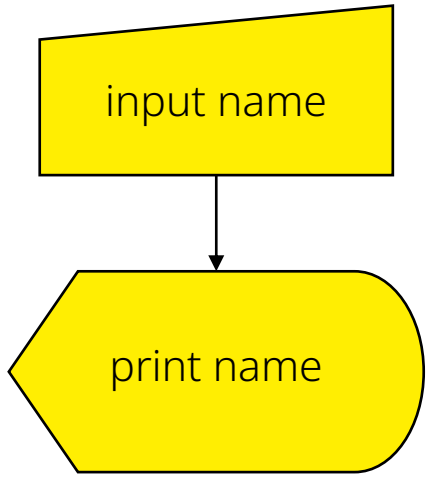
แต่ละลักษณะมีความสำคัญและทำหน้าที่ที่แตกต่างกันในการควบคุมการทำงานของโปรแกรม

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)

คำสั่งเงื่อนไขช่วยให้โปรแกรมสามารถเลือกทำงานทางเลือกต่างๆ 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 2.1
2.sun = True
3.if sun:
4.print("Morning")
```

Morning

```
1.# Code 2.2
2.sun = False
3.if sun:
4.print("Morning")
```

Conditional Statement (คำสั่งเงื่อนไข) – if

```
height = 175
weight = 62
if(height>170 and height<=180):
    print("Height: Perfect")
```

A

Height: Perfect

B

Nil

Conditional Statement (คำสั่งเงื่อนไข) – if

Set a condition to print 'You are tall.' if the height in the variable my_height is greater than 170

```
my_height = 175

if ___________ :
    print('You are tall.')
```

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.

```
my_salary = _______
my_height = _______

if ____________ :
    print('You are wealthy and tall.')
```

Conditional Statement (คำสั่งเงื่อนไข) – if

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

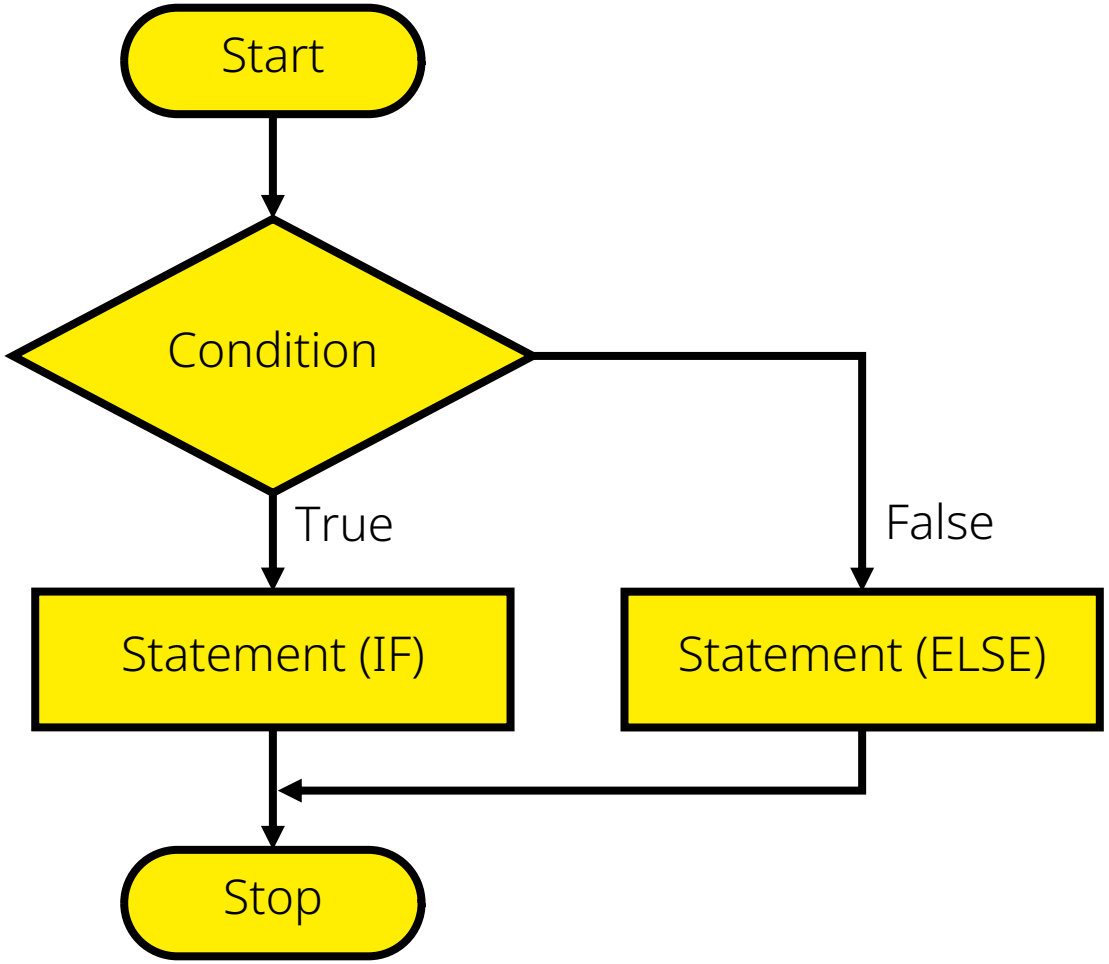
A

There is fruit in the basket

B

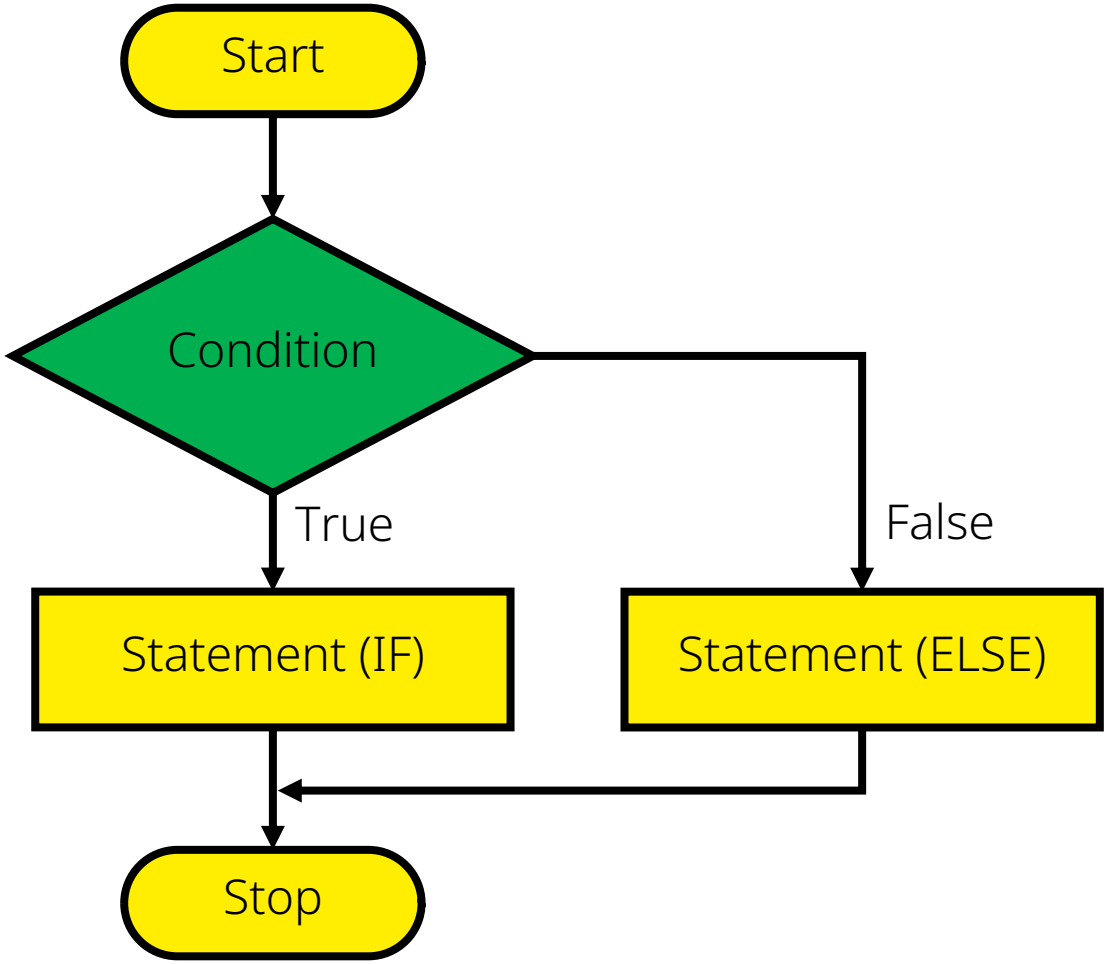
Nil

Conditional Statement (คำสั่งเงื่อนไข) – if else



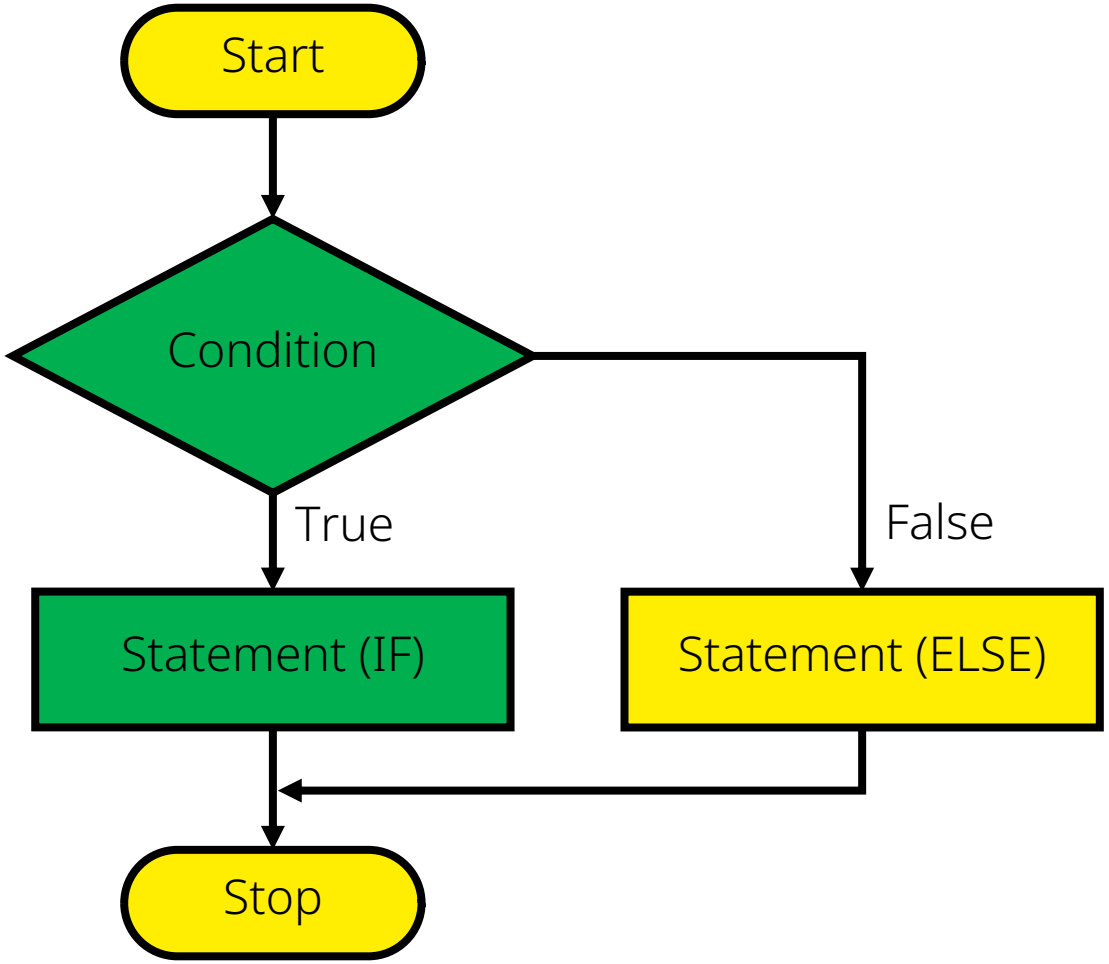

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

Conditional Statement (คำสั่งเงื่อนไข) – if else



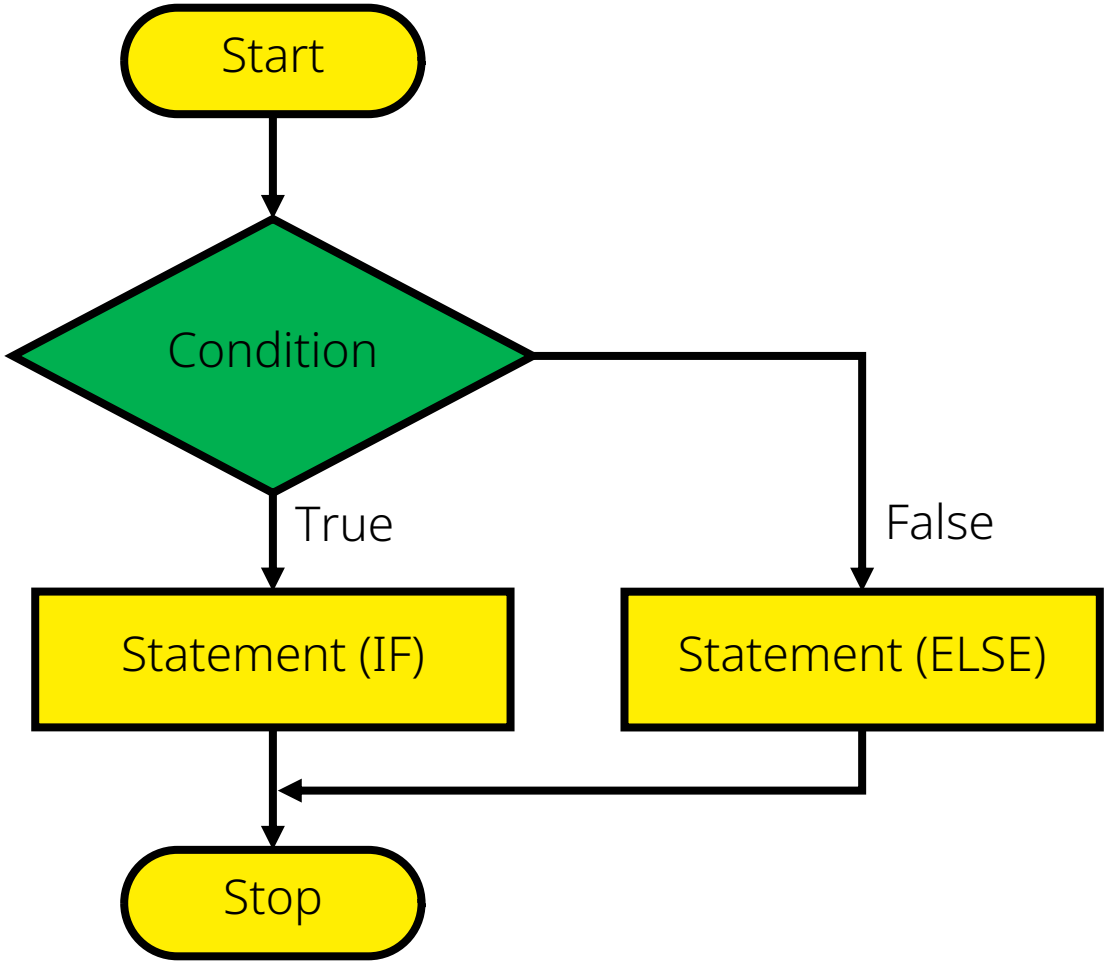

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

Conditional Statement (คำสั่งเงื่อนไข) – if else



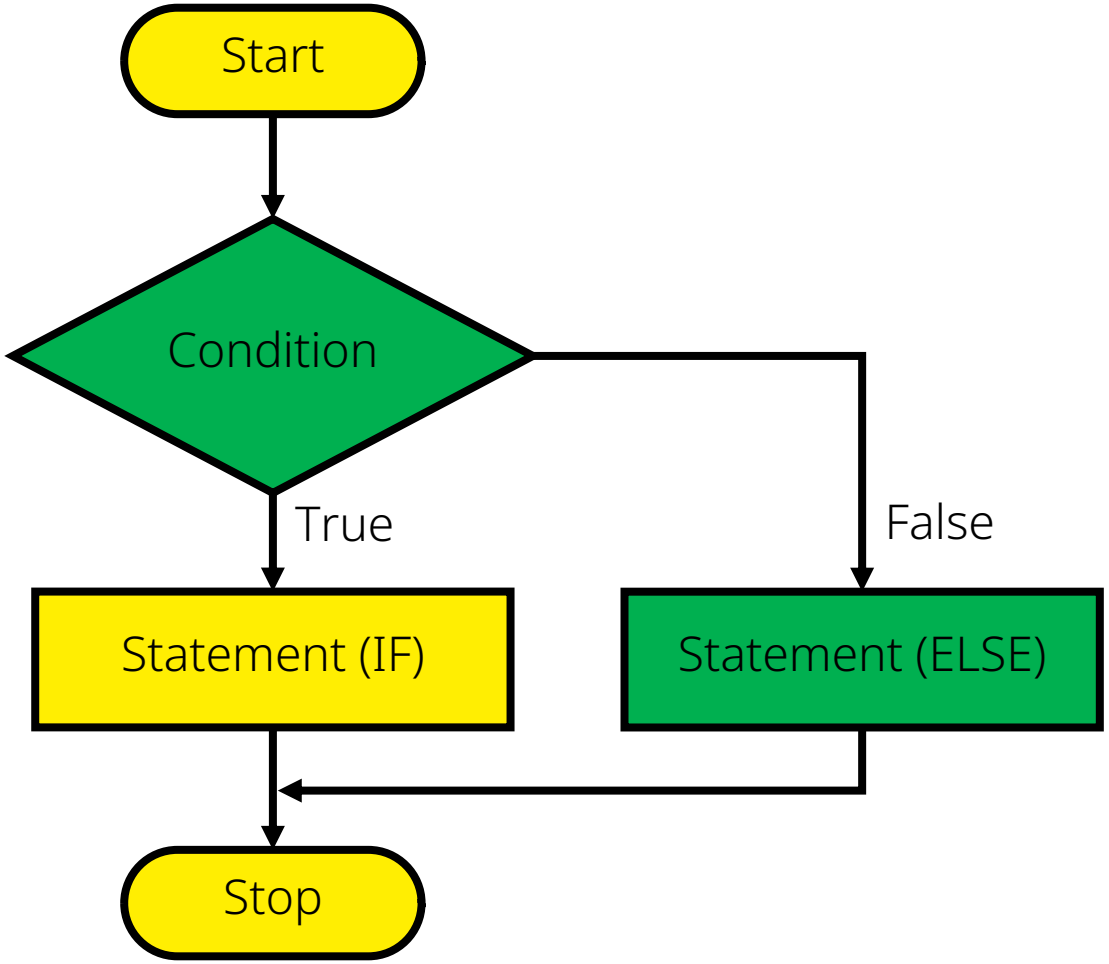

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

Conditional Statement (คำสั่งเงื่อนไข) – if else




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

Conditional Statement (คำสั่งเงื่อนไข) – if else



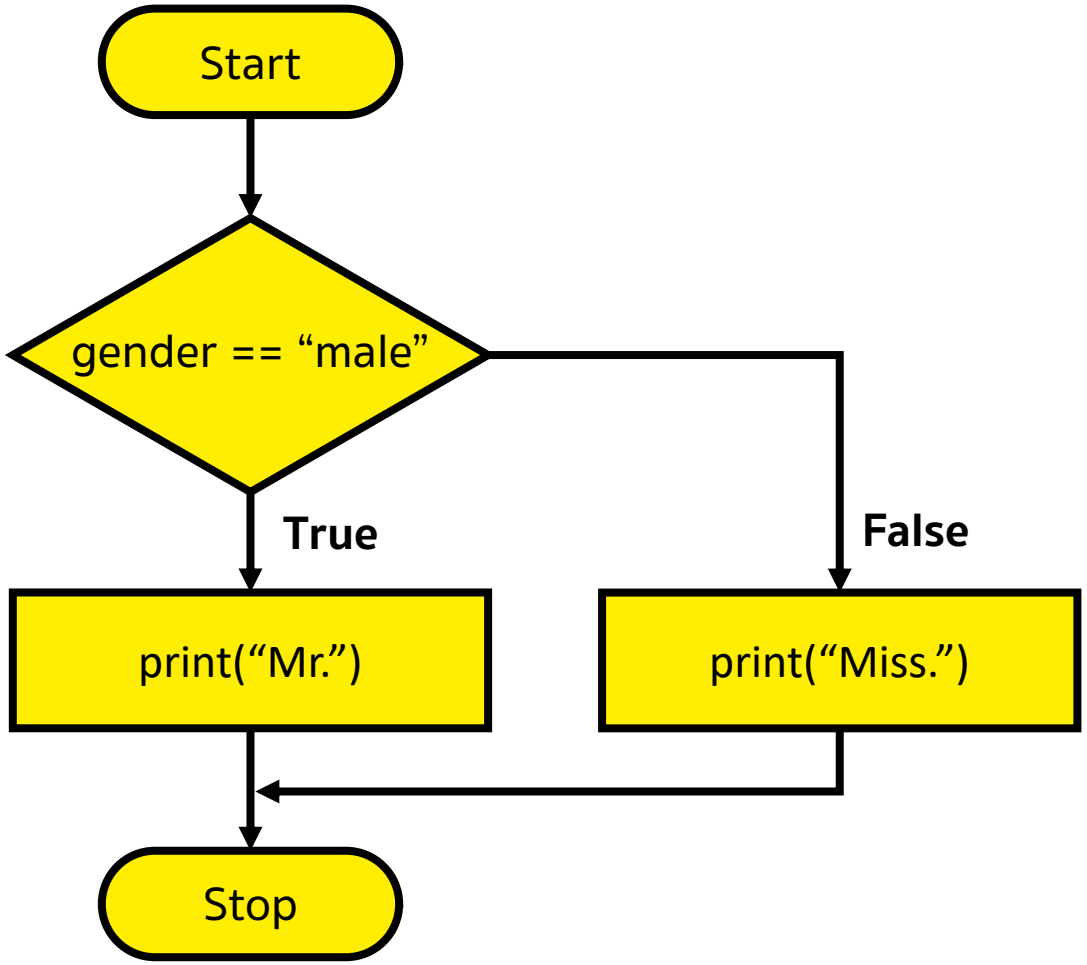

```
if( Condition ):
        Statement (IF)
else:
        Statement (ELSE)
```

ตัวดำเนินการเปรียบเทียบ (Comparison Operators)

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

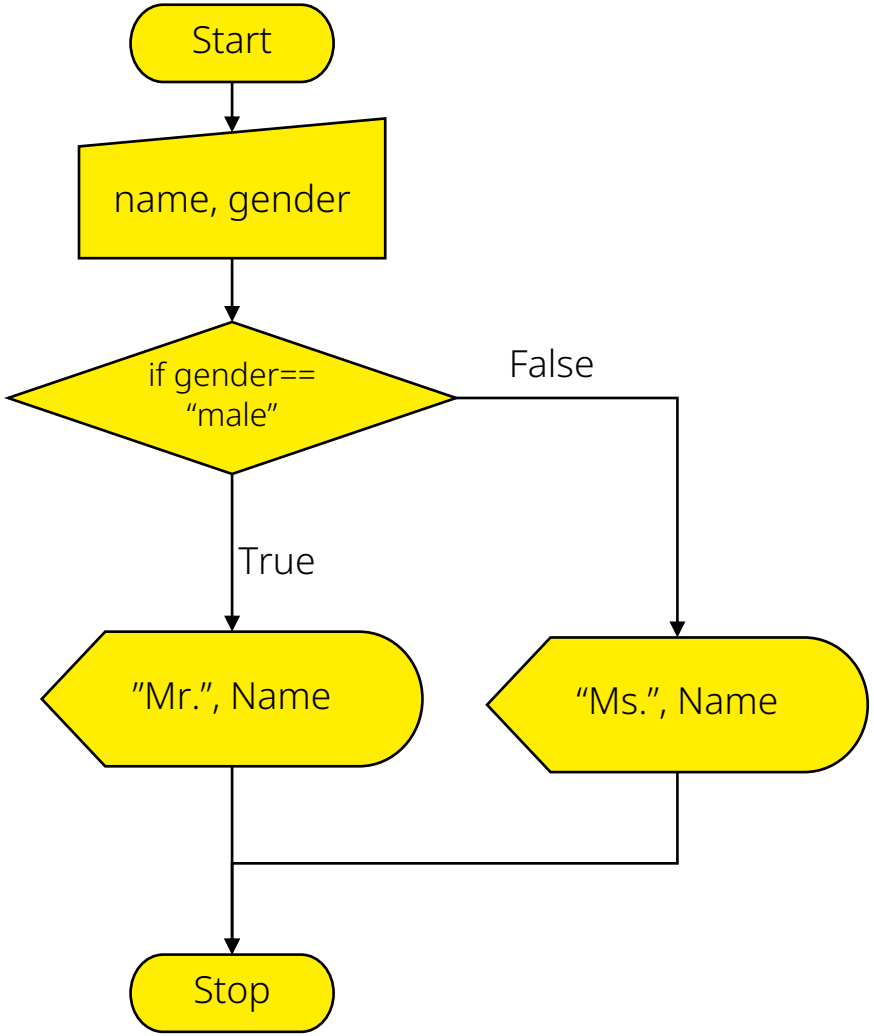
1. เท่ากับ (==): ตรวจสอบว่าค่าของสองค่าเท่ากันหรือไม่
   ตัวอย่าง: x == y จะเป็น True ถ้า x มีค่าเท่ากับ y
2. ไม่เท่ากับ (!=): ตรวจสอบว่าค่าของสองค่าไม่เท่ากัน
   ตัวอย่าง: x != y จะเป็น True ถ้า x ไม่เท่ากับ y
3. มากกว่า (>): ตรวจสอบว่าค่าด้านซ้ายมากกว่าค่าด้านขวา
   ตัวอย่าง: x > y จะเป็น True ถ้า 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 else




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

Practice 1: ให้รับข้อมูลเพศ และเติมคำนำหน้าชื่อตาม Flowchart

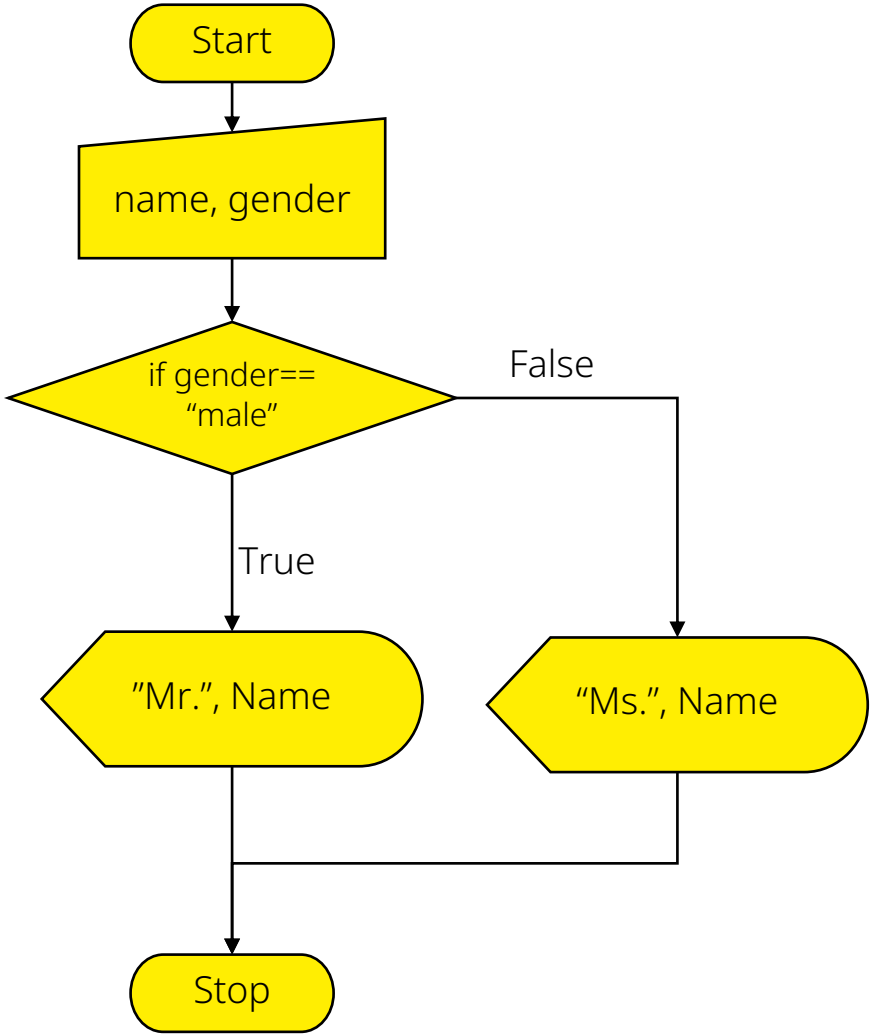


Input Name: ___Nutthapat___
Input Gender:__male___

↓

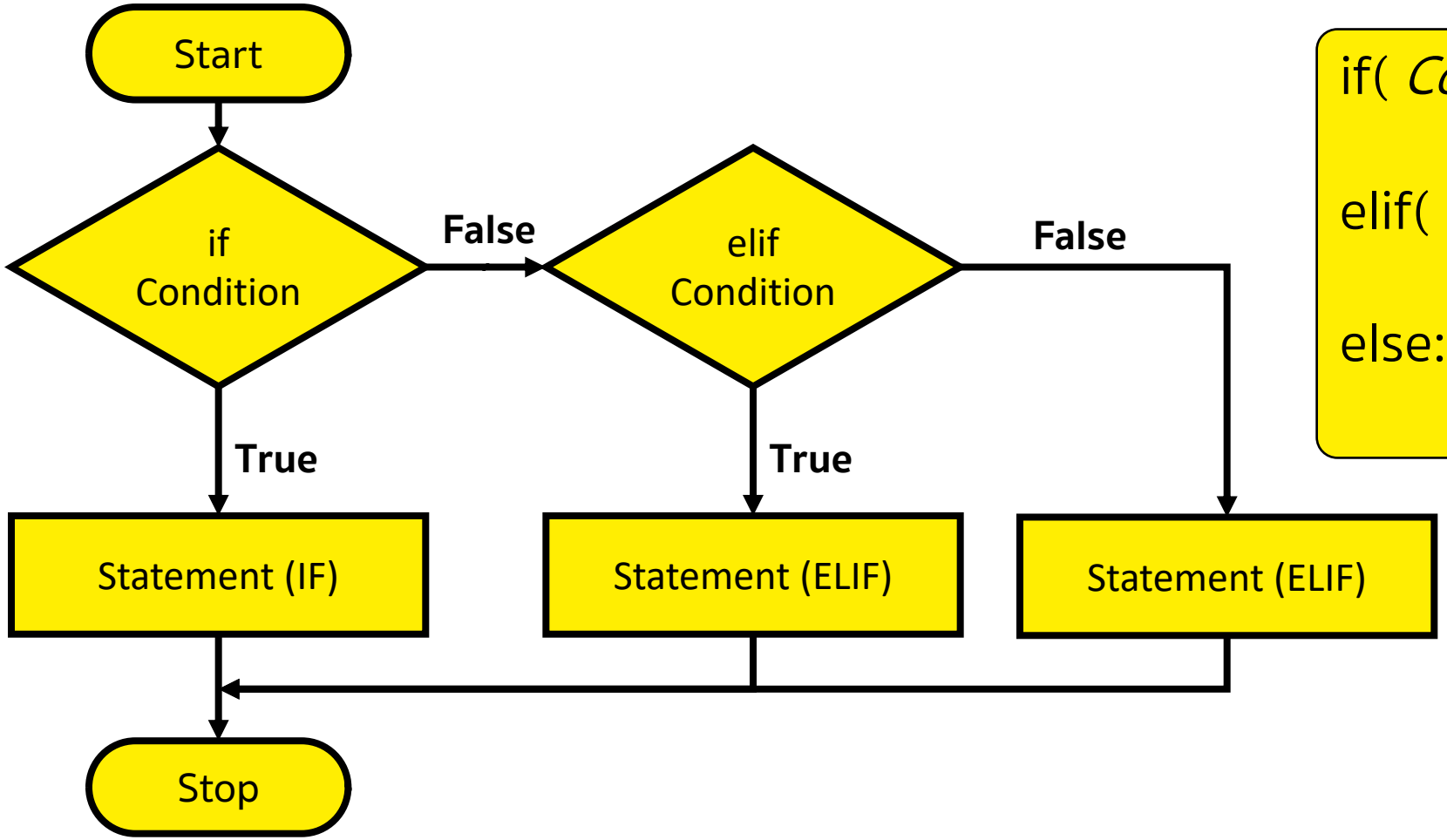
Mr.Nutthapat

Practice 1: ให้รับข้อมูลเพศ และเติมคำนำหน้าชื่อตาม Flowchart



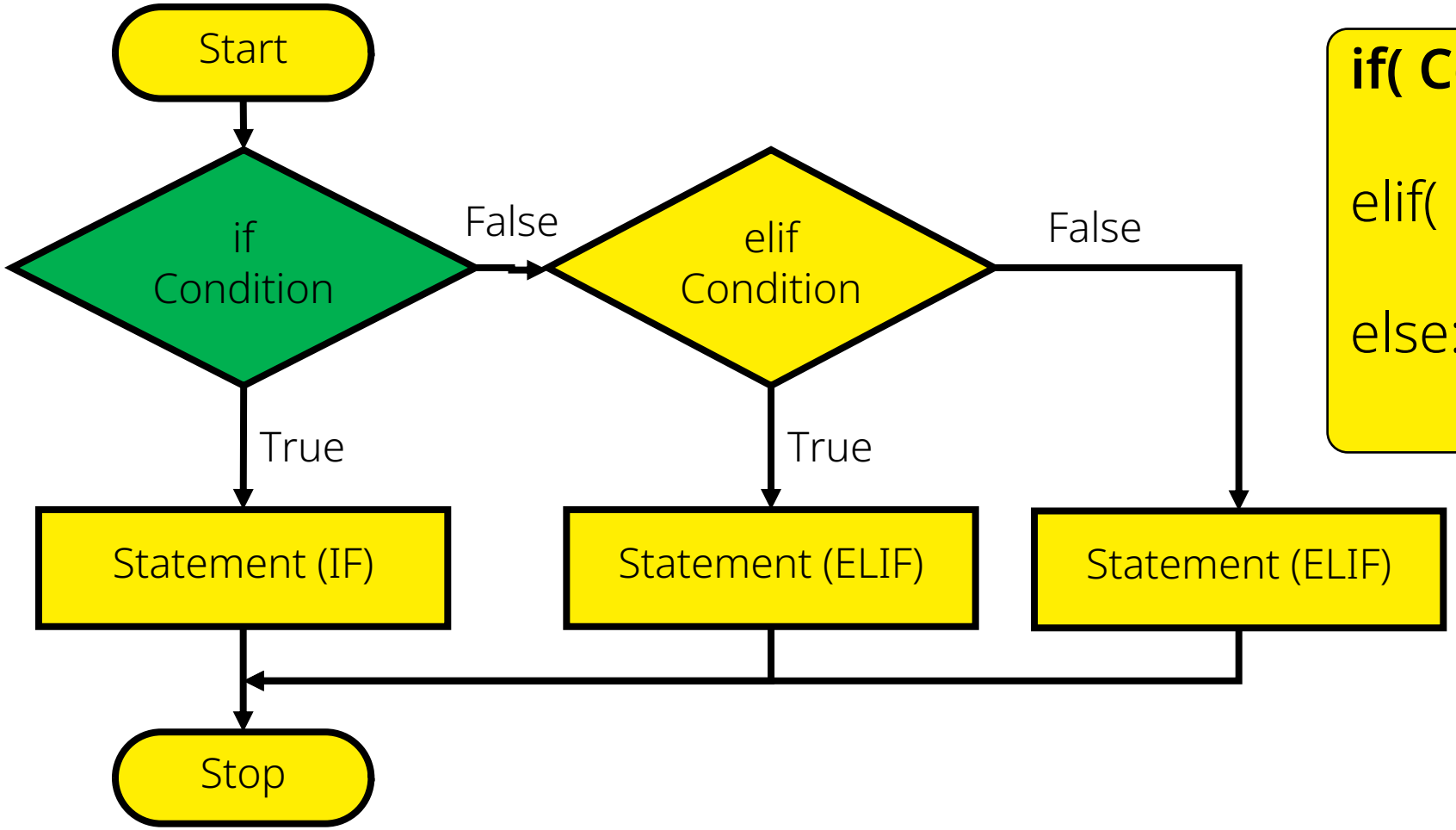

```
name = input("Name: ")
gender = input("Gender: ")
if(gender == "male"):
        print("Mr. "+ name)
else:
        print("Ms. "+ name)
```

Conditional Statement (คำสั่งเงื่อนไข) – if elif else



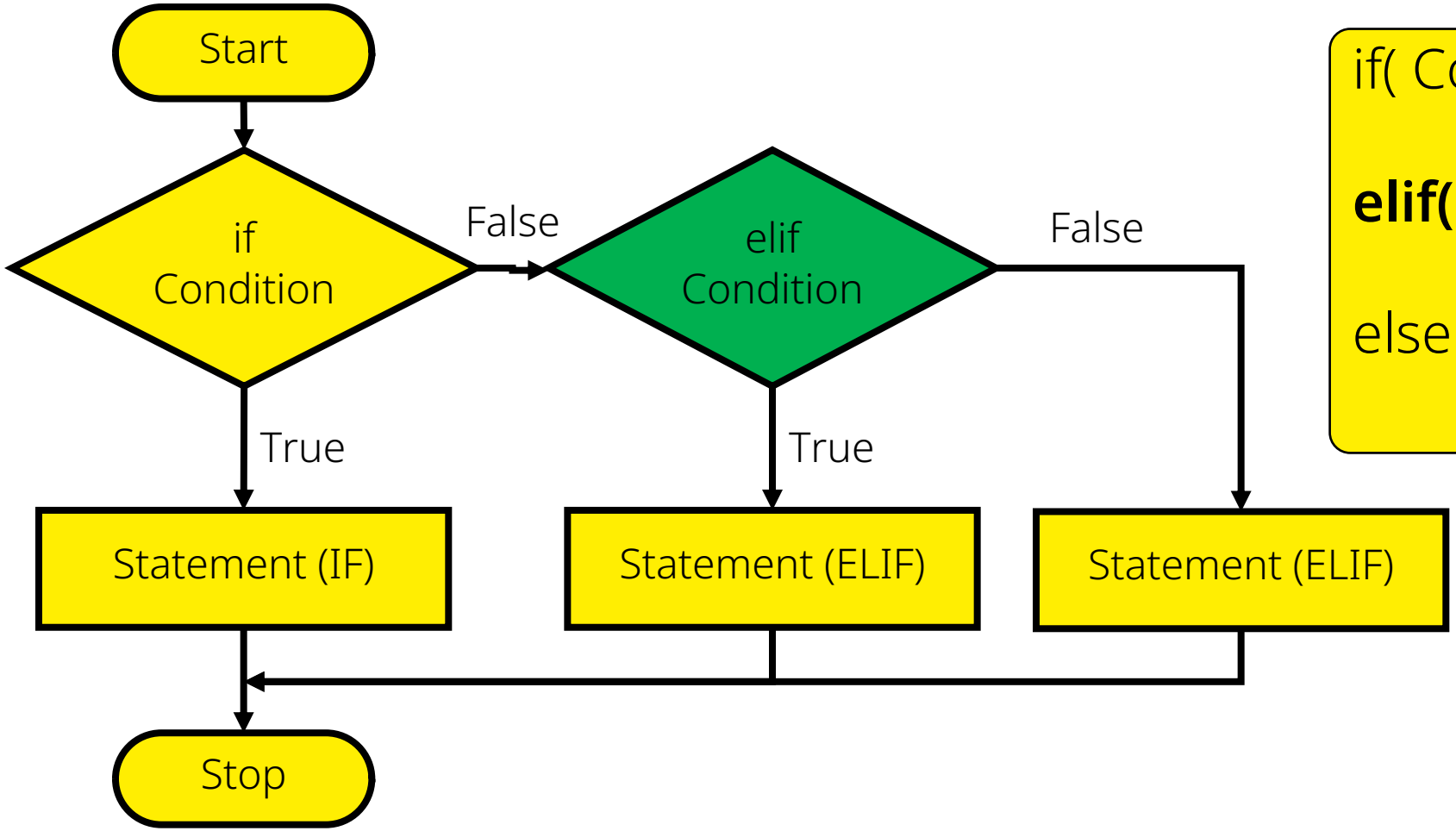

```
if( Condition ):
        Statement (IF)
elif( Condition) :
        Statement (ELIF)
else:
        Statement (ELSE)
```

Conditional Statement (คำสั่งเงื่อนไข) – if elif else



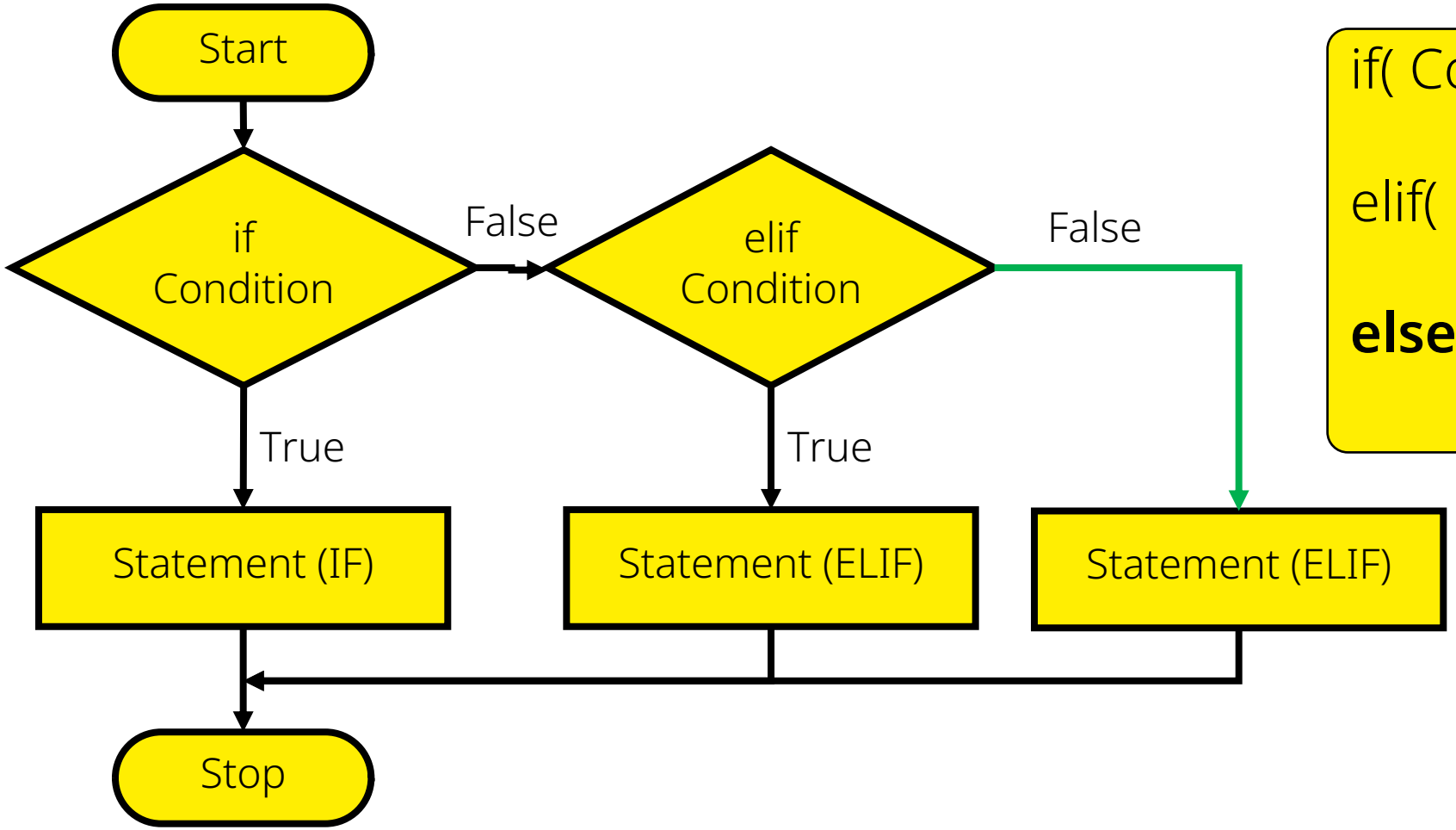

```
if( Condition ):
        Statement (IF)
elif( Condition) :
        Statement (ELIF)
else:
        Statement (ELSE)
```

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 2
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 ให้ พิมพ์
   ข้อความ product, "20 Thai Baht”

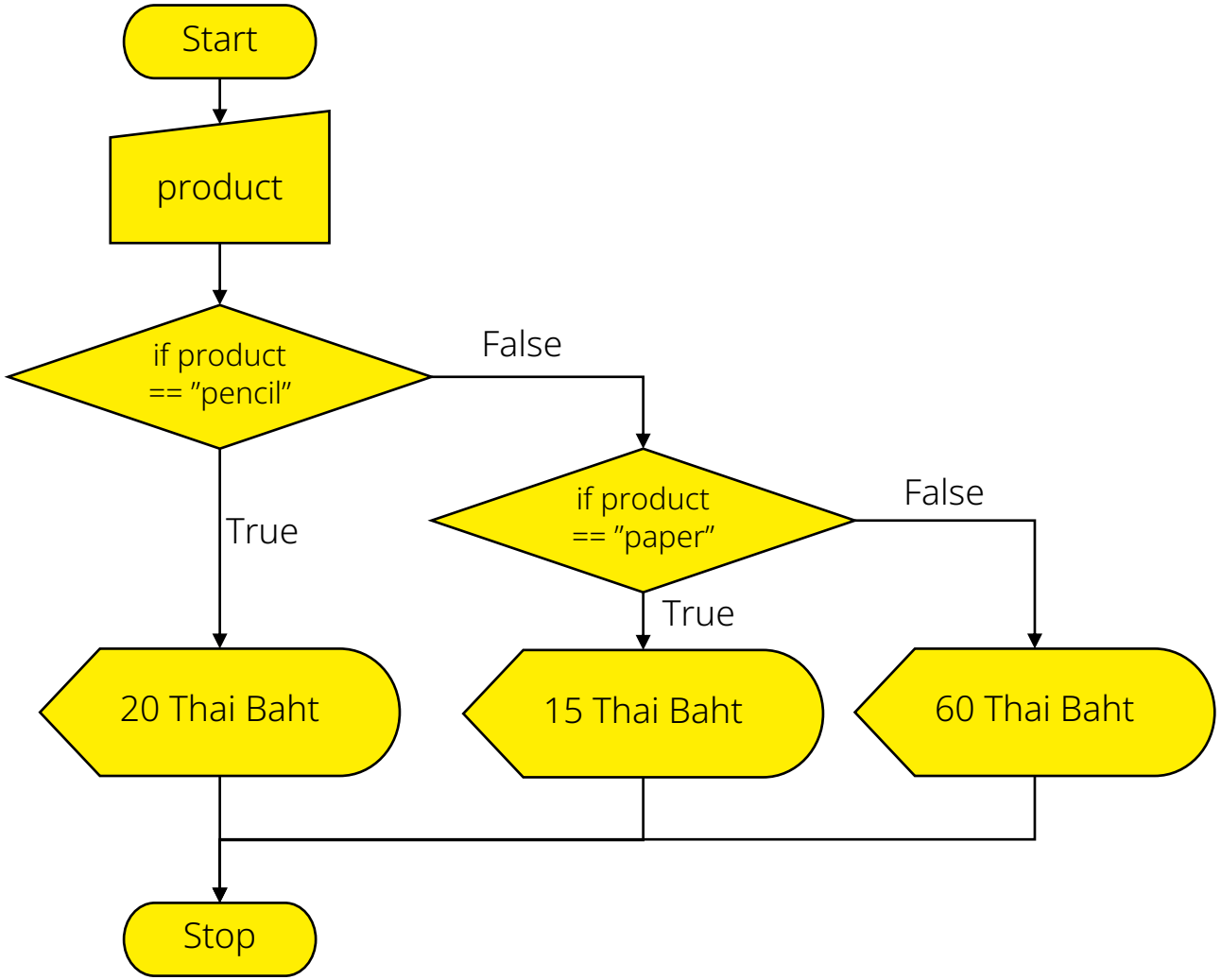
4.ตรวจสอบเงื่อนไขที่สอง (ถ้าเงื่อนไขแรกไม่เป็นจริง):
   ถ้า product == "paper" เป็นจริง ให้ พิมพ์
   ข้อความ product, "15 Thai Baht”

5.ถ้าเงื่อนไขแรกและเงื่อนไขที่สองไม่เป็นจริง ให้ไปที่ else:
   พิมพ์ข้อความ product, "60 Thai Baht”

6.จบการทำงาน

Conditional Statement (คำสั่งเงื่อนไข) – if elif else

```
# Condition Statement
# Code 2
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 แล้วเหลือเศษ 1

20 % 4 จะได้ผลลัพธ์เป็น 0 เพราะ 20 หาร 4 ลงตัว ไม่มีเศษ

15 % 7 จะได้ผลลัพธ์เป็น 1 เพราะ 15 หาร 7 แล้วเหลือเศษ 1

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

modulus (หรือเครื่องหมาย %)

```
number = 8
if number % 2 == 0:
  print(number,"เป็นเลขคู่")
else:
  print(number,"เป็นเลขคี่")
```

8 เป็นเลขคู่

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 ในการเขียนโปรแกรมและการแก้ปัญหาทางคณิตศาสตร์

การสุ่มจำนวน (Random numbers)

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

```
1. # Code 3
2. import random

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

6. # สุ่มจำนวนทศนิยมในช่วง 0.0 ถึง 1.0
7. random_float = random.random()
8. 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)

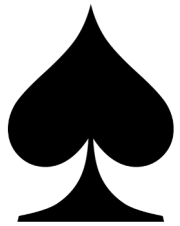
Practice 2: สุ่มและบอกว่าค่านั้นเป็นเลขคู่ หรือ เลขคี่

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

```
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("\u2660 ")
```

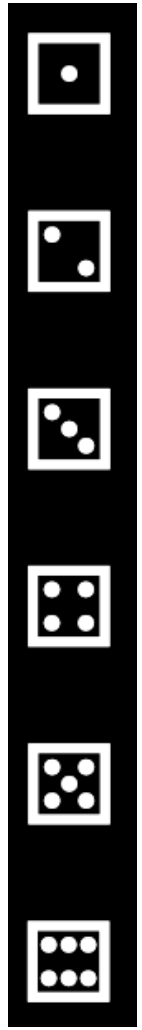


```
1  print("\u2660 ")
2  print("\u2665 ")
3  print("\u2666 ")
4  print("\u2663 ")
```

♠
♥
♦
♣

print( ) and Unicode

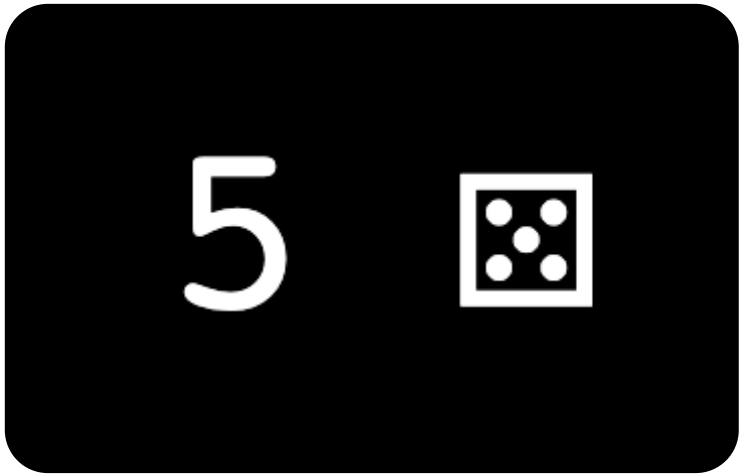
```
print("\u2680")
print("\u2681")
print("\u2682")
print("\u2683")
print("\u2684")
print("\u2685")
```



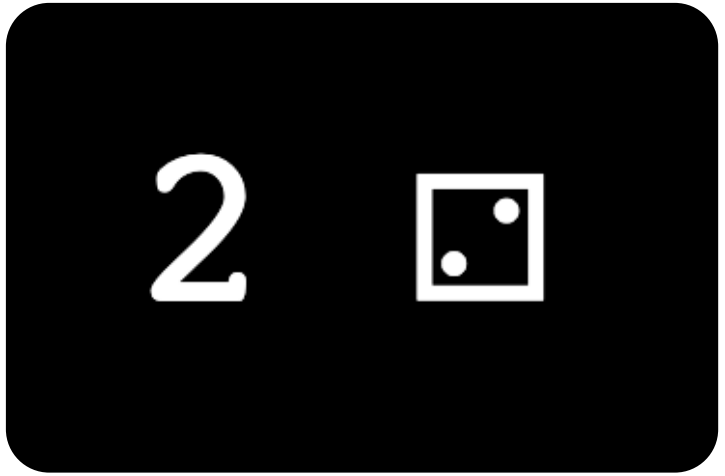
Practice 3 : Randomize numbers and convert them to the sides of the dice

ให้เขียนโปรแกรมเพื่อสุ่มตัวเลขและแปลงให้เป็นด้านของลูกเต๋า

Randomize numbers



Randomize numbers

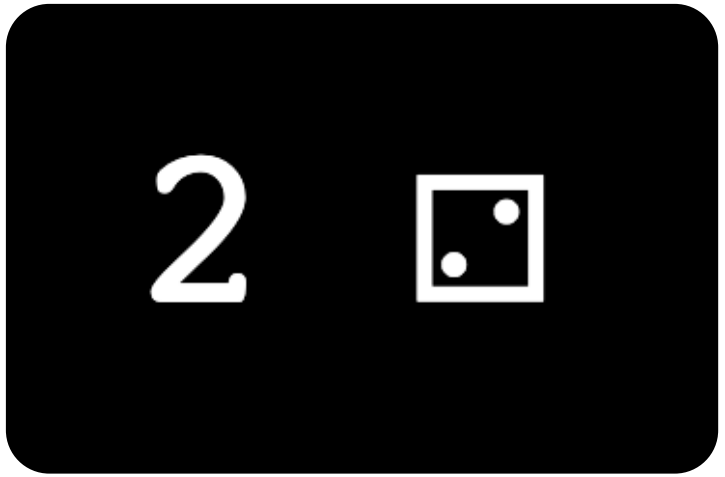


Practice 3 : Randomize numbers and convert them to the sides of the dice

ให้เขียนโปรแกรมเพื่อสุ่มตัวเลขและแปลงให้เป็นด้านของลูกเต๋า

```
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)
```

Randomize numbers



Assignment 3: Rock–Paper–Scissors

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

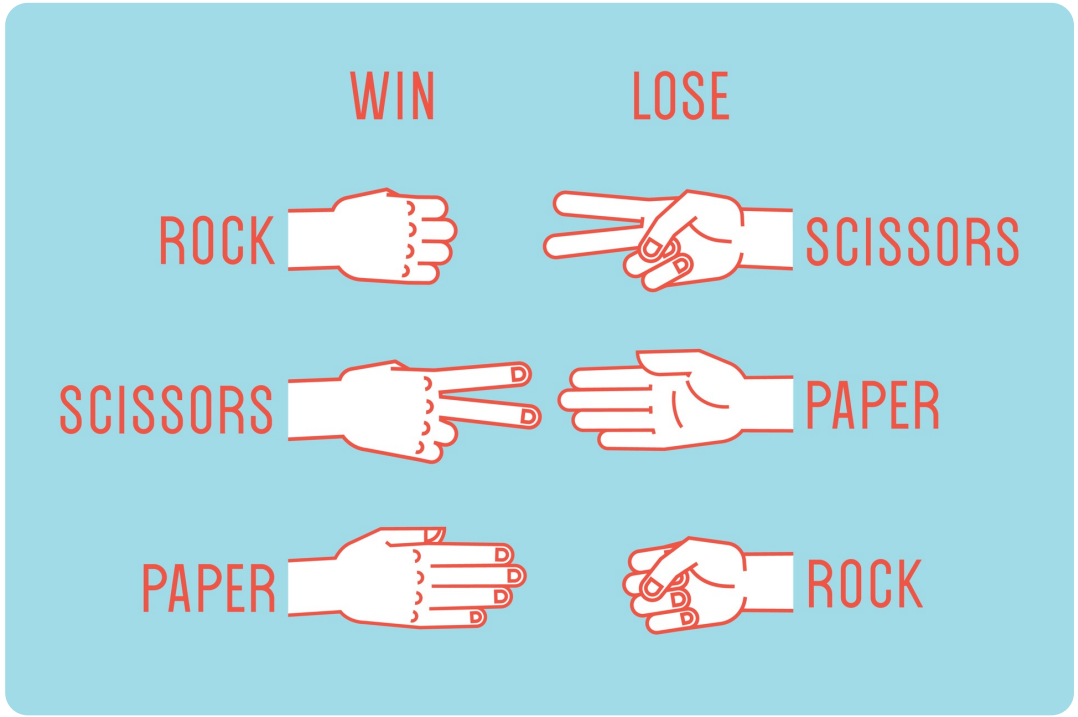
```
Rock-Paper-Scissors by Nutthapat
1 = Rock
2 = Scissors
3 = Paper
Your turn:
```

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

Assignment 3: Rock–Paper–Scissors

```python
import random
print("Rock-Paper-Scissors by Nutthapat")
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 Nutthapat
1 = Rock
2 = Scissors
3 = Paper
Your turn: 1
Your choice: Rock and Bot: Scissors
 You Win!
```



โครงสร้างการควบคุมโปรแกรม (Program Control Structures)

โครงสร้างการควบคุมโปรแกรม (Program Control Structures) ในการเขียนโปรแกรมมี 3 ลักษณะหลัก ได้แก่

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

แต่ละลักษณะมีความสำคัญและทำหน้าที่ที่แตกต่างกันในการควบคุมการทำงานของโปรแกรม

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

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

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

```
Hello, Nutthapat
Hello, Thanadol
Hello, Natthapol
Hello, Vilasinee
Hello, Wichuda
Hello, Chinchira
Hello, Phachaya
Hello, Manop
Hello, Panita
Hello, Naphasri
```

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

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

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

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

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
```

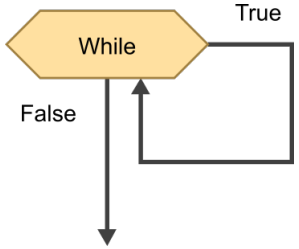
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็นเท็จ (False)

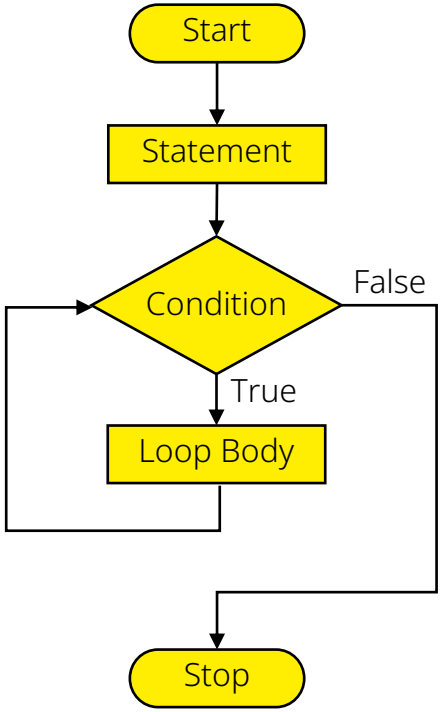
```
for loop syntax:

for item in collection:
    # คำสั่งที่ต้องการทำงานกับ item
```

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

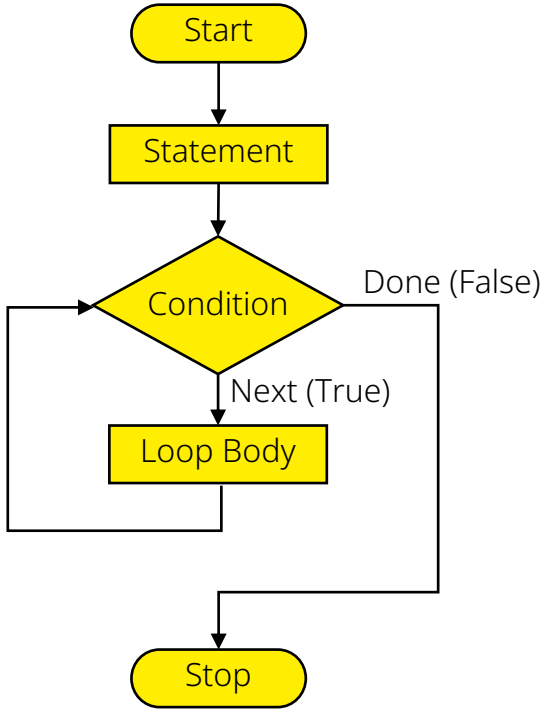
การโปรแกรมแบบทำซ้ำ (Iteration Statement)

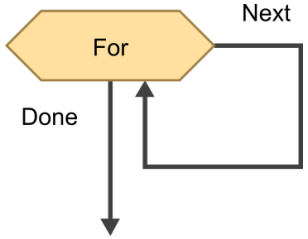
โครงสร้างการควบคุมทำซ้ำแบบ while
While Statement





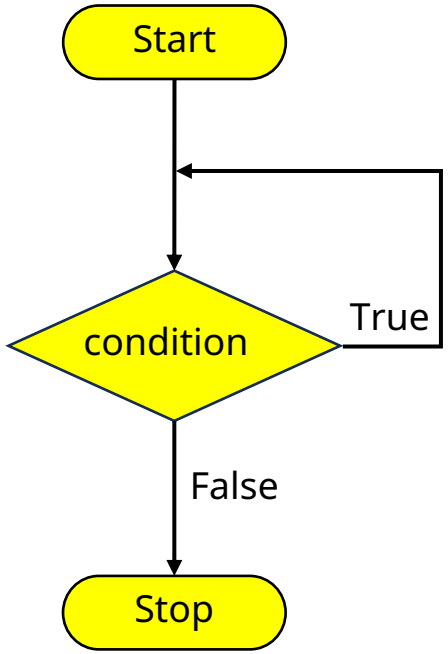
โครงสร้างการควบคุมทำซ้ำแบบ for
For Statement

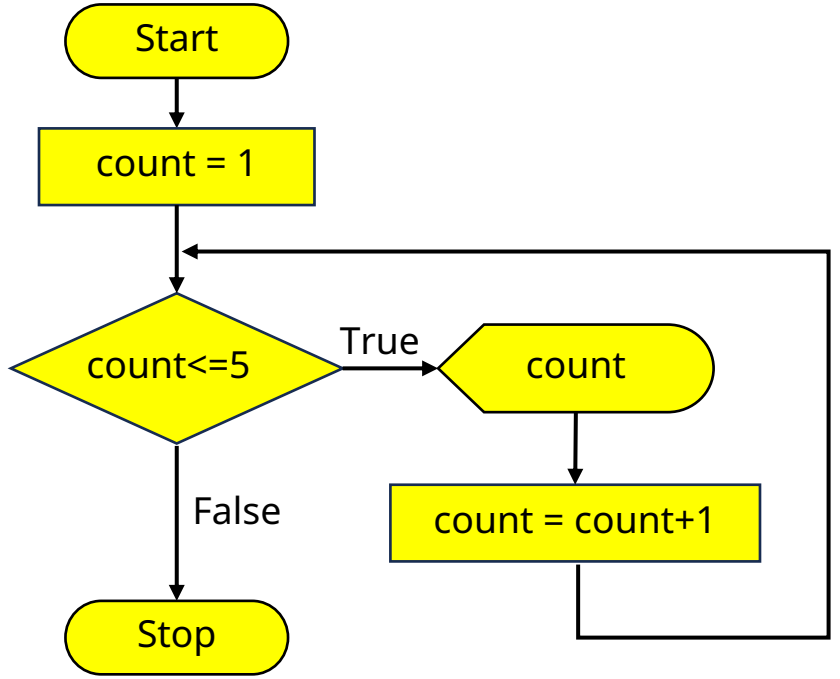




การโปรแกรมแบบทำซ้ำ (Iteration Statement)

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



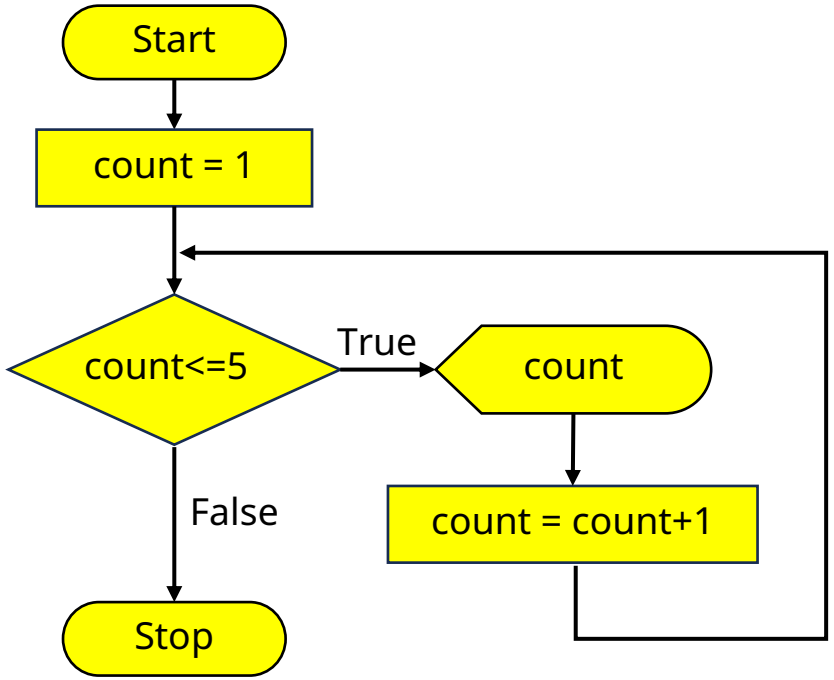


Python

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

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

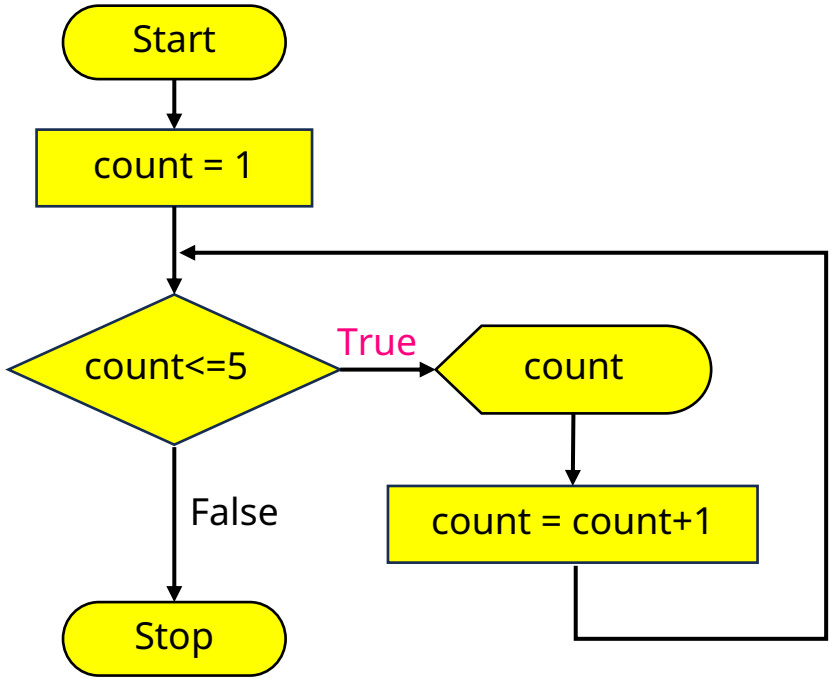
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



รอบที่	ค่าของตัวแปร count	เงื่อนไข (True, False)	แสดงผล

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

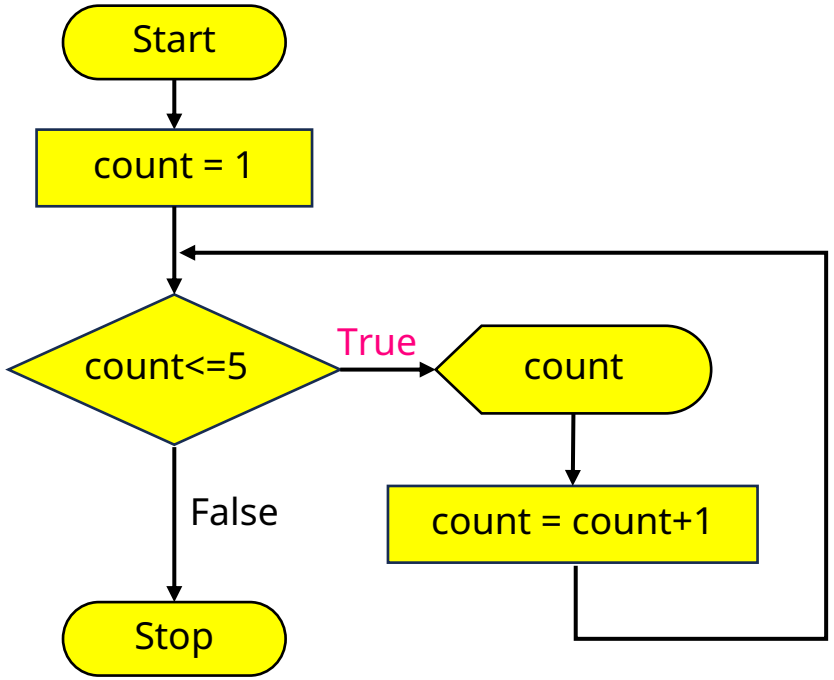
การใช้ while loop เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



รอบที่	ค่าของตัวแปร count	เงื่อนไข (True, False)	แสดงผล
1	1	True	1

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

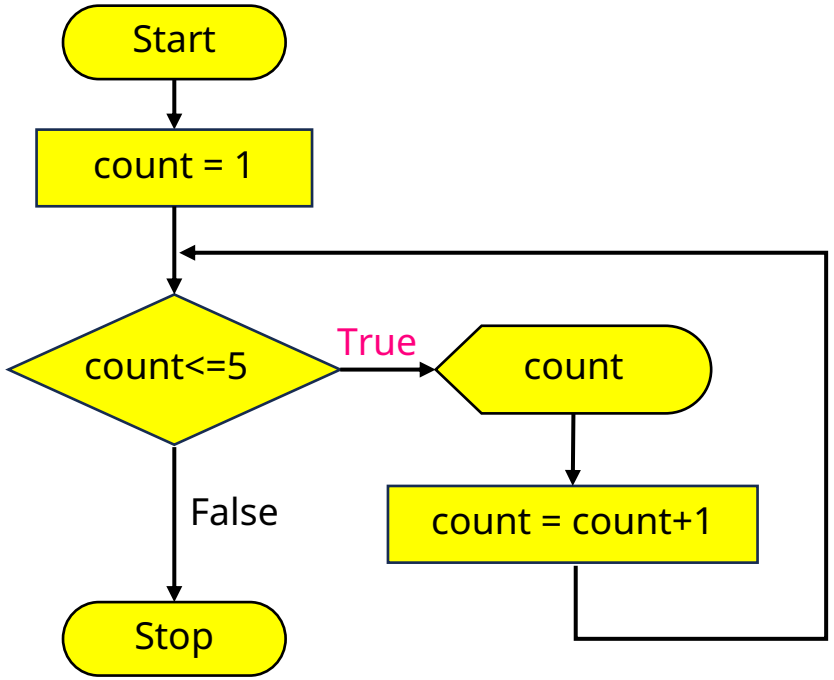
การใช้ **while loop** เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



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

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

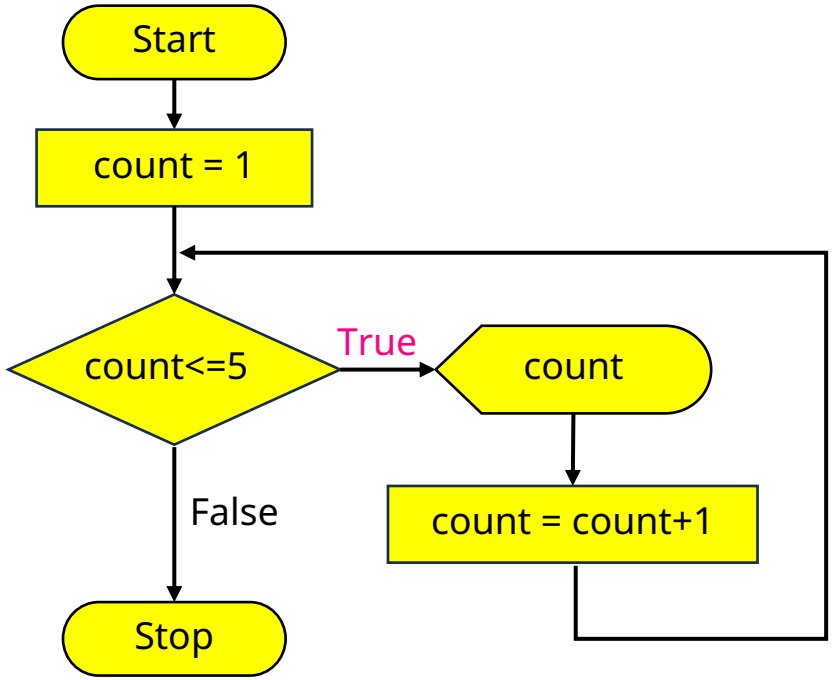
การใช้ while loop เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



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

การโปรแกรมแบบทำซ้ำ (Iteration Statement)

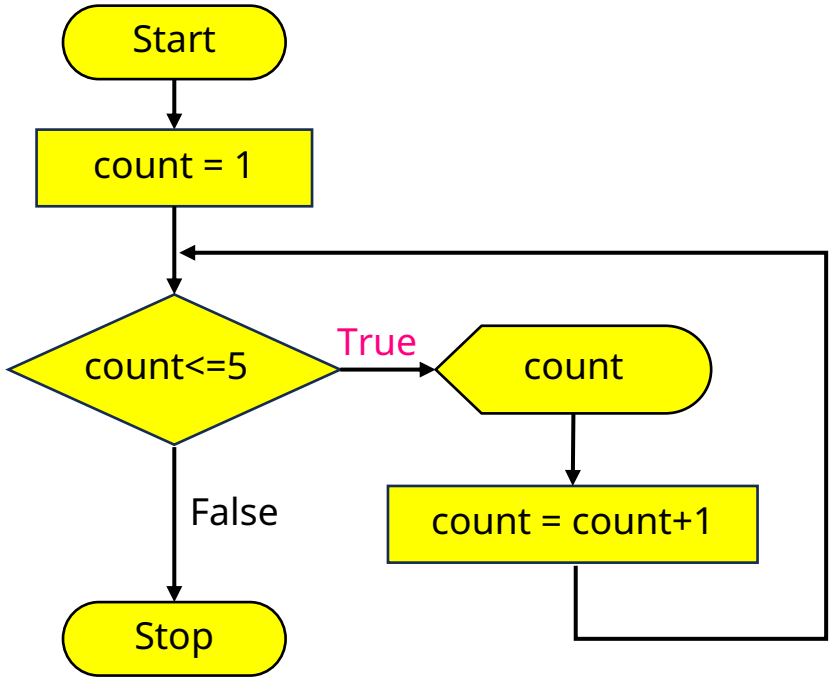
การใช้ while loop เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



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)

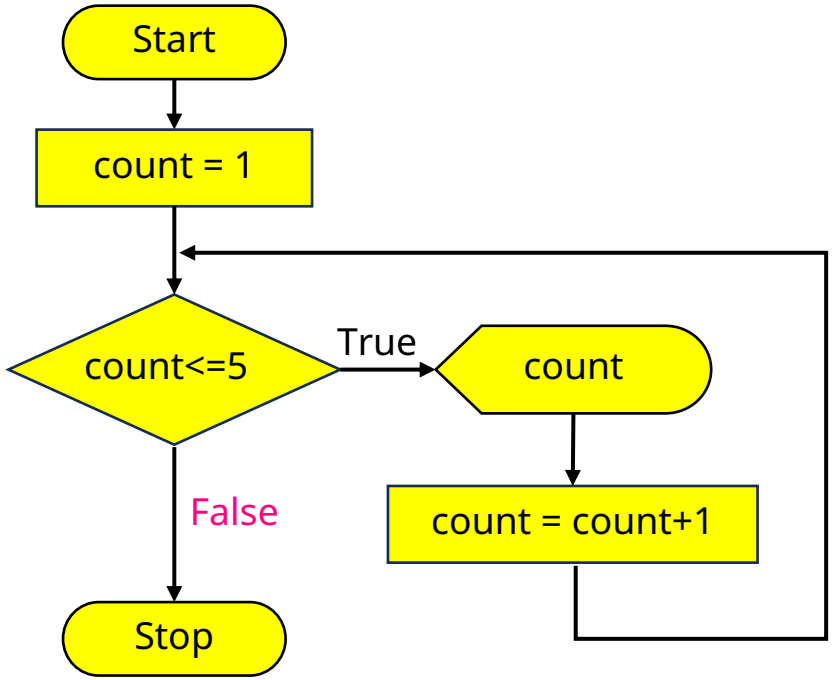
การใช้ while loop เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



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)

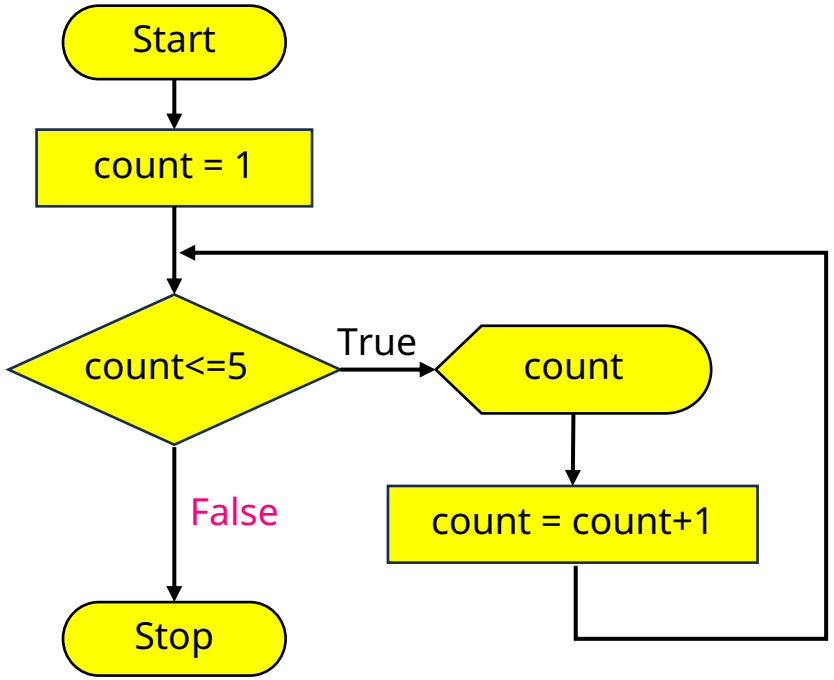
การใช้ while loop เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ (False)**



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)

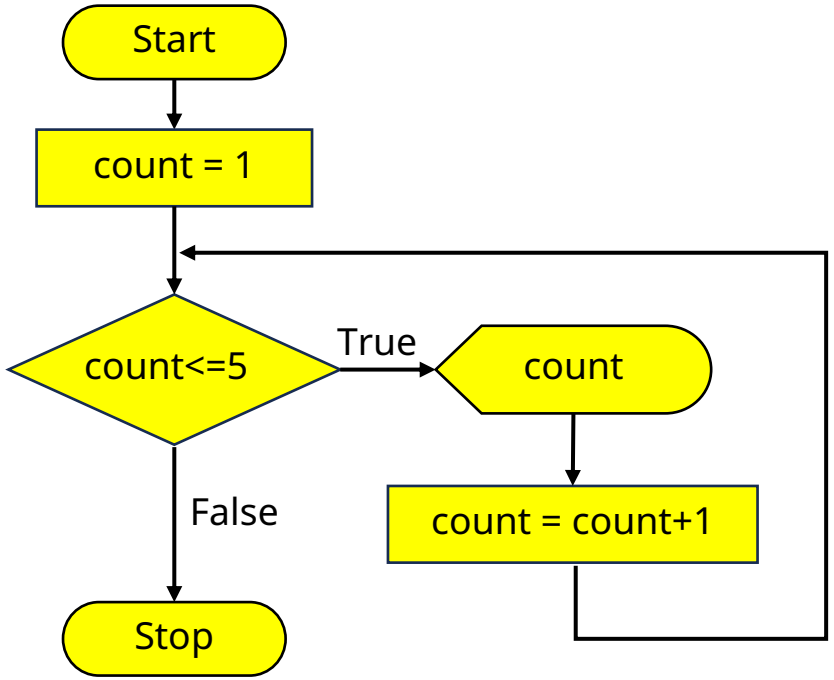
การใช้ while loop เพื่อทำงานซ้ำจนกว่าเงื่อนไขที่กำหนดจะเป็น**เท็จ** (False)



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

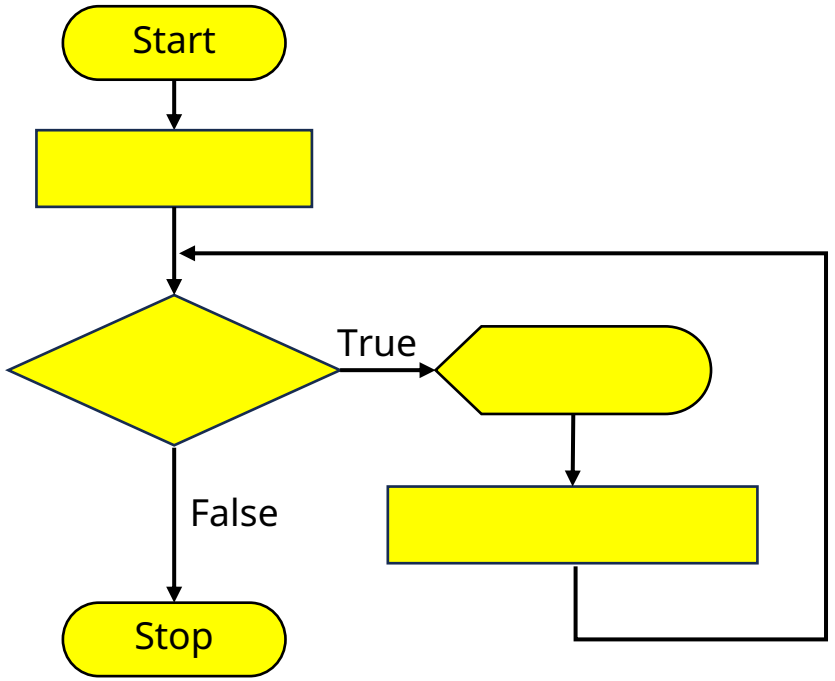



```
1.count = 1
2.while count<=5:
3.   print(count)
4.   count+=1
```

Python Tutor
https://cutt.ly/DwGxR4R2

Iteration Statement – while

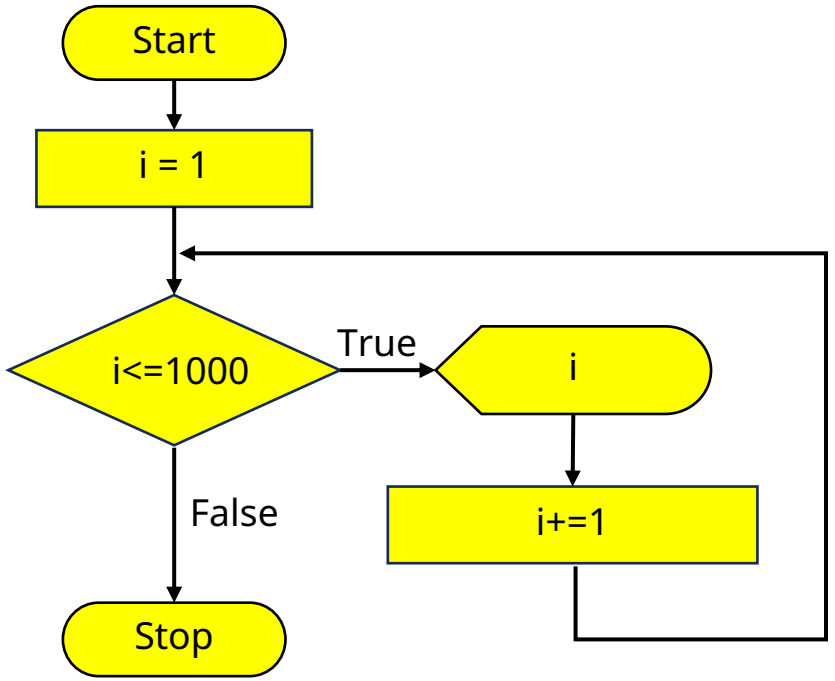
2. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 1000




```
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
```

Python Tutor
https://cutt.ly/MwGxAiG8

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 = 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

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

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


```
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

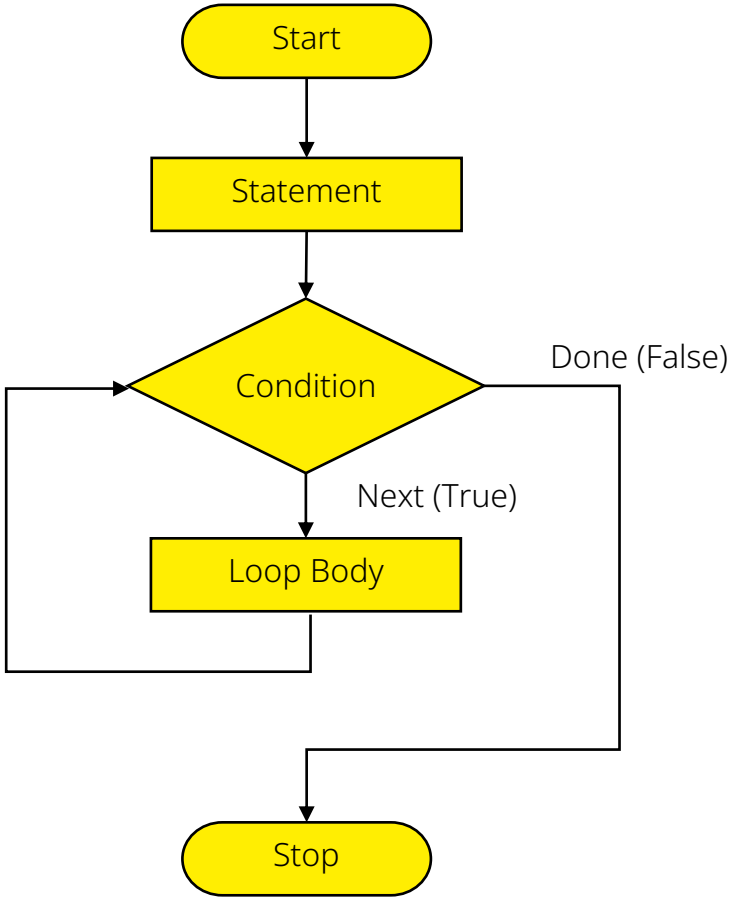
- 3! แสดงผล **3! = 3x2x1 = 6**

```python
# 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



ฟังก์ชั่น range( )

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

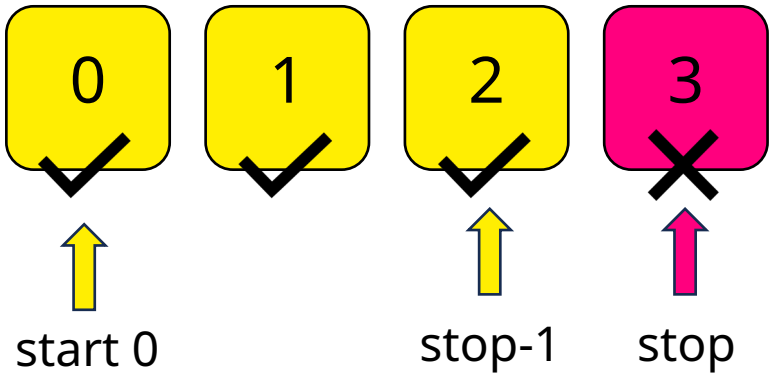
range() มีรูปแบบการใช้งาน 3 ลักษณะ ดังนี้

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

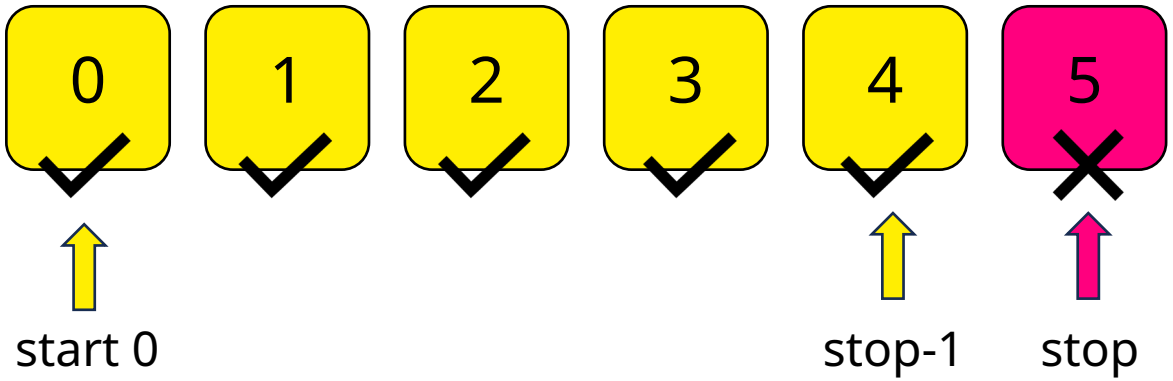
```
range(3)
```



ฟังก์ชั่น range( )

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

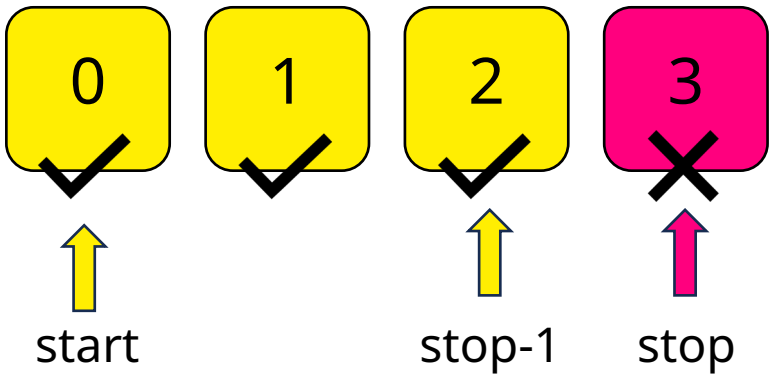
```
range(5)
```



ฟังก์ชั่น range( )

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

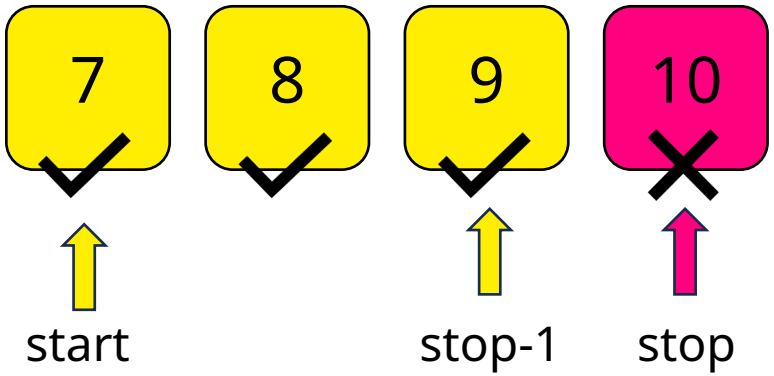
```
range(0,3)
```



ฟังก์ชั่น range( )

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

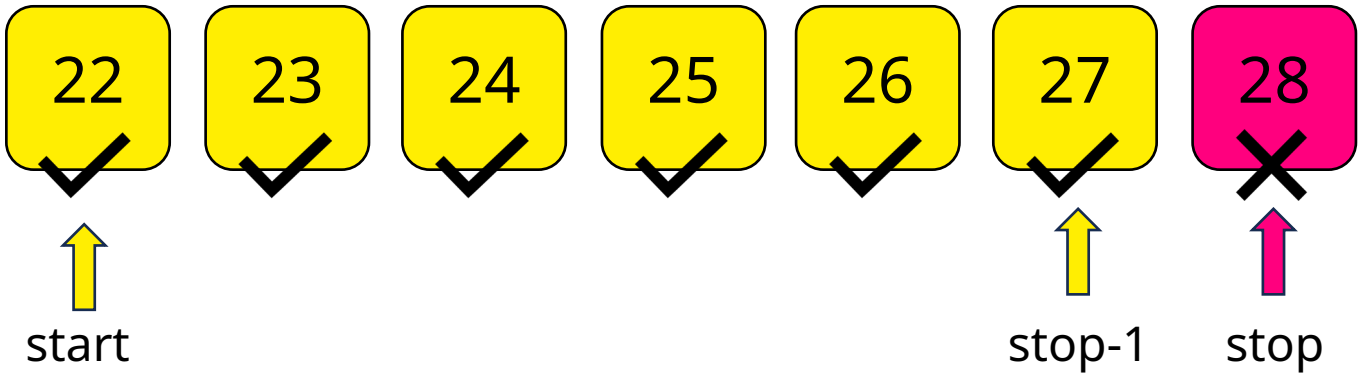
```
range(7,10)
```



ฟังก์ชั่น range( )

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

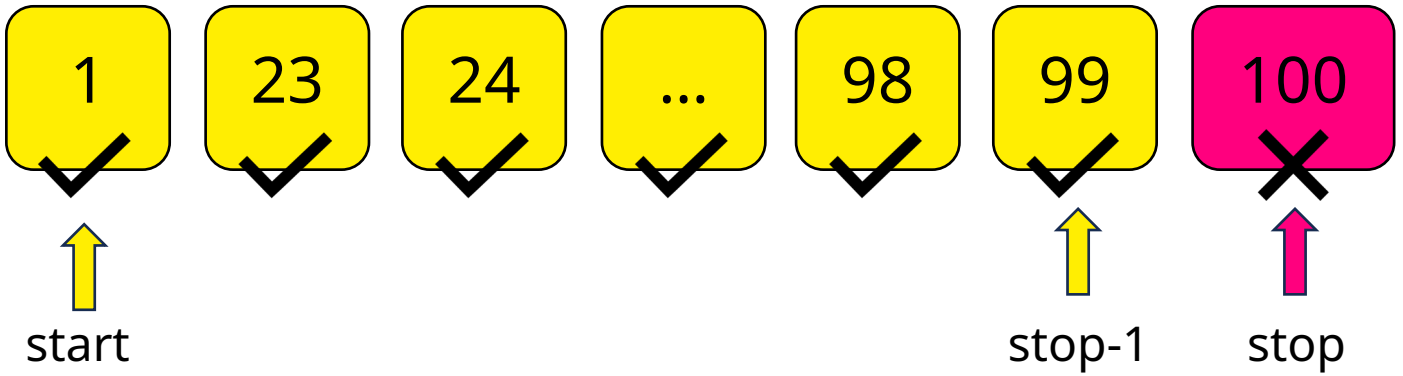
```
range(22,28)
```



ฟังก์ชั่น range( )

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

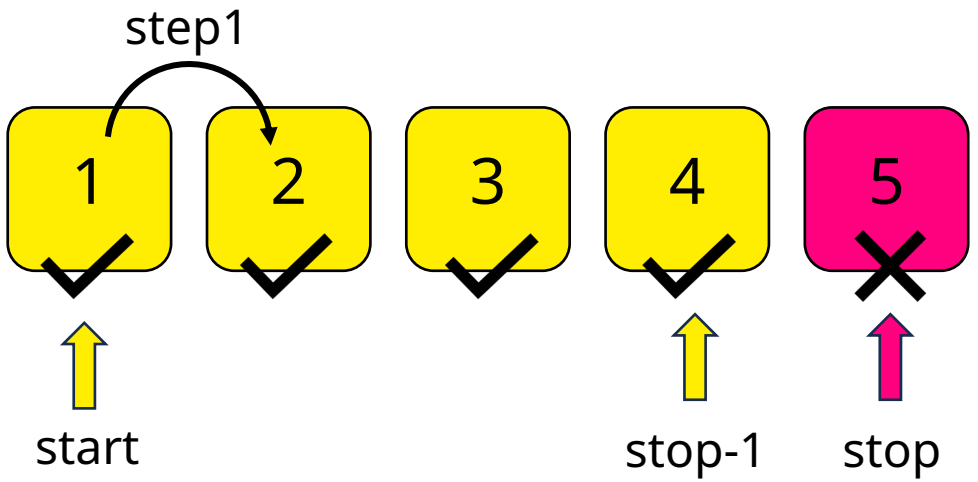
```
range(1,100)
```



ฟังก์ชั่น range( )

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

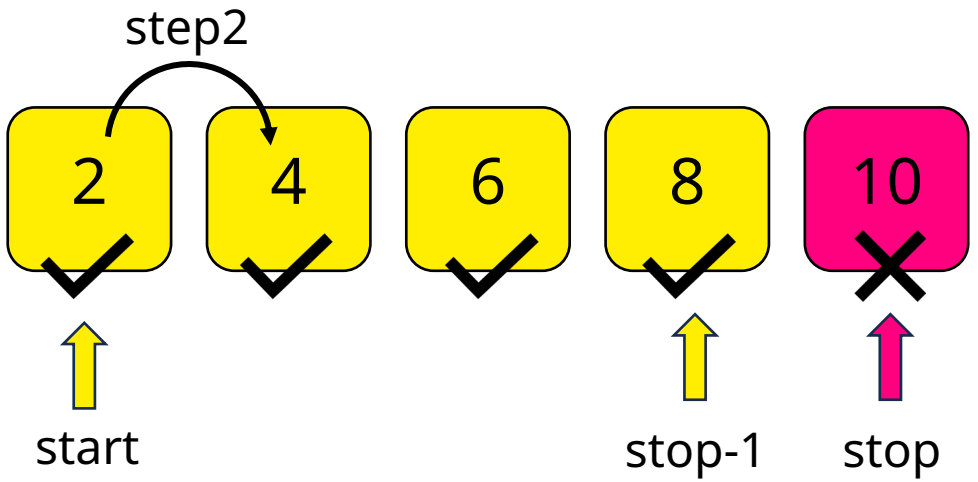
```
range(1,5,1)
```



ฟังก์ชั่น range( )

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

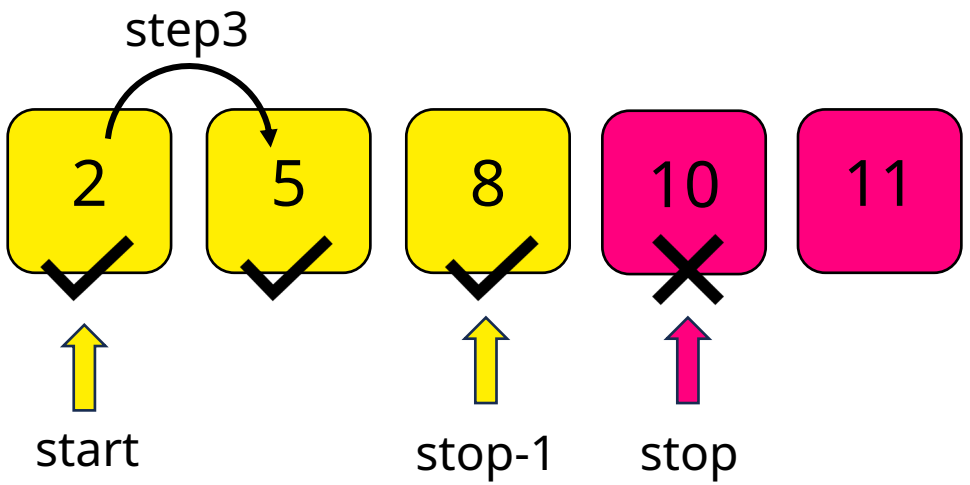
```
range(2,10,2)
```



ฟังก์ชั่น range( )

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

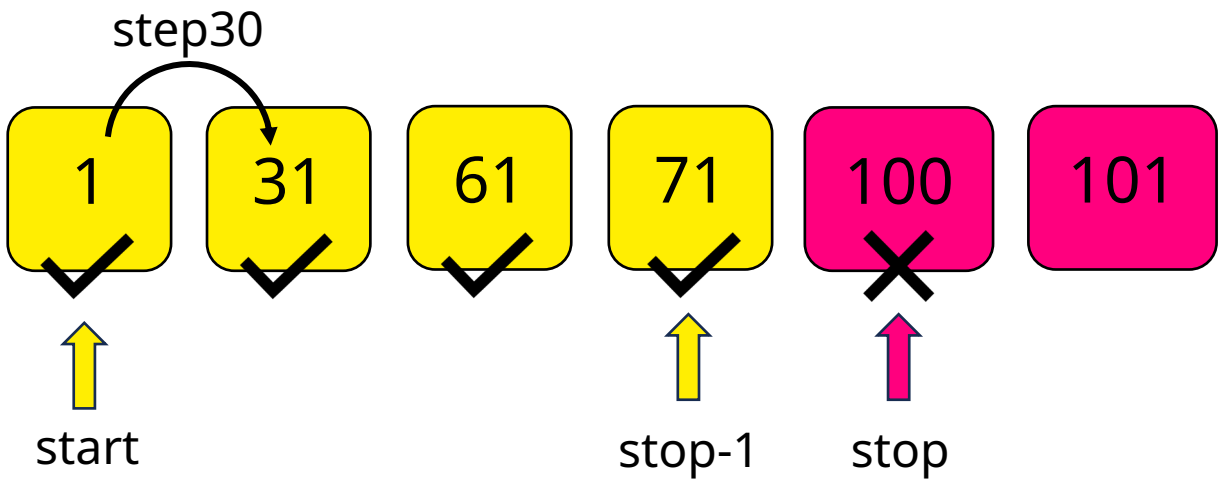
```
range(2,10,3)
```



ฟังก์ชั่น range( )

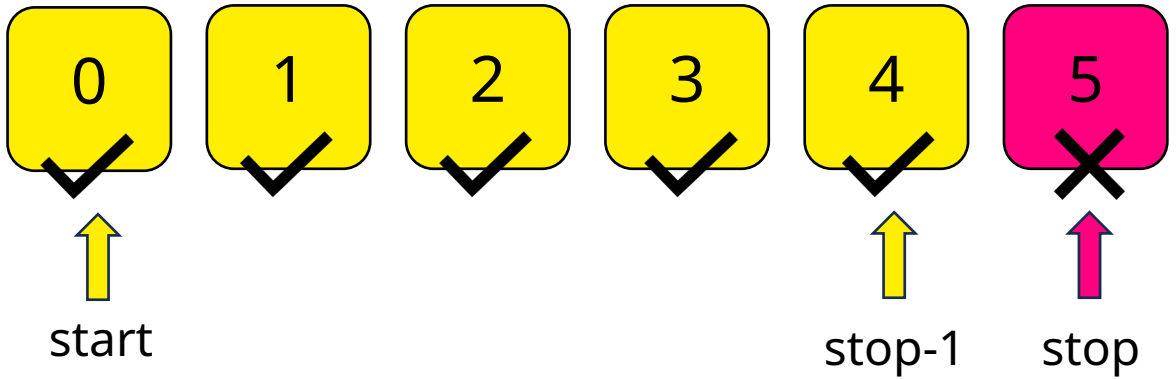
3. range(start, stop, step): สร้างชุดตัวเลขตั้งแต่ start ถึง stop - 1 โดยที่ลำดับของตัวเลขมีการเพิ่มขึ้นที่ละ step

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



Iteration Statement – for

1. ต้องการแสดงค่าข้อมูลใน range( ) จาก 0 ถึง 4



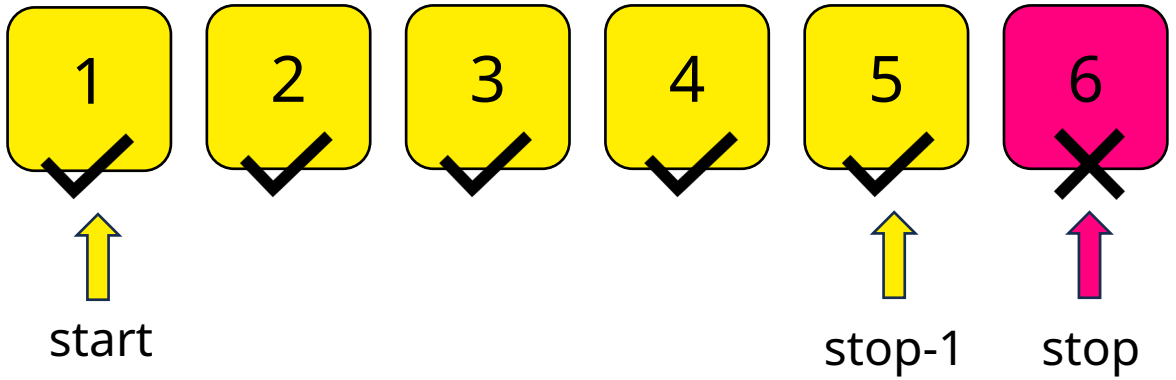

```
for i in range(5):
   print(i)
```

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

```
0
1
2
3
4
```

Iteration Statement – for

2. ต้องการแสดงค่าข้อมูลใน range( ) จาก 1 ถึง 5



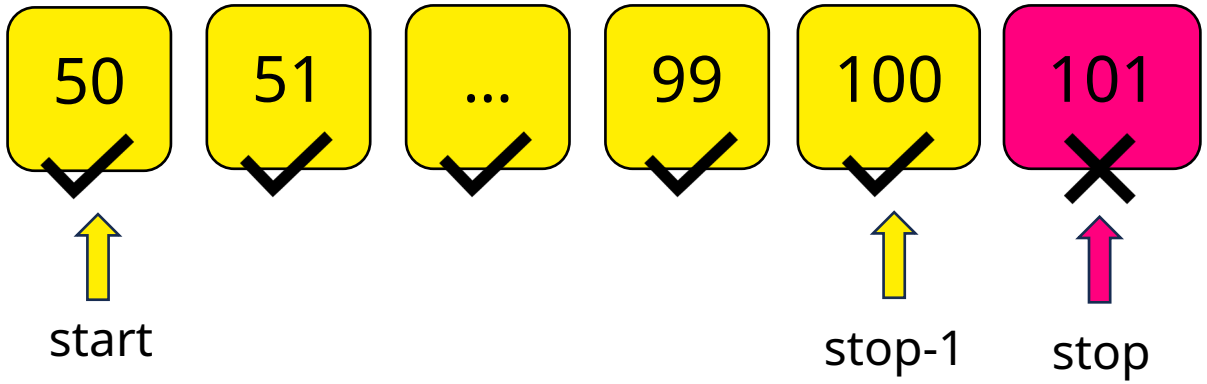

```
for i in range(1,6):
   print(i)
```

```
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)
```

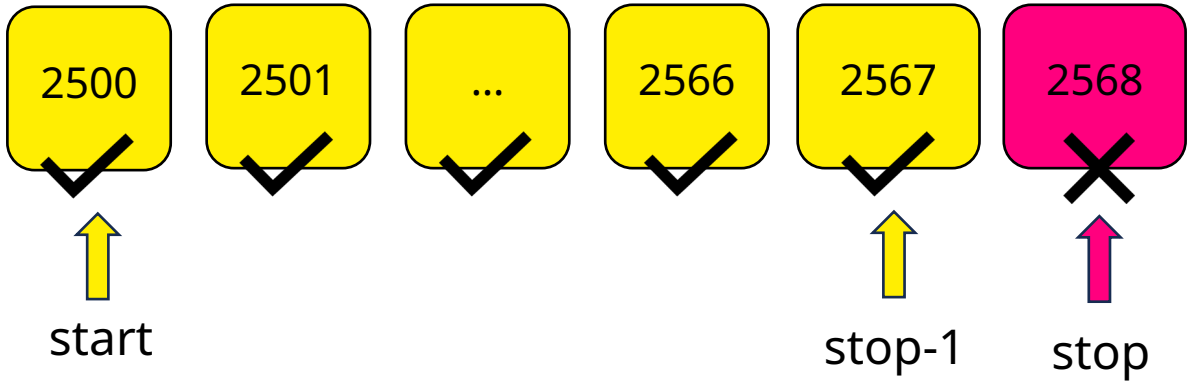
```
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
```

```
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

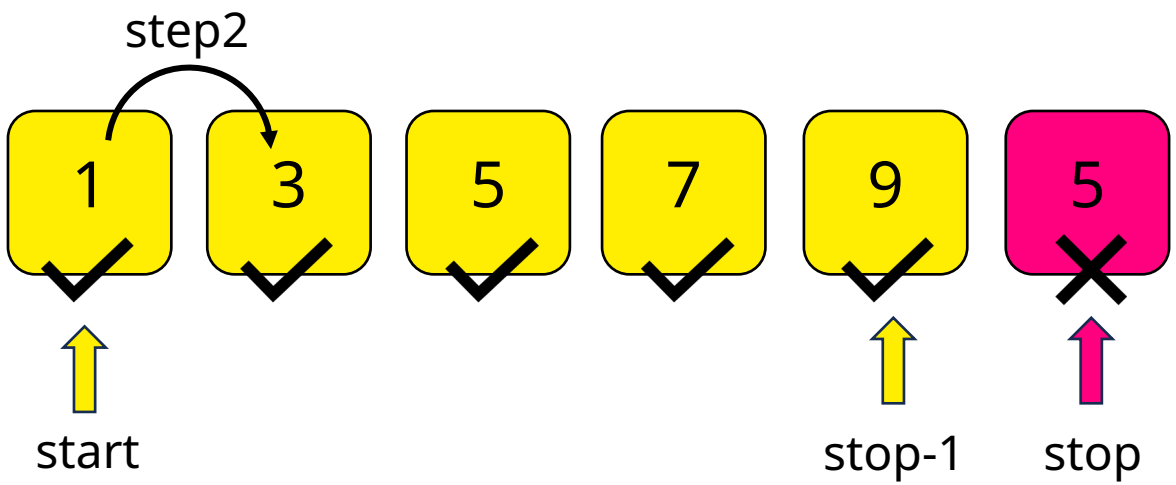
```
for i in range(_________,_____________):
  print(i)
```



Iteration Statement – for

5. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 1 ถึง 10 เพิ่มทีละ 2

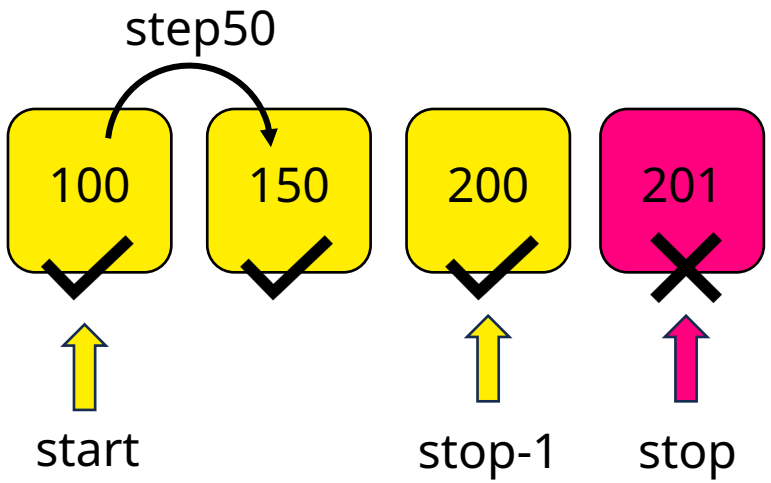
```
for i in range(_________,__________,________):
    print(i)
```



Iteration Statement – for

6. การโปรแกรมเพื่อแสดงผลจำนวนตั้งแต่ 100 ถึง 200 เพิ่มทีละ 50

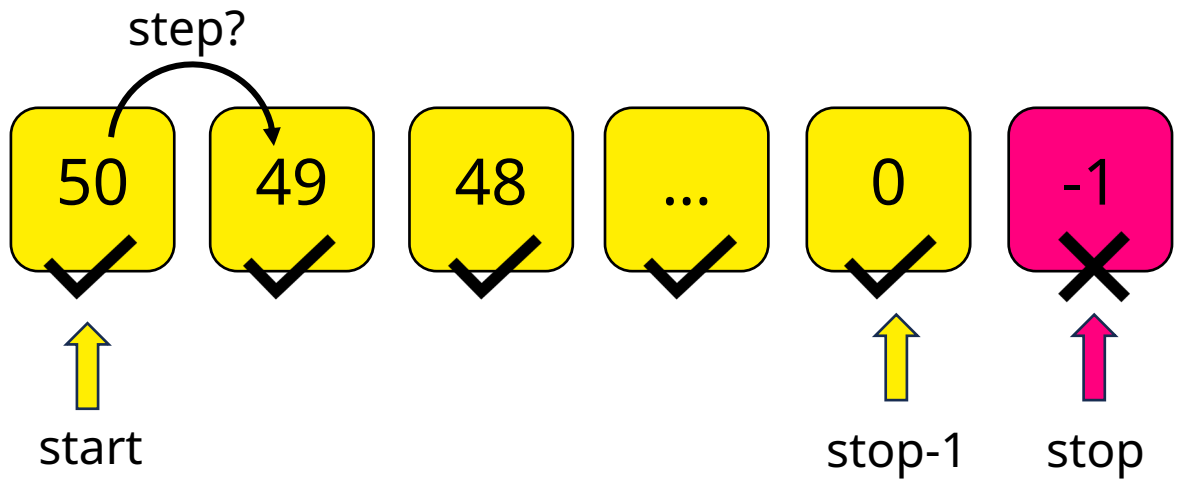
```
for i in range(_________,__________,________):
    print(i)
```



Iteration Statement – for

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

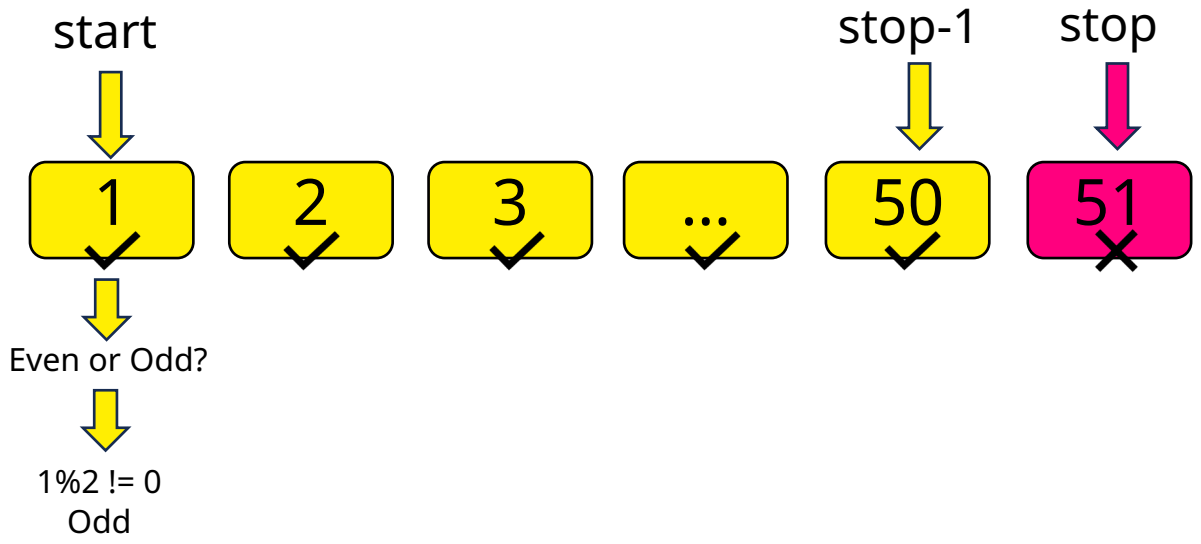
```
for i in range(_____,_____,_____):
    print(i)
```



Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

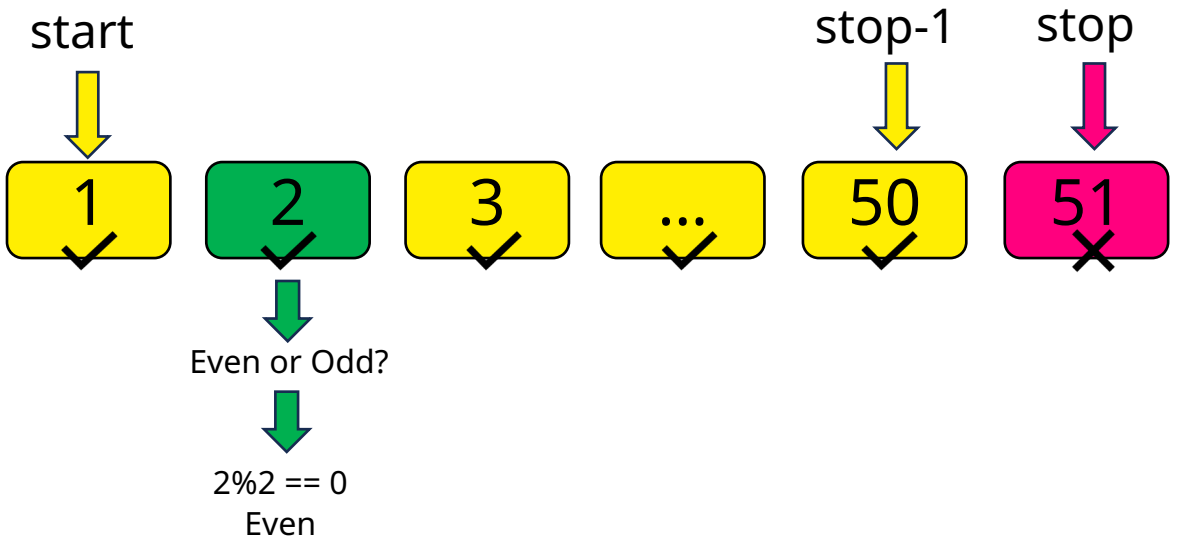
```
for i in range(_______,_______):
  if ___________:
    print(i)
```



Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

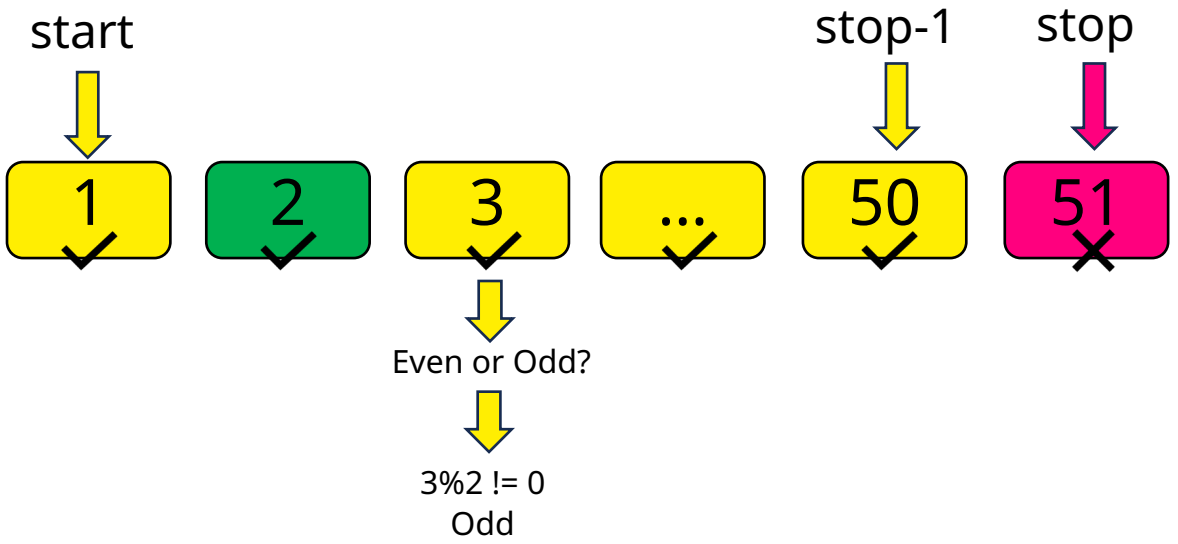
```
for i in range(_______,_______):
  if ___________:
    print(i)
```



Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

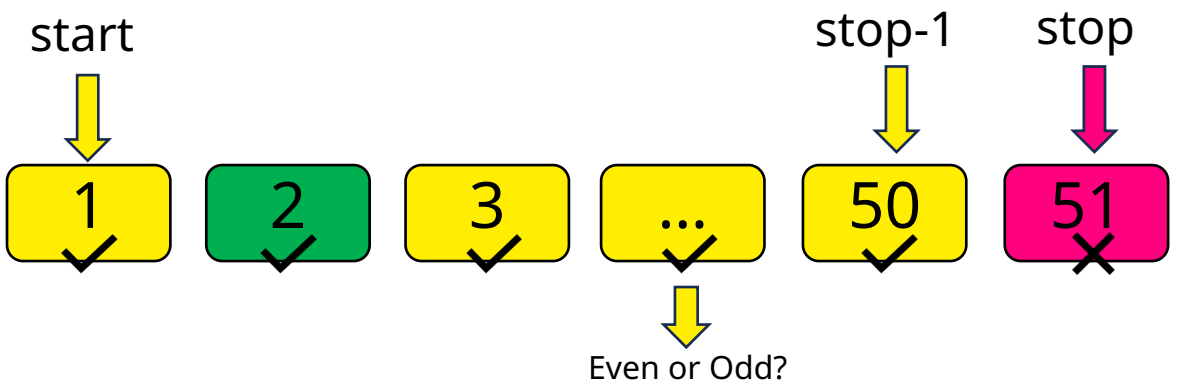
```
for i in range(_______,_______):
  if ___________:
    print(i)
```



Iteration Statement – for

8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

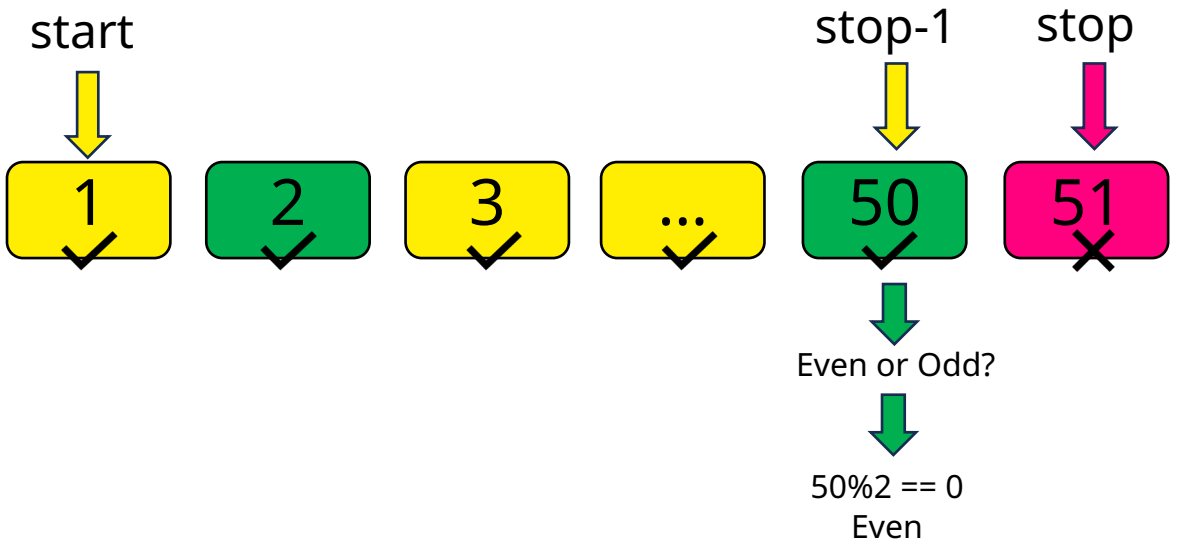
```
for i in range(_______,_______):
  if ___________:
    print(i)
```



Iteration Statement – for

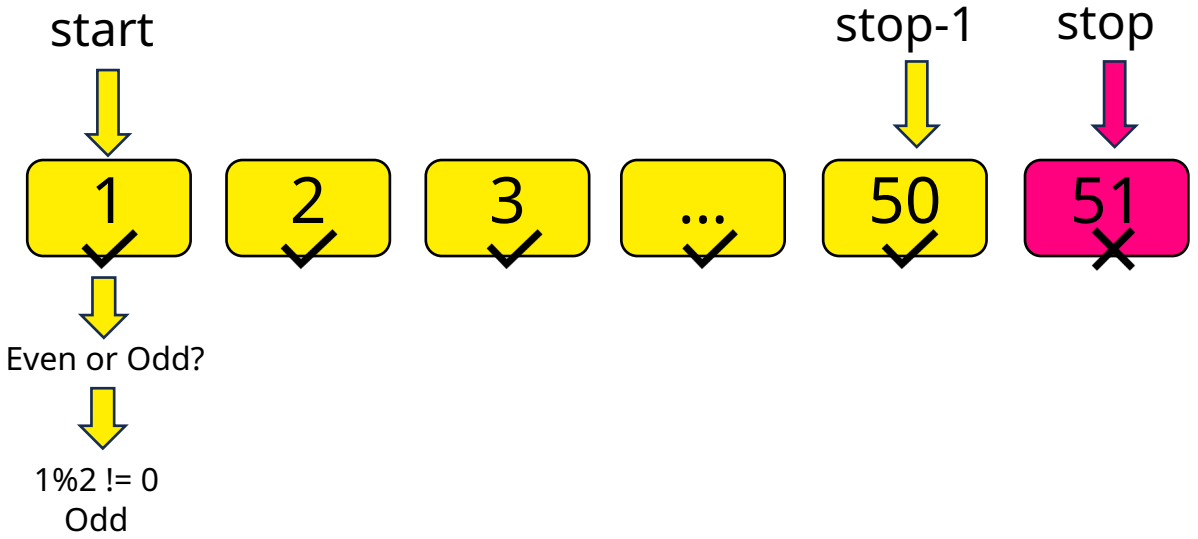
8. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50

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



Iteration Statement – for

9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

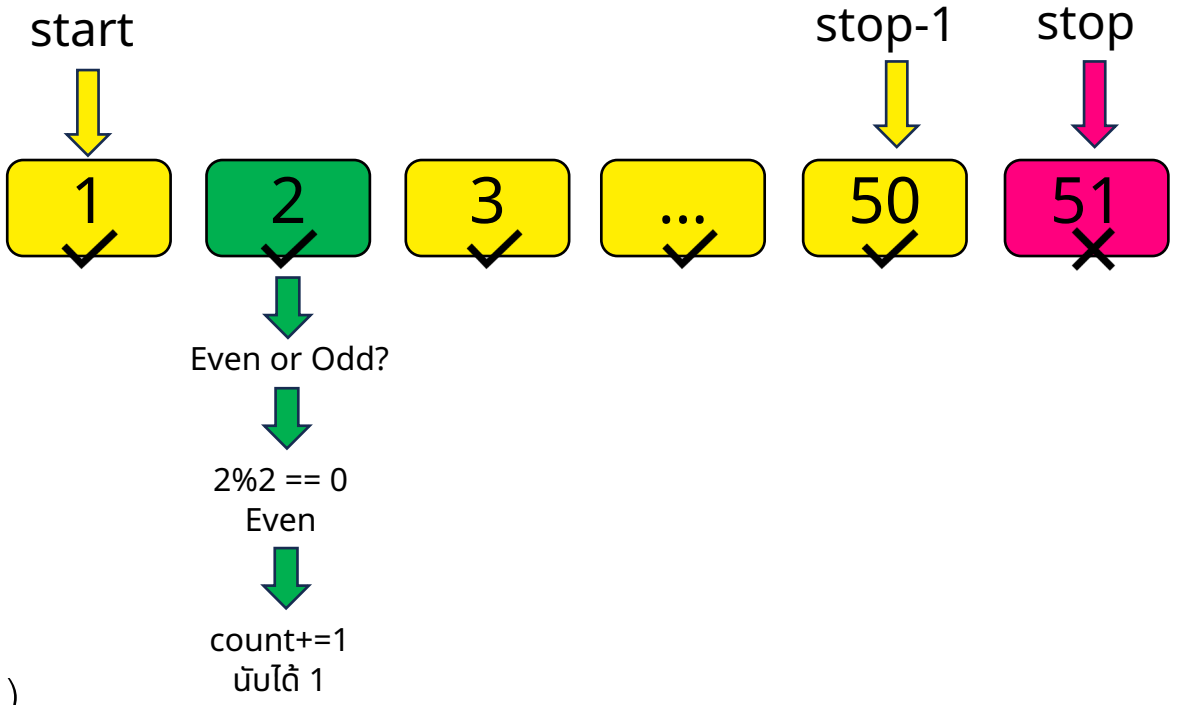



```
________ = ________
for i in range(1,50+1):
  if i%2==0:
    print(i)
      ________ = ________

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

Iteration Statement – for

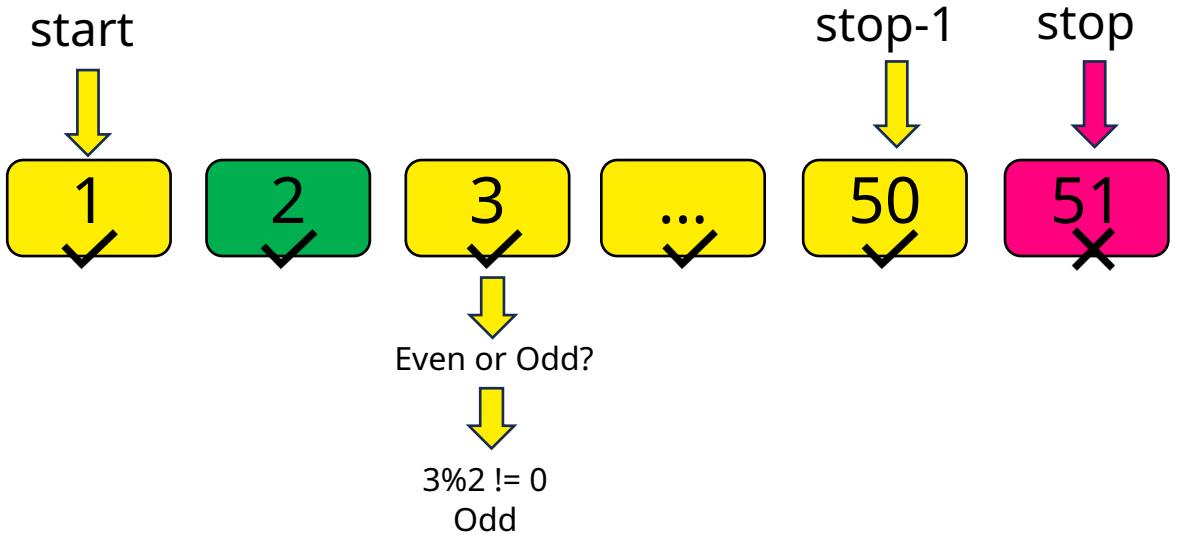
9. การโปรแกรมเพื่อแสดงผลจำนวนที่เป็นจำนวนคู่ (Even) ตั้งแต่ 1 ถึง 50 และสรุปว่ามีทั้งหมดกี่จำนวน

```
________ = ________
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

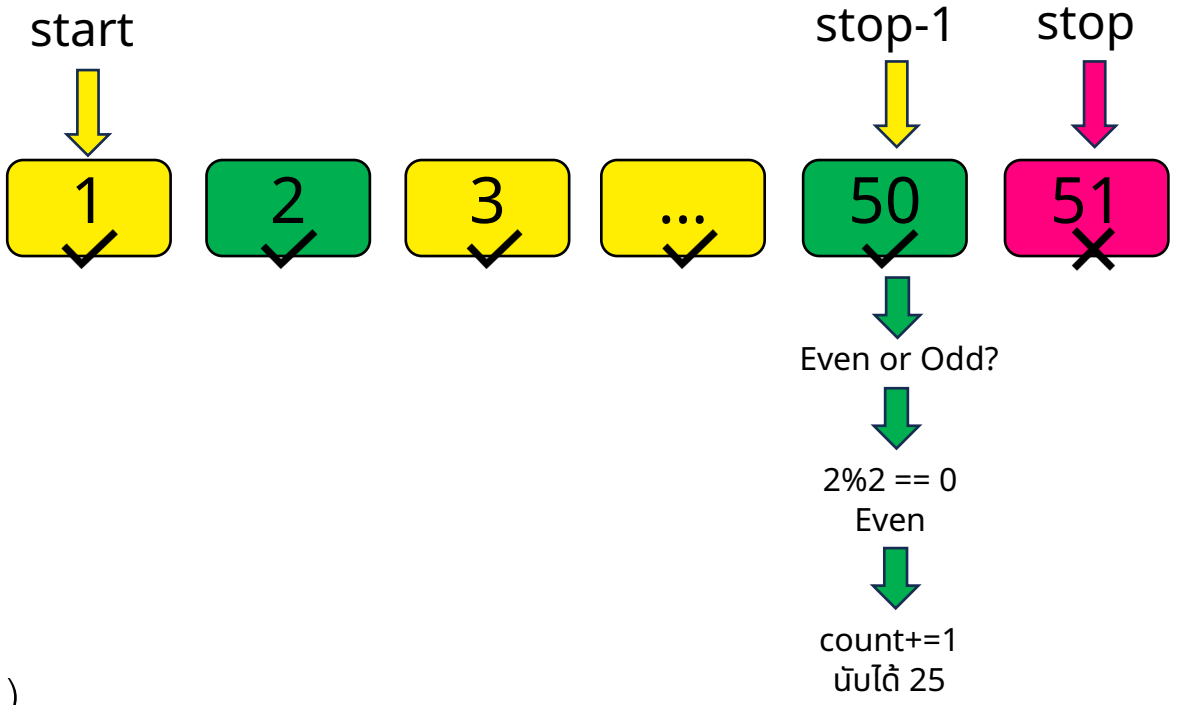



```
________ = ________
for i in range(1,50+1):
  if i%2==0:
    print(i)
    ________ = ________

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

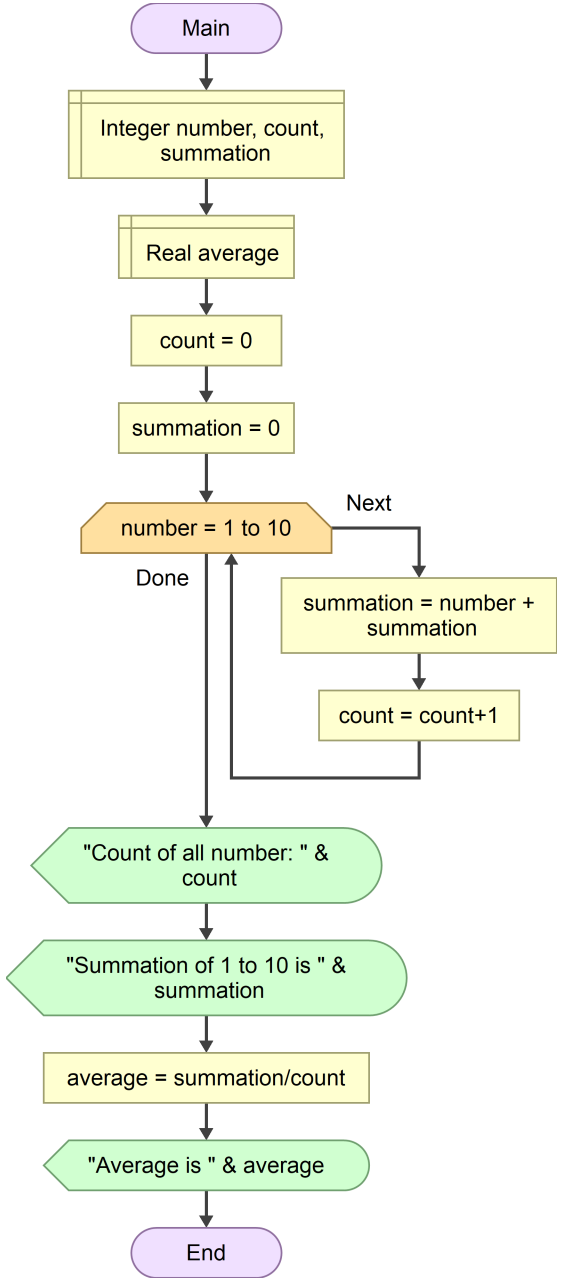
```
summation = 0
for i in range(2, 20+1, 2):
  summation = summation+i

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

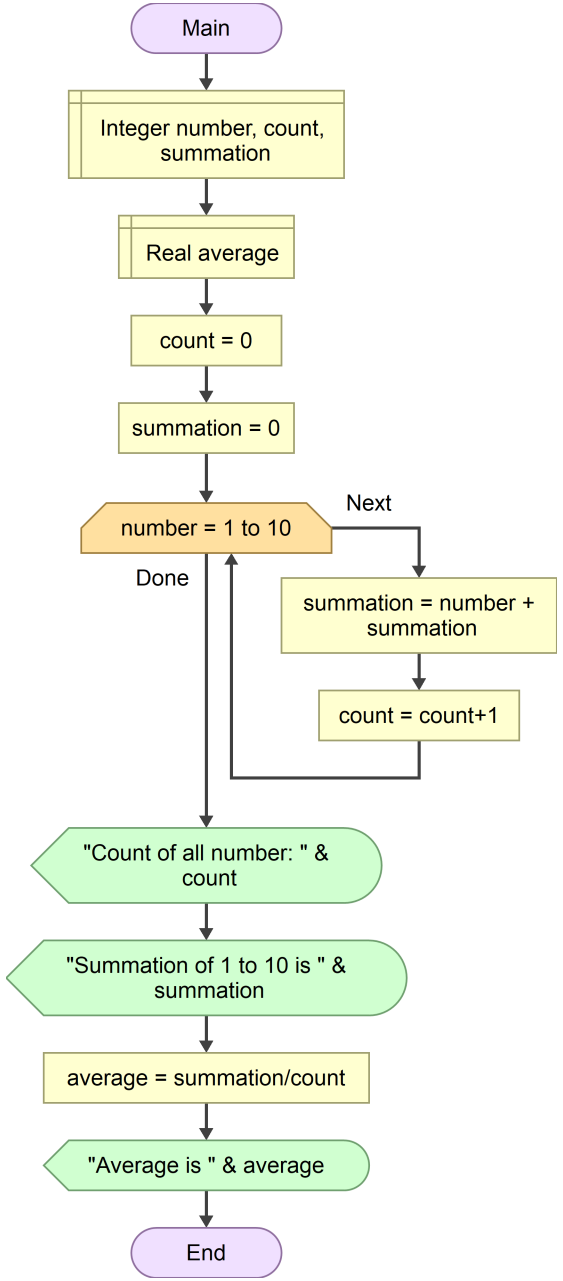
```
summation = _____
_______   = _____
_______   = _____
for i in range(___,____):
  summation = ___________________
  count = _________________

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



Iteration Statement – for

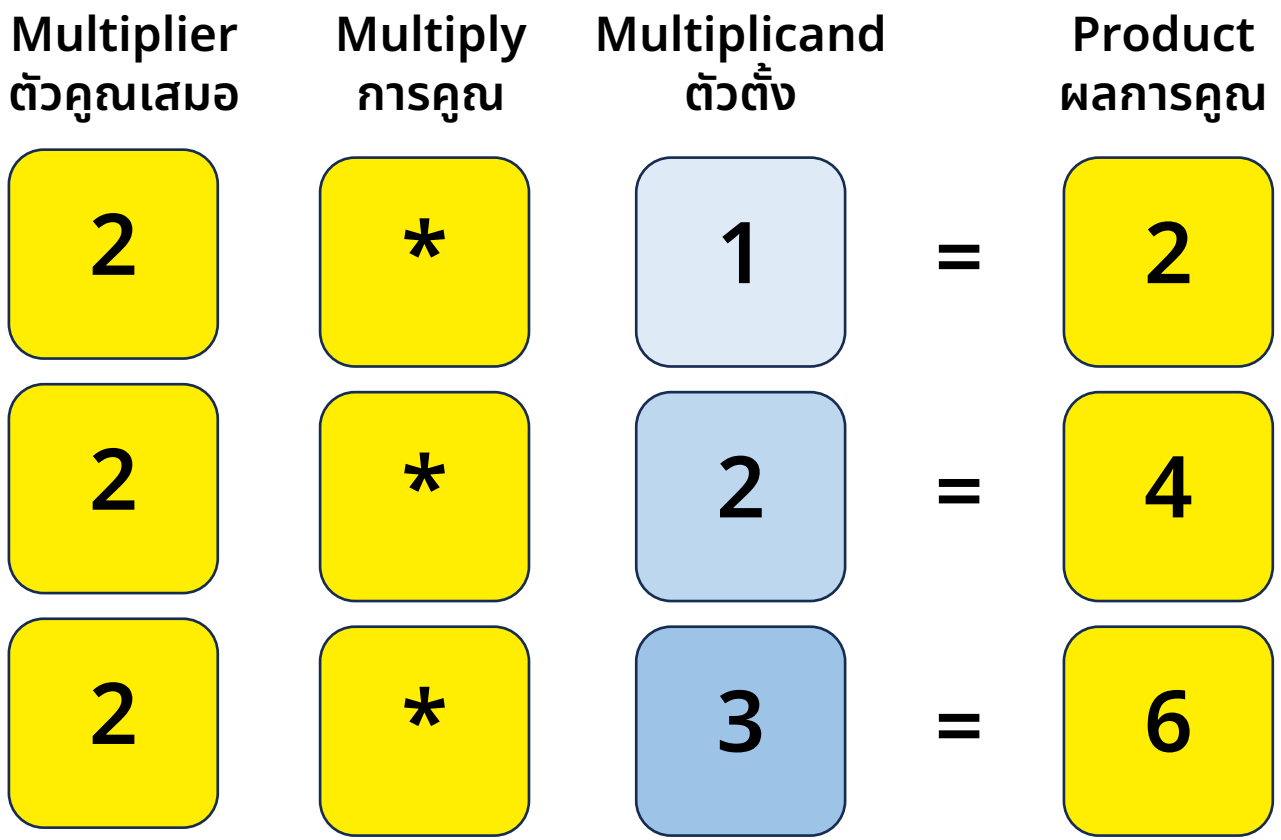
12. การโปรแกรมเพื่อหาค่าเฉลี่ย (average) ของจำนวนตั้งแต่ 1 ถึง 10 โดยค่าเฉลี่ยมาจาก ผลรวมของจำนวนทั้งหมด หารด้วย จำนวนทั้งหมด หรือ average = summation/count

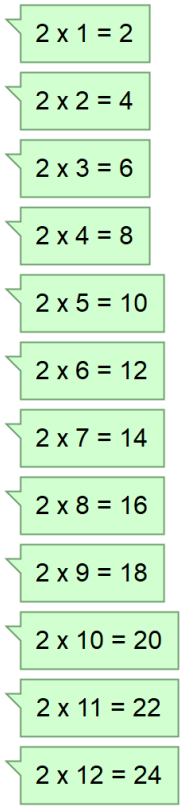
```
summation = 0
count = 0
average = 0
for i in range(1,10+1):
  summation = summation+i
  count = count+1

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



Iteration Statement – for

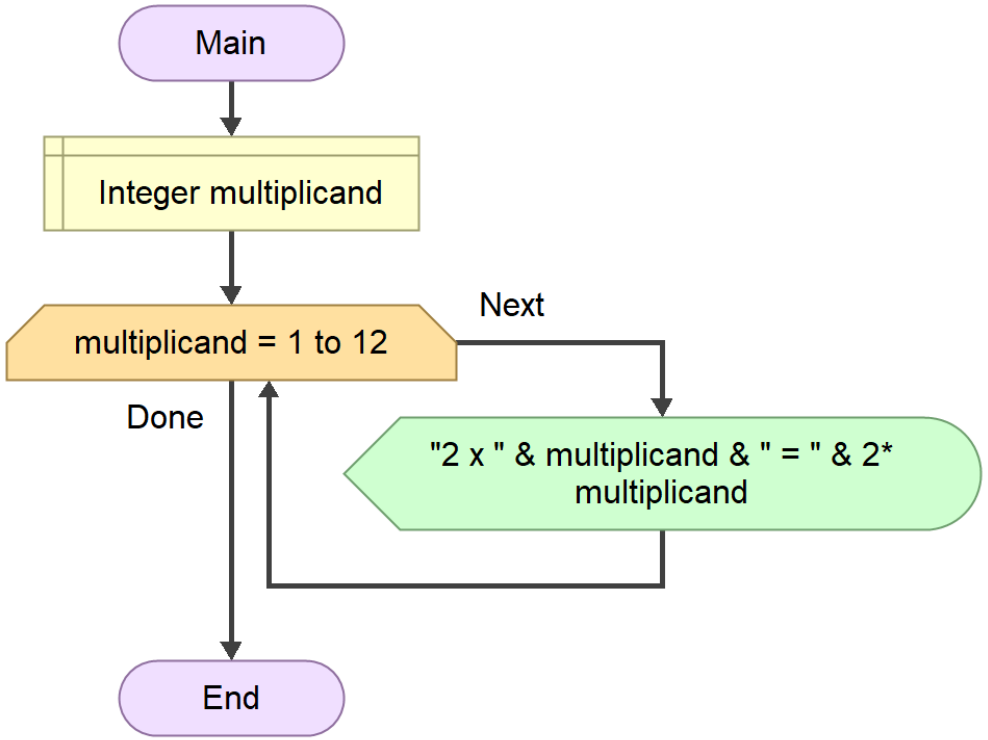
13. แสดงสูตรคูณแม่ 2 โดยเริ่มตั้งแต่ 2x1=2 ถึง 2x12=24 ด้วย for loop





Iteration Statement – for

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



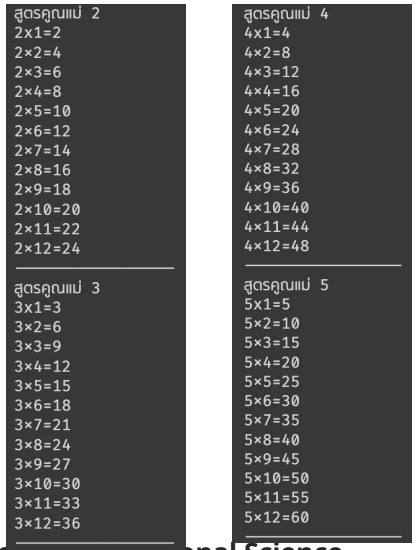

```
for i in range(1,12+1):
    print(f"2 x {i} = {2*i}")
```

```
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

สูตรคูณแม่ 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("-----------------")
```

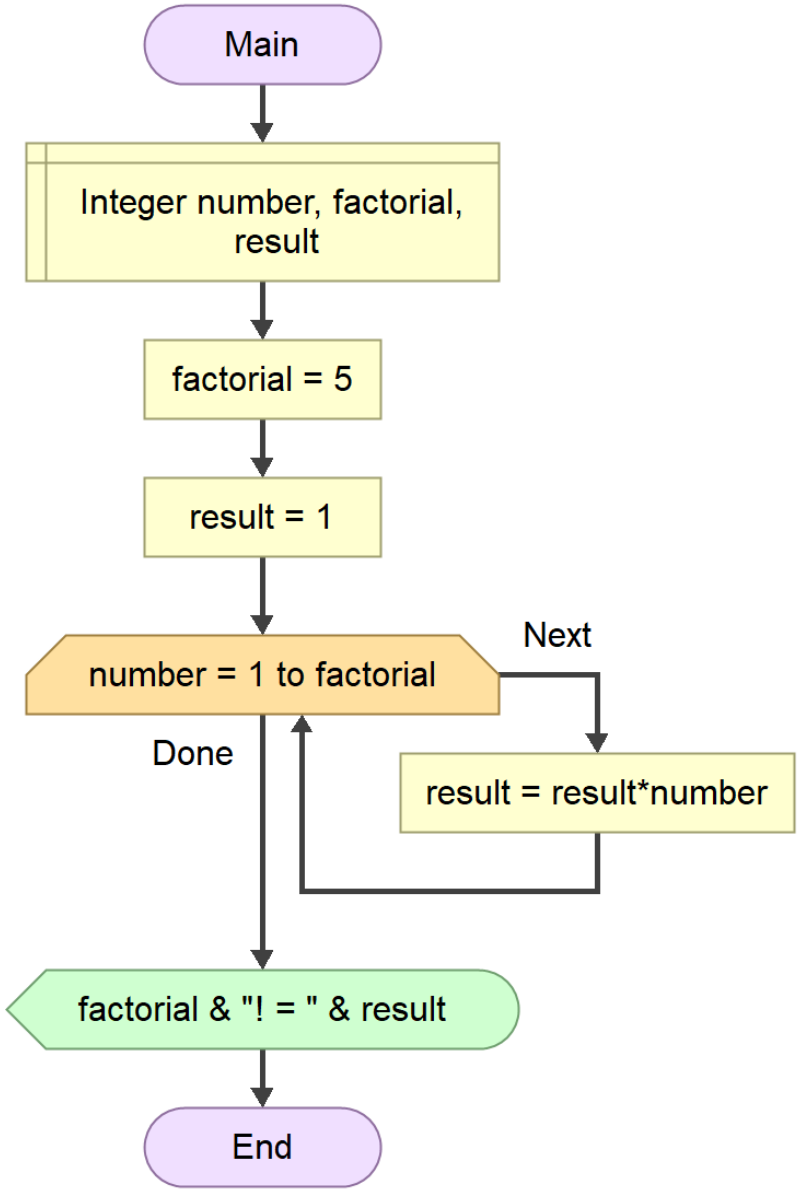


Iteration Statement – for

14. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5!
Factorial (n!) คือ ผลคูณของจำนวนเต็มบวก ตั้งแต่ 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 ผลคูณของจำนวนเต็มบวก ตั้งแต่ 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}")
```

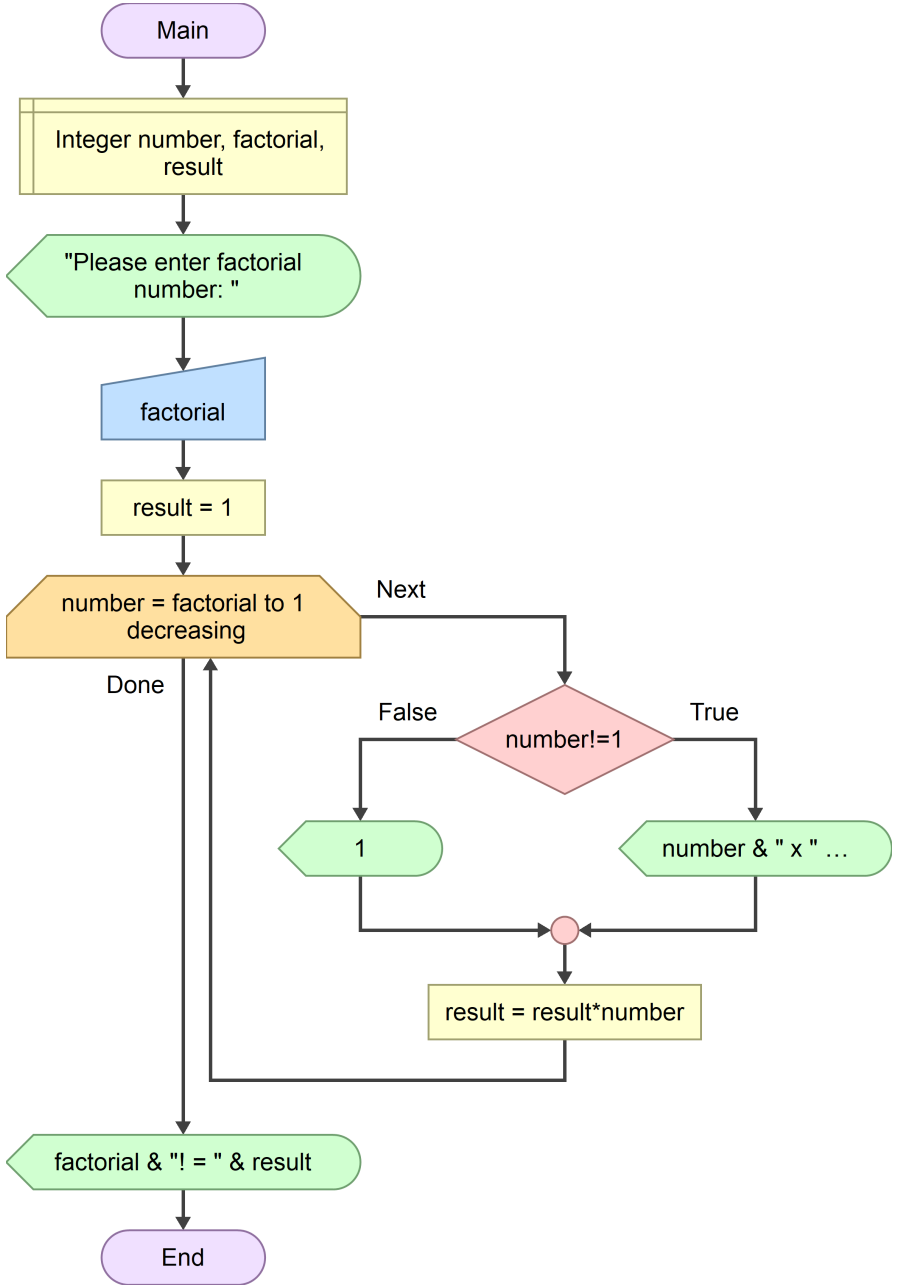
```
5! = 120
```

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

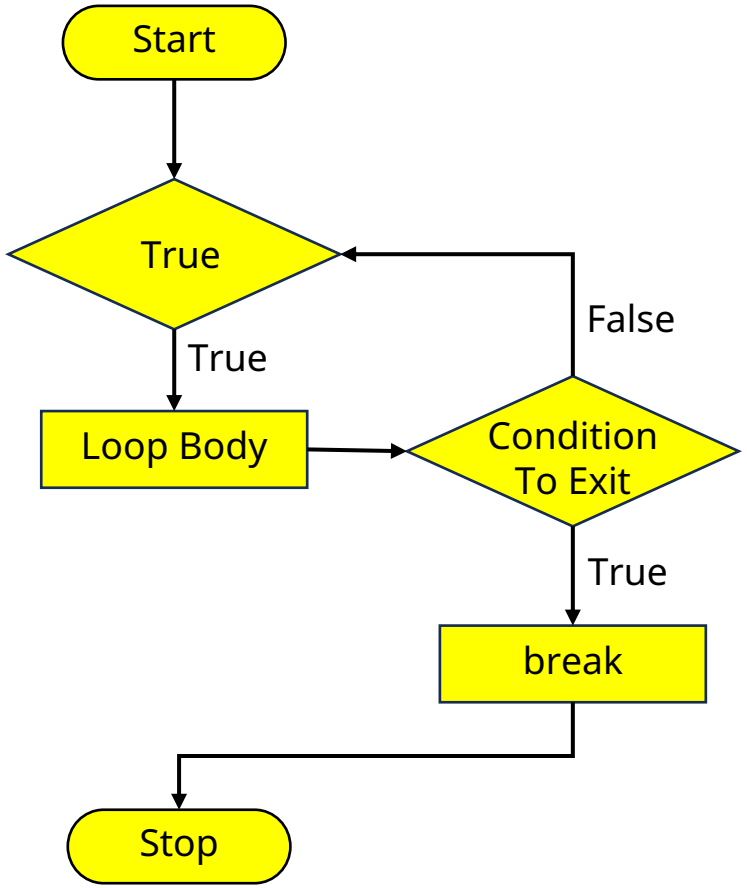
15. การโปรแกรมเพื่อหาค่า Factorial ของ 5 หรือ 5! โดยแสดงเป็น 5! = 5 x 4 x 3 x 2 x 1 = 120

```
# 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 เหมือนกับภาษาอื่นๆ 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) กรณีกรอกผิดให้ทำการถามใหม่วนซ้ำไป จนกว่าจะตอบถูก 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 บทที่ 4 Iteration Statement (3 คะแนน)

1. เขียนโปรแกรมเพื่อแสดงจำนวนตั้งแต่ 1 ถึง 10,000 ที่หาร 3 และหาร 5 ลงตัว และสรุปด้วยว่ามีกี่จำนวน
2. เขียนโปรแกรมหาค่าเฉลี่ยของจำนวนตั้งแต่ 500 ถึง 1,001
3. เขียนโปรแกรมสูตรคูณแม่ 2 ถึง สูตรคูณแม่ 12 ในการรันโปรแกรมครั้งเดียว
4. เขียนโปรแกรมรับค่ารหัสผ่าน
   หากรหัสผ่านไม่ตรงตามที่กำหนดไว้ ให้แจ้งว่า **รหัสผ่านผิด** และวนรับค่ารหัสผ่านใหม่
   หากรหัสผ่านตรงตามที่กำหนดไว้ ให้แจ้งว่า **รหัสผ่านถูกต้อง** และหยุดรับรหัสผ่าน