


Class period 3

บทที่ 3 โปรแกรมวนซ้ำและการเงื่อนไขในภาษาไพธอน
Function_Loop_Condition 2


1



ตัวอย่าง LOOP การวนซ้ำ

- ```
for i in 'Thanapong':
 print(f'{i} -> /')
```
- หมายความว่า ใหวนลูปอ่านค่าสมาชิกใน string 'Thanapong' โดยแทนค่าสมาชิกที่อ่านทีละตัวด้วยตัวแปร i
- ภายในลูป นำตัวแปร i ไปใน f '{i}' -> '/' และ print string


2



### ตัวอย่าง LOOP การวนซ้ำ

- ผลลัพธ์ที่ได้ จะเห็นว่า loop ทำการ print string ปกติ ตามค่า i คือตัวอักษรที่อยู่ใน string ที่ละตัวตามลำดับจากตัวที่ 0 ไปจนถึงลำดับสุดท้าย
- ```
T -> /  
h -> /  
a -> /  
n -> /  
a -> /  
p -> /  
o -> /  
n -> /  
g -> /
```


3



การสั่ง loop ให้ print แบบไม่เว้นบรรทัด

- โดยคำสั่ง print จะมี key argument และค่า default สำหรับตัวมันเองคือ end='\n'
- ```
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)
```
- Prints the values to a stream, or to sys.stdout by default.
- Optional keyword arguments:
- file: a file-like object (stream); defaults to the current sys.stdout.
- sep: string inserted between values, default a space.
- end: string appended after the last value, default a newline.
- flush: whether to forcibly flush the stream.


4



### ตัวอย่างการสั่ง loop ให้ print แบบไม่เว้นบรรทัด

- end='\n'
- \n ในภาษา python สำหรับการพิมพ์ string หมายถึง การเว้นบรรทัด
- ถ้าต้องการ print แบบไม่เว้นบรรทัด ให้ใส่ input กำหนดค่า end ด้วยค่าที่เราต้องการ
- ```
for i in 'Thanapong':  
    print(f'{i} -> / ', end = '')
```
- กำหนด end = '' คือไม่ให้ใส่เลย ผลลัพธ์จะได้
- ```
T -> / h -> / a -> / n -> / a -> / p -> / o -> / n -> / g -> /
```

5



### ตัวอย่างการสั่ง loop ให้ print แบบไม่เว้นบรรทัด

- ```
for i in 'Thanapong':  
    print(f'{i} -> / ', end = ',')
```
- กำหนด end = ',' คือ ใส่สัญลักษณ์ , ผลลัพธ์จะได้
- ```
T -> / , h -> / , a -> / , n -> / , a -> / , p -> / , o -> / , n -> / , g -> / ,
```

6

range() การสร้าง list ตัวเลขแบบอัตโนมัติ

- `range()` คือคำสั่งที่ใช้สร้าง list ของตัวเลข เช่น
  - `range_output = range(5)`
  - `print(list(range5_output))`
- หมายความว่า โค้ดสร้าง list ตัวเลขจำนวน 5 ตัว เริ่มจาก 0 และเก็บไว้ในตัวแปร `range5_output`
- จากนั้น `print` ตัวแปร `range5_output` ในรูปแบบของ list
- ผลลัพธ์จะได้
  - `[0, 1, 2, 3, 4]`

7

ตัวอย่างการใช้งาน range() สร้าง list ตัวเลขในการวนลูป

- [illegible]

8

### Key argument ของ range()

- `range(stop)` -> range object
- `range(start, stop[, step])` -> range object
- Return an object that produces a sequence of integers from start (inclusive) to stop (exclusive) by step. `range(i, j)` produces `i, i+1, i+2, ..., j-1`. start defaults to 0, and stop is omitted. `range(4)` produces 0, 1, 2, 3. These are exactly the valid indices for a list of 4 elements. When step is given, it specifies the increment (or decrement).
- range() สามารถกำหนดตัวเลขที่ต้องการ เริ่ม(start), หยุด(stop) และ step ได้

9

ตัวอย่างการใช้งาน range() ด้วย key argument start, stop, step

- `list(range(1,11))`
  - หมายถึงว่า ใช้ `range()` กำหนดให้สร้าง list ตัวเลข เริ่มจาก 1 ถึง 10
  - ผลลัพธ์จะได้
    - `[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`
- `list(range(-3,20,4))`
  - หมายถึงว่า ใช้ `range()` กำหนดให้สร้าง list ตัวเลข เริ่มจาก -3 ถึง 19 และให้ `step=4`
  - ผลลัพธ์จะได้
    - `[-3, 1, 5, 9, 13, 17]`

10

loop in loop

- สามารถสร้าง loop ภายใน loop ได้
- โดยลำดับการทำงาน จะทำงานตาม การเว้นวรรค (indent) โดยส่วนประมวลผลใน loop1 จะทำงานก่อน
- จากนั้นเมื่อมีการเขียน loop2 ภายใน loop1 loop2 จะถูกนับว่าเป็นส่วนประมวลผลของ loop1
- ดังนั้น loop2 จะทำงานวนซ้ำเพื่อตรวจสอบเงื่อนไขจนจบการรอบก่อน ถึงนับเป็นวนซ้ำ 1 รอบของ loop1

11

ตัวอย่าง loop in loop

- for loop1 in range(2,5): # (2, 3, 4)
  - print(f'now loop1 = {loop1}')
  - for loop2 in range(1,13):
  - print(loop1, ' x ', loop1, ' = ', loop1 \* loop2)
  - print('the inner loop is end')
- 
- หมายความว่า ไหวนลูปอ่านค่าสมาชิกใน range(2,5) ซึ่งคือ [2, 3, 4] ที่ละตัวแทนค่าด้วยตัวแปร loop1
  - ภายในลูปที่ print(f'now loop1 = {loop1}')
  - โดยลูปที่ 2 ไหวนลูปอ่านค่าสมาชิกใน range(1,13) ซึ่งคือ list 1 ถึง 12 ที่ละตัวแทนค่าด้วยตัวแปร loop2
  - ภายในลูปที่ 2 ที่ print(loop1, ' x ', loop1, ' = ', loop1 \* loop2) รวนลูปที่ 2 จนครบ 12
  - เมื่อจบลูปที่ 2 ที่ print('the inner loop is end') จากนั้นก็จะกลับไปวนลูปใหม่นวนถัดจากตัวแปร range(2,5)

12

### ตัวอย่าง loop in loop ผลลัพธ์จะได้

|                         |                         |                         |
|-------------------------|-------------------------|-------------------------|
| • now name1 = 2         | • now name1 = 3         | • now name1 = 4         |
| • 2 x 1 = 2             | • 3 x 1 = 3             | • 4 x 1 = 4             |
| • 2 x 2 = 4             | • 3 x 2 = 6             | • 4 x 2 = 8             |
| • 2 x 3 = 6             | • 3 x 3 = 9             | • 4 x 3 = 12            |
| • 2 x 4 = 8             | • 3 x 4 = 12            | • 4 x 4 = 16            |
| • 2 x 5 = 10            | • 3 x 5 = 15            | • 4 x 5 = 20            |
| • 2 x 6 = 12            | • 3 x 6 = 18            | • 4 x 6 = 24            |
| • 2 x 7 = 14            | • 3 x 7 = 21            | • 4 x 7 = 28            |
| • 2 x 8 = 16            | • 3 x 8 = 24            | • 4 x 8 = 32            |
| • 2 x 9 = 18            | • 3 x 9 = 27            | • 4 x 9 = 36            |
| • 2 x 10 = 20           | • 3 x 10 = 30           | • 4 x 10 = 40           |
| • 2 x 11 = 22           | • 3 x 11 = 33           | • 4 x 11 = 44           |
| • 2 x 12 = 24           | • 3 x 12 = 36           | • 4 x 12 = 48           |
| • the inner loop is end | • the inner loop is end | • the inner loop is end |

13

### loop in function

- สามารถวนลูปในฟังก์ชันได้ โดยใช้ การเว้นวรรค (indent) กำหนดลำดับการทำงาน
- def print\_feelings(I, friendS, feeling = 'คิดถึง':
  - for friend in friendS:
    - print(f'{I} {feeling} {friend}')
- หมายความว่า กำหนดฟังก์ชันชื่อ print\_feelings
- มี input 3 อย่าง คือ I, friendS, feeling = 'คิดถึง' (default)
- ภายใน def ให้นิยามค่าสมาชิกในตัวแปร friendS (list) ที่จะตัวแปรค่าด้วยตัวแปร friend
- ภายในลูป print(f'{I} {feeling} {friend}')

14

### ตัวอย่างการใช้งาน loop in function

- list\_friends = ['พี่ตูน', 'ดาญ์', 'แดงโม', 'ฝน', 'พีเจพี', 'น้องแอมป์']
- print\_feelings('เจมส์', list\_friends)
- กำหนดตัวแปร list\_friends เป็นข้อมูล list มีสมาชิก ['พี่ตูน', 'ดาญ์', 'แดงโม', 'ฝน', 'พีเจพี', 'น้องแอมป์']
- ใช้งานฟังก์ชัน print\_feelings('เจมส์', list\_friends)
- ชื่อฟังก์ชันตามตัวอย่างที่ต้องการ input ตามลำดับตัวแปรที่กำหนดไว้ตอนสร้างฟังก์ชัน 'เจมส์' = I, list\_friends = friendS และตัวแปร feeling ไม่ได้ input ค่าดังนั้นจะถูกใช้ค่า default คือ 'คิดถึง'

15

### ตัวอย่างการใช้งาน loop in function

- ผลลัพธ์ของ print\_feelings('เจมส์', list\_friends) ไม่มี input ค่าตัวแปร feeling จะได้
- เจมส์ คิดถึง พี่ตูน
- เจมส์ คิดถึง ดาญ์
- เจมส์ คิดถึง แแดงโม
- เจมส์ คิดถึง ฝน
- เจมส์ คิดถึง พีเจพี
- เจมส์ คิดถึง น้องแอมป์

```
list_friends = ['พี่ตูน', 'ดาญ์', 'แดงโม', 'ฝน', 'พีเจพี', 'น้องแอมป์']
def print_feelings(I, friendS, feeling = 'คิดถึง'):
 for friend in friendS:
 print(f'{I} {feeling} {friend}')
```

16

### ตัวอย่างการใช้งาน loop in function

- ผลลัพธ์ของ print\_feelings('เจมส์', list\_friends, 'รัก') จะได้
- เจมส์ รัก พี่ตูน
- เจมส์ รัก ดาญ์
- เจมส์ รัก แแดงโม
- เจมส์ รัก ฝน
- เจมส์ รัก พีเจพี
- เจมส์ รัก น้องแอมป์

```
list_friends = ['พี่ตูน', 'ดาญ์', 'แดงโม', 'ฝน', 'พีเจพี', 'น้องแอมป์']
def print_feelings(I, friendS, feeling = 'คิดถึง'):
 for friend in friendS:
 print(f'{I} {feeling} {friend}')
```

17

### conditional คือ เงื่อนไข / ประโยคเงื่อนไข

- if condition1 :
  - do something
- elif condition2 : #elif #else if
  - do another thing
- else:
  - do ...
- สามารถกำหนดเงื่อนไขก่อนเข้าการทำงานของส่วนประมวลผล โดยผ่านเงื่อนไขของ if condition1 ได้ตามประมวลผลภายใน if จะทำงาน
- ถ้าไม่ผ่านจะไปเงื่อนไขถัดไป elif condition2 ตามลำดับ (elif สามารถมีได้มากกว่า 1)
- แต่ถ้าหากไม่ผ่านเงื่อนไขใดเลย ได้ประมวลผลของ else จะทำงาน

18

### ตัวอย่าง Loop แบบปกติ

```

• list_friends = ['พี่ตูน', 'ตูน', 'น้องไม้ม', 'พี่เจี๊ยบ', 'น้องแอมป์']

• for name1 in list_friends :
 for name2 in list_friends :
 print(name1, 'รัก', name2)

• วนลูป for ปกติเพื่อ print(name1, 'รัก', name2) เมื่อได้ผลลัพธ์ออกมา สังเกตว่าจะมีชื่อซ้ำกันรักกันเอง

```

18

19

### ผลลัพธ์จะได้

|                        |                         |                           |
|------------------------|-------------------------|---------------------------|
| • พี่ตูน รัก พี่ตูน    | • แดงไม้ม รัก พี่ตูน    | • พี่เจี๊ยบ รัก พี่ตูน    |
| • พี่ตูน รัก ตูน       | • แดงไม้ม รัก ตูน       | • พี่เจี๊ยบ รัก ตูน       |
| • พี่ตูน รัก แดงไม้ม   | • แดงไม้ม รัก แดงไม้ม   | • พี่เจี๊ยบ รัก แดงไม้ม   |
| • พี่ตูน รัก พี่ตูน    | • แดงไม้ม รัก พี่ตูน    | • พี่เจี๊ยบ รัก พี่ตูน    |
| • พี่ตูน รัก พี่เจี๊ยบ | • แดงไม้ม รัก พี่เจี๊ยบ | • พี่เจี๊ยบ รัก พี่เจี๊ยบ |
| • พี่ตูน รัก น้องแอมป์ | • แดงไม้ม รัก น้องแอมป์ | • พี่เจี๊ยบ รัก น้องแอมป์ |
| • ตูน รัก พี่ตูน       | • พี่ตูน รัก พี่ตูน     | • น้องแอมป์ รัก พี่ตูน    |
| • ตูน รัก ตูน          | • พี่ตูน รัก ตูน        | • น้องแอมป์ รัก ตูน       |
| • ตูน รัก แดงไม้ม      | • พี่ตูน รัก แดงไม้ม    | • น้องแอมป์ รัก แดงไม้ม   |
| • ตูน รัก พี่ตูน       | • พี่ตูน รัก พี่ตูน     | • น้องแอมป์ รัก พี่ตูน    |
| • ตูน รัก พี่เจี๊ยบ    | • พี่ตูน รัก พี่เจี๊ยบ  | • น้องแอมป์ รัก พี่เจี๊ยบ |
| • ตูน รัก น้องแอมป์    | • พี่ตูน รัก น้องแอมป์  | • น้องแอมป์ รัก น้องแอมป์ |

19

20

### ตัวอย่างการใช้งาน conditional

```

• for name1 in list_friends:
 for name2 in list_friends:
 if name1 != name2: # ไม่ให้พิมพ์ตัวเองซ้ำ
 print(name1, 'รัก', name2)

• สามารถเพิ่มเงื่อนไข if name1 != name2: ไปส่วนประมวลผลภายในลูป เมื่อผ่านเงื่อนไขนี้จึงจะไปทำงานส่วนประมวลผลภายใน if คือ print(name1, 'รัก', name2)

• ดังนั้น ตามเงื่อนไข name1 ไม่เท่ากับ name2 โค้ดจะปรี้นแต่ชื่อที่ไม่ซ้ำกันเท่านั้น

```

20

21

### ผลลัพธ์จะได้

|                        |                         |                           |
|------------------------|-------------------------|---------------------------|
| • พี่ตูน รัก ตูน       | • แดงไม้ม รัก พี่ตูน    | • พี่เจี๊ยบ รัก พี่ตูน    |
| • พี่ตูน รัก แดงไม้ม   | • แดงไม้ม รัก ตูน       | • พี่เจี๊ยบ รัก ตูน       |
| • พี่ตูน รัก พี่ตูน    | • แดงไม้ม รัก พี่ตูน    | • พี่เจี๊ยบ รัก แดงไม้ม   |
| • พี่ตูน รัก พี่เจี๊ยบ | • แดงไม้ม รัก พี่เจี๊ยบ | • พี่เจี๊ยบ รัก พี่ตูน    |
| • พี่ตูน รัก น้องแอมป์ | • แดงไม้ม รัก น้องแอมป์ | • พี่เจี๊ยบ รัก น้องแอมป์ |
| • ตูน รัก พี่ตูน       | • พี่ตูน รัก พี่ตูน     | • น้องแอมป์ รัก พี่ตูน    |
| • ตูน รัก แดงไม้ม      | • พี่ตูน รัก ตูน        | • น้องแอมป์ รัก ตูน       |
| • ตูน รัก พี่ตูน       | • พี่ตูน รัก แดงไม้ม    | • น้องแอมป์ รัก แดงไม้ม   |
| • ตูน รัก พี่เจี๊ยบ    | • พี่ตูน รัก พี่เจี๊ยบ  | • น้องแอมป์ รัก พี่เจี๊ยบ |
| • ตูน รัก น้องแอมป์    | • พี่ตูน รัก น้องแอมป์  | • น้องแอมป์ รัก พี่เจี๊ยบ |

21

22

### ข้อจำกัดของการใช้งาน conditional

```

• for name1 in list_friends:
 for name2 in list_friends:
 if name1 == name2:
 'do nothing'
 else:
 print(name1, 'รัก', name2)

• ภายในเงื่อนไข if เป็นต้องมีส่วนประมวลผล ไม่สามารถปล่อยว่างไว้ได้

```

22


23

### สัญลักษณ์ที่ใช้ใน conditional

- == เท่ากับ
- != ไม่เท่ากับ
- >= มากกว่าหรือเท่ากับ ใช้ในการตรวจสอบตัวเลข
- <= น้อยกว่าหรือเท่ากับ ใช้ในการตรวจสอบตัวเลข
- < น้อยกว่า ใช้ในการตรวจสอบตัวเลข
- > มากกว่า ใช้ในการตรวจสอบตัวเลข

23

24




### Ture False ใน conditional

- หากหลังเงื่อนไข(if) เป็น True จะทำงานส่วนประมวลผลภายใน if
- if True:
  - print('Yes')
  - ผลลัพธ์จะได้
- Yes
- หากหลังเงื่อนไข(if) เป็น False โค้ดจะไม่เข้าไปทำงานส่วนประมวลผลภายใน if เลย ดังนั้น
- if False:
  - print('No')
  - ผลลัพธ์จะได้
- ไม่ป้อนอะไรออกมาเลย เพราะโค้ดไม่เข้าไปทำงานส่วนประมวลผลภายใน if หรือก็คือปล่อยผ่าน

24

25




### ตัวอย่าง Ture False ใน conditional

- 'ที่เยี่ยม' == 'ที่เจ็ท'
  - ผลลัพธ์จะได้
- False
- 'ที่เยี่ยม' != 'ที่เจ็ท'
  - ผลลัพธ์จะได้
- True

25

26



### Homework class period 3

- เขียน function วนลูปเกรด โดยที่ input เป็นคะแนน(0-100) , output เป็นเกรด (F-A)
- วนลูปเกรด input = [1,50,65,90,101,-5,49]
- กำหนด
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 0 แต่น้อยกว่า 50 จะได้เกรด F
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 50 แต่น้อยกว่า 55 จะได้เกรด D
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 55 แต่น้อยกว่า 60 จะได้เกรด D+
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 60 แต่น้อยกว่า 65 จะได้เกรด C
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 65 แต่น้อยกว่า 70 จะได้เกรด C+
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 70 แต่น้อยกว่า 75 จะได้เกรด B
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 75 แต่น้อยกว่า 80 จะได้เกรด B+
- ถ้าคะแนนอยู่ระหว่างมากกว่าหรือเท่ากับ 80 จะได้เกรด A
- แต่ถ้าคะแนนจะต้องไม่ต่ำกว่า 0 และมากกว่า 100 ไม่งั้นจะเกิด error

26

27