Class period 3

บทที่ 3 โปรแกรมวนซ้ำและการใช้เงื่อนไขในภาษาไพธอน

Function_Loop_Condition 2

1

ตัวอย่าง LOOP การวนซ้ำ

```
for i in 'Thanapong':
    print(f'{i} -> /')
```

- หมายความว่า ให้วนลูปอ่านค่าสมาชิกใน string 'Thanapong' โดยแทนค่าสมาชิกที่อ่านทีละตัวด้วยตัวแปร i
- ภายในลูป นำตัวแปร i ไสใน f' {i} -> / ' และ print string

2

ตัวอย่าง LOOP การวนซ้ำ

- ผลลัพธ์จะได้ จะเห็นว่า loop ทำการ print string ปกติ ตามค่า i คือตัวอักษรที่อยู่ใน string ที่ละตัวตามลำดับจากลำดับที่ 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 = '' คือไม่ใส่อะไรเลย 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
- range5_output = range(5)
- print(list(range5_output))
- หมายความว่า ให้สร้าง 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

- for i in range(100):
- print('%s',end=' ')

- หมายความว่า ให้วนลูป 100 รอบ โดยอ่านค่าสมาชิกใน range(100) ซึ่งคือ list 0 ถึง 99
- ภายในลูปให้ print('%s',end=' ')
- ผลลัพธ์จะได้
- 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99

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))`
- หมายความว่า 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))`
- หมายความว่า ใช้ range() กำหนดให้สร้าง list ตัวเลข เริ่มจาก -3 ถึง 19 และให้ step=4
- ผลลัพธ์จะได้
- `[-3, 1, 5, 9, 13, 17]`

9

10

loop in loop

- สามารถสร้าง loop ภายใน loop ได้
- โดยลำดับการทำงาน จะทำงานตาม การเว้นวรรค (indent) โดยส่วนประธรมสมลงใน loop1 จะทำงานก่อน
- จากนั้นเมื่อมีการเขียน loop2 ภายใน loop1 loop2 จะถูกนับว่าเป็นส่วนประธรมสมลลของ loop1
- ดังนั้น loop2 จะทำงานวนลูปของตัวเองจนจบทุกรอบก่อน จึงนับเป็นวนลูป 1 รอบของ loop1

10

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 ให้วนลูปอ่านค่าสมาชิกใน range(2,5) ซึ่งคือ [2, 3, 4] ที่จะตัวแทนค่าด้วยตัวแปร loop1
- ภายในลูปให้ print(f'now loop1 = {loop1}') ตอด้วย ลูปที่ 2
- โดยลูปที่ 2 ให้วนลูปอ่านค่าสมาชิกใน range(1,13) ซึ่งคือ list 1 ถึง 12 ที่จะตัวแทนค่าด้วยตัวแปร loop2
- ภายในลูปที่ 2 ให้ print(loop1,' x ', loop1,' = ', loop1 * loop2) วนลูปที่ 2 จนครบ 12
- เมื่อจบลูปที่ 2 ให้ print('the inner loop is end') จากนั้นก็จะกลับไปวนลูปใหม่จนกว่าจะครบตาม range(2,5)

11

12

ตัวอย่าง loop in loop ผลลัพธ์จะได้

- now name1 = 2
- 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
- the inner loop is end

- now name1 = 3
- 3 x 1 = 3
- 3 x 2 = 6
- 3 x 3 = 9
- 3 x 4 = 12
- 3 x 5 = 15
- 3 x 6 = 18
- 3 x 7 = 21
- 3 x 8 = 24
- 3 x 9 = 27
- 3 x 10 = 30
- 3 x 11 = 33
- 3 x 12 = 36
- the inner loop is end

- now name1 = 4
- 4 x 1 = 4
- 4 x 2 = 8
- 4 x 3 = 12
- 4 x 4 = 16
- 4 x 5 = 20
- 4 x 6 = 24
- 4 x 7 = 28
- 4 x 8 = 32
- 4 x 9 = 36
- 4 x 10 = 40
- 4 x 11 = 44
- 4 x 12 = 48
- the inner loop is end

12

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 กำหนดฟังก์ชั่นชื่อ print_feelings
- มี input 3 อย่าง คือ (friendS,feeling = 'คิดถึง' (default)
- ภายใน def ให้วนลูปอ่านค่าสมาชิกในตัวแปร friendS (list) ที่จะตัวแทนค่าด้วยตัวแปร friend
- ภายในลูป print(f'{I} {feeling} {friend}')

13

14

ตัวอย่างการใช้งาน loop in function

- `list_friends = ['พี่บูม','ตายน์','แดงโม้','ฟ่น','พี่เจ๊ท','น้องเคนบ์']`
- `print_feelings('เจมส์',list_friends)`

- กำหนดตัวแปร list_friends เป็นข้อมูล list มีสมาชิก ['พี่บูม','ตายน์','แดงโม้','ฟ่น','พี่เจ๊ท','น้องเคนบ์']
- ใช้งานฟังก์ชั่น print_feelings('เจมส์',list_friends)
- ซึ่งฟังก์ชั่นตามด้วยค่าที่ต้องการ input ตามลำดับตัวแปรที่กำหนดไว้ตอนสร้างฟังก์ชั่น 'เจมส์' = I, list_friends = friendS และตัวแปร feeling ไม่ได้ input ค่าดังนั้นจะถูกใช้ค่า default คือ 'คิดถึง'

14

15

ตัวอย่างการใช้งาน loop in function

- ผลลัพธ์ของ `print_feelings('เจมส์',list_friends)` ไม่มี input ค่าตัวแปร feeling จะได้
- เจมส์ คิดถึง พี่บูม
- เจมส์ คิดถึง ตายน์
- เจมส์ คิดถึง แดงโม้
- เจมส์ คิดถึง ฟ่น
- เจมส์ คิดถึง พี่เจ๊ท
- เจมส์ คิดถึง น้องเคนบ์

```
list_friends = ['พี่บูม','ตายน์','แดงโม้','ฟ่น','พี่เจ๊ท','น้องเคนบ์']

def print_feelings(I,friendS,feeling = 'คิดถึง'):
  for friend in friendS:
        print(f'{I} {feeling} {friend}')
```

15

16

ตัวอย่างการใช้งาน loop in function

- ผลลัพธ์ของ `print_feelings('เจมส์',list_friends,'รัก')` จะได้
- เจมส์ รัก พี่บูม
- เจมส์ รัก ตายน์
- เจมส์ รัก แดงโม้
- เจมส์ รัก ฟ่น
- เจมส์ รัก พี่เจ๊ท
- เจมส์ รัก น้องเคนบ์

```
list_friends = ['พี่บูม','ตายน์','แดงโม้','ฟ่น','พี่เจ๊ท','น้องเคนบ์']

def print_feelings(I,friendS,feeling = 'คิดถึง'):
  for friend in friendS:
        print(f'{I} {feeling} {friend}')
```

16

17

conditional คือ เงื่อนไข /ประโยคเงื่อนไข

- `if condition1 :`
- `    do something`
- `elif condition2 :  #elif คือelse if`
- `    do another thing`
- `else:`
- `    do ...`

- สามารถกำหนดเงื่อนไขก่อนเข้าการทำงานของส่วนประธรมสมลผล โดยถ้าผ่านเงื่อนไขของ if condition1 ให้ประธรมสมลผลภายใน if จะทำงาน
- ถ้าไม่ผ่านจะไปเงื่อนไขถัดไป elif condition2 ตามลำดับ (elif สามารถมีได้มากกว่า 1)
- และถ้าหากไม่ผ่านเงื่อนไขใดเลย โค้ดประธรมสมลผลของ else จะทำงาน

17

18

ตัวอย่าง Loop แบบปกติ

- `list friends = ['พี่บูม','ตายน์','แตงโม','ฝน','พี่เจ๊ท','น้องแคนบ์']`
- `for name1 in list_friends :`
  - `for name2 in list_friends :`
    - `print(name1, 'รัก',name2)`
- วนลูป for ปกติเพื่อ print(name1,'รัก',name2) เมื่อได้ผลลัพธ์ออกมา ล้วนทวาระมีชื่อซ้ำกันรักกันเอง

18

19

ผลลัพธ์จะได้

- พี่บูม รัก พี่บูม
- พี่บูม รัก ตายน์
- พี่บูม รัก แตงโม
- พี่บูม รัก ฝน
- พี่บูม รัก พี่เจ๊ท
- พี่บูม รัก น้องแคนบ์
- ตายน์ รัก พี่บูม
- ตายน์ รัก ตายน์
- ตายน์ รัก แตงโม
- ตายน์ รัก ฝน
- ตายน์ รัก พี่เจ๊ท
- ตายน์ รัก น้องแคนบ์
- แตงโม รัก พี่บูม
- แตงโม รัก ตายน์
- แตงโม รัก แตงโม
- แตงโม รัก ฝน
- แตงโม รัก พี่เจ๊ท
- แตงโม รัก น้องแคนบ์
- ฝน รัก พี่บูม
- ฝน รัก ตายน์
- ฝน รัก แตงโม
- ฝน รัก ฝน
- ฝน รัก พี่เจ๊ท
- ฝน รัก น้องแคนบ์
- พี่เจ๊ท รัก พี่บูม
- พี่เจ๊ท รัก ตายน์
- พี่เจ๊ท รัก แตงโม
- พี่เจ๊ท รัก ฝน
- พี่เจ๊ท รัก พี่เจ๊ท
- พี่เจ๊ท รัก น้องแคนบ์
- น้องแคนบ์ รัก พี่บูม
- น้องแคนบ์ รัก ตายน์
- น้องแคนบ์ รัก แตงโม
- น้องแคนบ์ รัก ฝน
- น้องแคนบ์ รัก พี่เจ๊ท
- น้องแคนบ์ รัก น้องแคนบ์

19

20

ตัวอย่างการใช้งาน conditional

- `for name1 in list_friends:`
  - `for name2 in list_friends:`
    - `if name1 != name2:` ตัวเดียวชื่อคนเดียวกันห้า
    - `print(name1,'รัก',name2)`
- สามารถเพิ่มเงื่อนไข if name1 != name2: ไปส่วนประมวลผลภายในลูป เมื่อผ่านเงื่อนไขนี้จึงจะไปทำงานส่วนประมวลผลภายใน # คือ print(name1,'รัก',name2)
- ดังนั้น ตามเงื่อนไข name1 ไม่เท่ากับ name2 โค้ดจะปริ้นเฉพาะชื่อที่ไม่ซ้ำกันเท่านั้น

20

21

ผลลัพธ์จะได้

- พี่บูม รัก ตายน์
- พี่บูม รัก แตงโม
- พี่บูม รัก ฝน
- พี่บูม รัก พี่เจ๊ท
- พี่บูม รัก น้องแคนบ์
- ตายน์ รัก พี่บูม
- ตายน์ รัก แตงโม
- ตายน์ รัก ฝน
- ตายน์ รัก พี่เจ๊ท
- ตายน์ รัก น้องแคนบ์
- แตงโม รัก พี่บูม
- แตงโม รัก ตายน์
- แตงโม รัก แตงโม
- แตงโม รัก ฝน
- แตงโม รัก พี่เจ๊ท
- แตงโม รัก น้องแคนบ์
- ฝน รัก พี่บูม
- ฝน รัก ตายน์
- ฝน รัก แตงโม
- ฝน รัก ฝน
- ฝน รัก พี่เจ๊ท
- ฝน รัก น้องแคนบ์
- พี่เจ๊ท รัก พี่บูม
- พี่เจ๊ท รัก ตายน์
- พี่เจ๊ท รัก แตงโม
- พี่เจ๊ท รัก ฝน
- พี่เจ๊ท รัก พี่เจ๊ท
- พี่เจ๊ท รัก น้องแคนบ์
- น้องแคนบ์ รัก พี่บูม
- น้องแคนบ์ รัก ตายน์
- น้องแคนบ์ รัก แตงโม
- น้องแคนบ์ รัก ฝน
- น้องแคนบ์ รัก พี่เจ๊ท
- น้องแคนบ์ รัก น้องแคนบ์

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 จำเป็นต้องมีส่วนประมวลผล ไม่สามารถปล่อยว่างไว้ได้

22

23

สัญลักษณ์ที่ใช้ใน conditional

- == คือ เท่ากับ
- != คือ ไม่เท่ากับ
- >= คือ มากกว่าหรือเท่ากับ ใช้ในกรณีตรวจสอบตัวเลข
- <= คือ น้อยกว่าหรือเท่ากับ ใช้ในกรณีตรวจสอบตัวเลข
- < คือ น้อยกว่า ใช้ในกรณีตรวจสอบตัวเลข
- > คือ มากกว่า ใช้ในกรณีตรวจสอบตัวเลข

23

24

Ture False ใน conditional

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

24

25

ตัวอย่าง Ture False ใน conditional

- 'พี่บูม' == 'พี่เจ๊ท'
- ผลลัพธ์จะได้
- False

- 'พี่บูม' != 'พี่เจ๊ท'
- ผลลัพธ์จะได้
- True

25

26

Homework class period 3

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

26

27