Mahidol University

Wisdom of the Land

Mahidol University

# Lecture 08: Review


Lecturers:

Aj. Jidapa Kraisangka

Aj. Akara Supratak

Aj. Tipajin Thaipisutikul

# `while` Statement



- ควบคุมโปรแกรมให้ทำงานบางอย่างซ้ำๆ ในขณะที่เงื่อนไข (expression) ของ loop นั้นยังคงเป็นจริง (ไม่เท่ากับ 0) อยู่
- มักถูกใช้เมื่อจำนวนรอบในการวน loop ไม่ทราบล่วงหน้า

```
while expression:
      statement(s)
```

**ขั้นตอนการทำงานของ `while` loop:**

1. Test the expression
2. If the expression is TRUE
   a. Execute the statement(s)
   b. Go to step 1

   Else exit the `while` loop

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# for Statement



- เป็นคำสั่งวนซ้ำที่ใช้ควบคุมการทำงานซ้ำๆ มักใช้สำหรับการวนอ่านค่าใน iterable object เช่น list, string, tuple
- มักใช้เมื่อทราบจำนวนรอบในการวน loop แน่นอน

```
for var in <sequence>:
    statement(s)
```

**ขั้นตอนการทำงาน for loop:**

1. Item แรกใน sequence ถูก assign เก็บใน **var**
2. ชุดคำสั่งใน for loop ถูก execute
3. If item ใน sequence ยังมีอยู่
   a. item ถัดมาใน sequence จะถูก assign ให้ **var**
   b. กลับไป step 2

   Else ออกจาก for loop

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# `range()` Function



- **`range()`** เป็น built-in function ที่มักใช้ร่วมกับ for loop
- ใช้ในการสร้าง sequence ของตัวเลข ภายใน range ที่กำหนด
- Default: เริ่มต้นที่ 0, เพิ่มขึ้นทีละ 1, และจบที่ตัวเลขสุดท้ายที่กำหนด

**Syntax:**

```
range(stop)
range(start, stop)
range(start, stop, step)
```

**มี parameter 3 ตัว เป็น type integer ได้แก่**

- **`start`:** ตัวเลขเริ่มต้น
- **`stop`:** ตัวเลขสุดท้าย (ไม่รวม)
- **`step`:** ค่าที่เปลี่ยนแปลง

**ไม่ support float numbers**

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# for vs. while



- **for:** ใช้เมื่อทราบจำนวนรอบในการ loop อย่างแน่นอน
- **while:** ใช้เมื่อจำนวนรอบในการ loop ไม่ทราบล่วงหน้า หรือทราบล่วงหน้าก็ได้ แล้วแต่ style การเขียน code

```
i = 1    initialization
while i<=10:    expression
    print(i, end=' ')
    i +=1 alteration
```

Generate a sequence of number from 0 to 9

```
for x in range(10):
    print(x, end=' ')
```

# การสร้าง List



การสร้าง list นั้นข้อมูลของ list จะอยู่ภายในเครื่องหมาย **[ ]** และสมาชิก (item) แต่ละตัวด้วยเครื่องหมาย **,**

การสร้าง list เปล่าที่ยังไม่มีข้อมูล

```
# การสร้าง List โดย [ ]
list1 = []
# หรือ ใช้คำสั่ง list()
list1 = list()
```

```
# List ของ integers
list1 = [1, 2, 3, 4, 5]

# List ของ string
list2 = ['a', 'b', 'c', 'd', 'a']

# List ของ integers, string และ list
list3 = [1, 2, 'b', 'c', list1]
```

Item ใน list สามารถซ้ำกันได้

Item ใน list สามารถมีได้หลายประเภท

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# การสร้าง 2D List



## การสร้าง 1D list ประกอบไปด้วย 0s ทั้งหมด n ตัว โดยใช้ *

```
n = 5
y = [0]* n
print(y)
```

**output**

```
[0, 0, 0, 0, 0]
```

## การสร้าง 2D list ขนาด rows x cols ประกอบไปด้วย 0s

```
rows = 3
cols = 5
x = [[0]*cols]*rows
print(x)
```

**output**

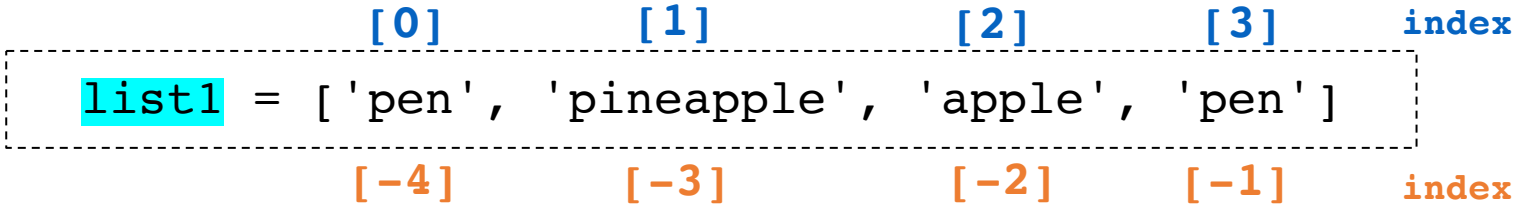
```
[[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# การเข้าถึงข้อมูลใน List



- List ใช้ index สำหรับการเข้าถึงข้อมูลแต่ละตัว
- หากมีข้อมูล n จำนวน เริ่มจากลำดับแรกด้วย index 0 จนถึงข้อมูลลำดับสุดท้ายคือ n-1
- Index สามารถติดลบได้ โดยเริ่มนับจากข้อมูลสุดท้ายของ list
  index -1 คือ สมาชิกตัวสุดท้ายของ list
  index -n คือ สมาชิกตัวแรกของ list



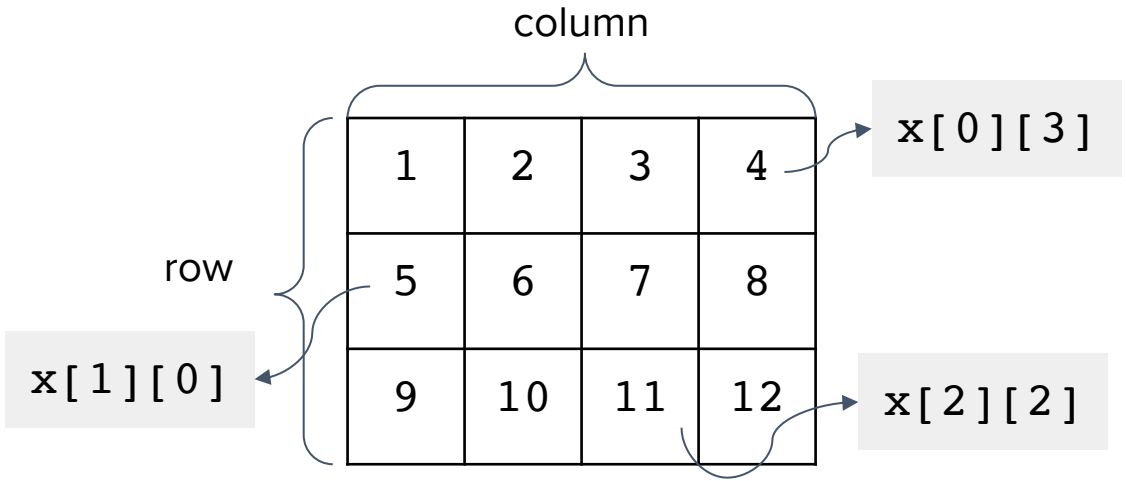

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# การเข้าถึงข้อมูลใน 2D List Index



- แต่ละข้อมูลใน 2D list สามารถเข้าถึงโดยการระบุตำแหน่ง **row** และ **col** เช่น **`x[row][column]`**

```
x = [[1, 2, 3, 4],[5, 6, 7, 8],[9, 10, 11, 12]]
```






Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# การเข้าถึงช่วงข้อมูลใน List



- การเข้าถึงช่วงของข้อมูลใน list สามารถทำได้ โดยการระบุ index เริ่มต้นและ index สุดท้าย (ไม่รวม) ของช่วงนั้น ๆ
- ถ้า **`start`** ไม่ถูกระบุ ข้อมูลจะเริ่มต้นที่ข้อมูลแรกใน list
- ถ้า **`end`** ไม่ถูกระบุ, range จะจบที่ข้อมูลสุดท้ายของ list

```
print(list2[0:2])    ผลลัพธ์ คือ    ['a', 'b']
print(list2[:2])     ผลลัพธ์ คือ    ['a', 'b']
print(list2[2:])     ผลลัพธ์ คือ    ['c', 'd', 1, 2, 3, 4]
```

# เปลี่ยนแปลงค่าใน List



ข้อมูลใน list สามารถถูกเปลี่ยนแปลงและแก้ไขได้ (mutable)

```
list1 = [1, 2, 3, 4]
list1[2] = 10
list1[-1] = 5
print(list1)
list1[0:2] = ['a','b']
print(list1)
```

```
[1, 2, 10, 5]
```

แก้ไขหลายข้อมูลพร้อมกัน

```
['a', 'b', 10, 5]
```

# คำสั่งพื้นฐานของ List



| Operation | Description | Example | Output |
|---|---|---|---|
| **Append** | ใส่ข้อมูลหนึ่ง item ที่ท้าย list | `x = [1, 2, 3]`<br>`x.append(10)`<br>`print(x)` | `[1, 2, 3, 10]` |
| **Insert** | ใส่ข้อมูลหนึ่ง item ที่ตำแหน่งที่กำหนด | `x = [1, 2, 3]`<br>`x.insert(1, 10)`<br>`print(x)` | `[1, 10, 1, 3]` |
| **Extend** | ใส่ข้อมูลหลาย item พร้อมๆกันที่ท้าย list | `x = [1, 2]`<br>`y = [3, 4]`<br>`y.extend(x)`<br>`print(y)` | `[3, 4, 1, 2]` |

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# คำสั่งพื้นฐานของ List



| Operation | Description | Example | Output |
|---|---|---|---|
| **Remove** | ลบข้อมูลที่ระบุจาก list | `x = [1, 2, 3]`<br>`x.remove(2)`<br>`print(x)` | `[1, 3]` |
| **Pop** | ลบและส่งข้อมูลตำแหน่งท้ายสุดของ list หรือข้อมูลในตำแหน่งที่ระบุ | `x = [1, 2, 3, 4]`<br>`x.pop() # pop 4`<br>`print(x)`<br>`x.pop(0) # pop 1`<br>`print(x)` | `[1, 2, 3]`<br>`[2, 3]` |
| **Clear** | ลบข้อมูลทั้งหมดจาก list | `x = [1, 2]`<br>`x.clear()`<br>`print(x)` | `[]` |



Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# List คำสั่งพื้นฐานของ List



| Operation | Description | Example | Output |
|---|---|---|---|
| **Len** | หาขนาดของ list หรือ จำนวนข้อมูลใน list | `x = [1, 2, 3]`<br>`n = len(x)`<br>`print(n)` | `3` |
| **Sort** | เรียงข้อมูลภาย list จากน้อยไปมาก หรือ จาก A-Z , a-z | `x = [4, 1, 3, 2]`<br>`x.sort()`<br>`print(x)` | `[1, 2, 3, 4]` |
| **Sort (reverse)** | เรียงข้อมูลภาย list จากน้อยไปมาก หรือ จาก A-Z , a-z | `x = [4, 1, 3, 2]`<br>`x.sort(reverse=True)`<br>`print(x)` | `[4, 3, 2, 1]` |

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# List คำสั่งพื้นฐานของ List



| Operation | Description | Example | Output |
|---|---|---|---|
| **Concatenation** | ใช้ในการเชื่อม list สอง list เข้าด้วยกัน | `x = [1, 2, 3]`<br>`y = [5, 4]`<br>`print(x+y)` | `[1, 2, 3, 4, 5]` |
| **Membership (in)** | ใช้ในการเช็คว่า item ที่ระบุอยู่ใน list หรือไม่ | `x = [1, 2, 3, 4]`<br>`print(2 in x)`<br>`print (10 in x)` | `True`<br>`False` |
| **Replication** | ใช้ในการสร้าง list ที่ประกอบไปด้วยข้อมูลซ้ำๆ | `x = 5*[0]`<br>`Y = 3*['a']`<br>`print(x)`<br>`print(y)` | `[0,0,0,0,0]`<br>`['a', 'a', 'a']` |

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX1



จงเขียนโปรแกรมเพื่อนับจำนวนสระใน `list_word` โดยไม่ใช้คำสั่ง count() และแสดงผลลัพธ์ในรูปแบบของ list ของจำนวนสระเป็น `'a', 'e', 'i', 'o', 'u'` ตามลำดับ
Hint: สามารถสร้าง `List` เพื่อเก็บจำนวน `['a', 'e', 'i', 'o', 'u']` ได้

ตัวอย่าง

```
Input:  ['p', 'r', 'o', 'g', 'r', 'a', 'm', 'm', 'i', 'n', 'g']
Output: [1, 0, 1, 1, 0]

Input:  ['r', 'e', 'a', 'l', 'i', 'z', 'e']
Output: [1, 2, 1, 0, 0]
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX1: Sample Solution

# EX1: Sample Solution



```
list_words= ['r', 'e', 'a', 'l', 'i', 'z', 'e'] # สามารถเปลี่ยนเป็นรับค่าจาก input ได้
vowels=['a', 'e', 'i', 'o', 'u']  # สร้าง vowels สำหรับตรวจสอบ
countvowels= [0]*5                # สร้าง list สำหรับเก็บผลลัพธ์การนับจำนวนสระ โดยทุกตัวมีค่าเป็น 0
```

# EX1: Sample Solution



```
list_words= ['r', 'e', 'a', 'l', 'i', 'z', 'e'] # สามารถเปลี่ยนเป็นรับค่าจาก input ได้
vowels=['a', 'e', 'i', 'o', 'u']  # สร้าง vowels สำหรับตรวจสอบ
countvowels= [0]*5                # สร้าง list สำหรับเก็บผลลัพธ์การนับจำนวนสระ โดยทุกตัวมีค่าเป็น 0

# ตรวจว่า แต่ละสระอยู่ใน list_word หรือไม่ ถ้ามี ให้นับทีละสระ
for i in range(len(vowels)):        # คำสั่ง len เป็นการหาขนาดของ list
        if (vowels[i] in list_words):       # คำสั่ง in เป็นการตรวจว่าค่านั้นอยู่ใน list หรือไม่
                #  หากเงื่อนไขเป็นจริง คือ ตรวจเจอสระ ให้นับไปเรื่อย ๆ จนกว่าหมด list_words
                for j in range(len(list_words)):
                        if vowels[i]==list_words[j]:
                                countvowels[i]+=1
```

# EX1: Sample Solution



```
list_words= ['r', 'e', 'a', 'l', 'i', 'z', 'e'] # สามารถเปลี่ยนเป็นรับค่าจาก input ได้
vowels=['a', 'e', 'i', 'o', 'u']  # สร้าง vowels สำหรับตรวจสอบ
countvowels= [0]*5                # สร้าง list สำหรับเก็บผลลัพธ์การนับจำนวนสระ โดยทุกตัวมีค่าเป็น 0

# ตรวจว่า แต่ละสระอยู่ใน list_word หรือไม่ ถ้ามี ให้นับทีละสระ
for i in range(len(vowels)):        # คำสั่ง len เป็นการหาขนาดของ list
        if (vowels[i] in list_words):       # คำสั่ง in เป็นการตรวจว่าค่านั้นอยู่ใน list หรือไม่
                #  หากเงื่อนไขเป็นจริง คือ ตรวจเจอสระ ให้นับไปเรื่อย ๆ จนกว่าหมด list_words
                for j in range(len(list_words)):
                        if vowels[i]==list_words[j]:
                                countvowels[i]+=1
# แสดงผลลัพธ์เป็นจำนวนสระ ตามลำดับ
print(countvowels)
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX2: Pattern



จงเขียนโปรแกรมเพื่อแสดงผลลัพธ์เป็นรูปจัตุรัสขนาด nxn โดย n>1 ตามรูปแบบต่อไปนี้

- เส้นทแยงมุมเป็น เลขหลักหน่วยของ n เช่น n=12 เลขหลักหน่วยคือ 2
- ครึ่งล่างของจัตุรัสเป็นเครื่องหมาย –
- ครึ่งบนของจัตุรัสเป็นเครื่องหมาย +

n=3

```
3 + +
– 3 +
– – 3
```

n=5

```
5 + + + +
– 5 + + +
– – 5 + +
– – – 5 +
– – – – 5
```

n=12

```
2 + + + + + + + + + + +
– 2 + + + + + + + + + +
– – 2 + + + + + + + + +
– – – 2 + + + + + + + +
– – – – 2 + + + + + + +
– – – – – 2 + + + + + +
– – – – – – 2 + + + + +
– – – – – – – 2 + + + +
– – – – – – – – 2 + + +
– – – – – – – – – 2 + +
– – – – – – – – – – 2 +
– – – – – – – – – – – 2
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX2: Pattern (วิเคราะห์โจทย์)



**วิเคราะห์ Pattern : n=5**

จากรูปแบบที่กำหนด เครื่องหมาย + - และเลขหลักหน่วย อยู่ตรงไหนบ้าง

- แต่ละแถว มี 5 สัญลักษณ์ **n=5**
- เส้นทแยงมุมเป็น เลขหลักหน่วยของ n เช่น n=5 เลขหลักหน่วยคือ 5

  การหาเลขหลักหน่วยสามารถใช้ Modulo (%) ได้
  เส้นทแยงมุม คือ เมื่อตำแหน่ง i เท่ากับ j

- ครึ่งล่างของจัตุรัสเป็นเครื่องหมาย –

  จำนวน - ในแต่ละแถว คือ 1 2 3 4 ตามลำดับ
  จากตำแหน่ง i ถึง j (ไม่รวม)

- ครึ่งบนของจัตุรัสเป็นเครื่องหมาย +

  จำนวน + ในแต่ละแถว คือ 4 3 2 1 ตามลำดับ
  จากตำแหน่ง i+1 ถึง n-1

| i\j | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 5 | + | + | + | + |
| 1 | – | 5 | + | + | + |
| 2 | – | – | 5 | + | + |
| 3 | – | – | – | 5 | + |
| 4 | – | – | – | – | 5 |

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX2: Sample Solution

# EX2: Sample Solution



```
n=int(input())    # รับค่า n
d= n%10           # หาหลักหน่วยของ n

# สร้าง loop เพื่อทำการ print แต่ละสัญลักษณ์
for i in range(n):          # คุมจำนวนแถว หรือ row
```

# EX2: Sample Solution



```
n=int(input())    # รับค่า n
d= n%10           # หาหลักหน่วยของ n

# สร้าง loop เพื่อทำการ print แต่ละสัญลักษณ์
for i in range(n):          # คุมจำนวนแถว หรือ row
        for j in range(i+1):        # สำหรับคุมสามเหลี่ยม (- และหลักหน่วย d)
                if(i==j):           # ตรวจสอบว่าอยู่ในแนวเส้นทแยงมุมหรือไม่
                        print(d, end=' ')
                else:
                        print('-', end=' ')
```

# EX2: Sample Solution



```
n=int(input())    # รับค่า n
d= n%10           # หาหลักหน่วยของ n

# สร้าง loop เพื่อทำการ print แต่ละสัญลักษณ์
for i in range(n):          # คุมจำนวนแถว หรือ row
        for j in range(i+1):        # สำหรับคุมสามเหลี่ยม (- และหลักหน่วย d)
                if(i==j):           # ตรวจสอบว่าอยู่ในแนวเส้นทแยงมุมหรือไม่
                        print(d, end=' ')
                else:
                        print('-', end=' ')
        for k in range(j, n-1):             # สำหรับคุมสามเหลี่ยม +
                print('+', end=' ')
        print()                             # สำหรับการขึ้นบรรทัดใหม่
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX3: เปรียบเทียบค่า x กับค่าในเมทริกซ์



จงเขียนโปรแกรมเพื่อเปรียบเทียบค่าในเมทริกซ์ mxn กับตัวเลขที่ input มา และแสดงผลลัพธ์

โปรแกรมจะรับจำนวนเต็มบวก m และ n ตามลำดับ โดย m คือจำนวนแถว และ n คือจำนวนหลักของเมทริกซ์ จากนั้น โปรแกรมจะวนรับค่าตัวเลขใด ๆ ไปเรื่อย ๆ เพื่อใส่ข้อมูลใน เมทริกซ์ขนาด mxn

เมื่อรับข้อมูลเรียบร้อยแล้ว โปรแกรมจะถามถึงตัวเลข (x) ที่ต้องการเปรียบเทียบ จากนั้นโปรแกรมจะเปรียบเทียบค่า x กับข้อมูลในเมทริกซ์ทีละตัว และแสดงผลลัพธ์เป็นเมทริกซ์ที่มี เครื่องหมาย > = หรือ < ตามผลการเปรียบเทียบ เช่น หาก x มีค่ามากกว่าข้อมูลในตำแหน่งใด ๆ ในเมทริกซ์ เมทริกซ์ผลลัพธ์ในตำแหน่งนั้น ๆ จะเป็น '>'

**ตัวอย่าง** Input m=2, n=3, x=5

$$mat = \begin{bmatrix} 2 & 5 & 7 \\ 8 & 9 & 1 \end{bmatrix}$$

ผลลัพธ์

$$result = \begin{bmatrix} > & = & < \\ < & < & > \end{bmatrix}$$

```
m:2
n:3
2
5
7
8
9
1
x:5
[[2, 5, 7], [8, 9, 1]]
[['>', '=', '<'], ['<', '<', '>']]
```

Contact: [jidapa.kra, akara.sup, tipajin.tha]@mahidol.edu

# EX3: Sample Solution

# EX3: Sample Solution



```
m = int(input('m:'))        # รับค่า m สำหรับจำนวนแถว หรือ row
n = int(input('n:'))        # รับค่า n สำหรับจำนวนหลัก หรือ column


# สร้าง 2D matrix และวนเพื่อรับค่าตัวเลขทีละค่า
mat=[]
for i in range(m):
        row=[]
        for j in range(n):
                num = int(input())
                row.append(num)
        mat.append(row)
```

# EX3: Sample Solution



ต่อจากหน้าที่แล้ว

```
# เมื่อรับข้อมูลเรียบร้อยแล้ว โปรแกรมจะถามถึงตัวเลข (x) ที่ต้องการเปรียบเทียบ
x = int(input('x:'))

result=[] # สร้างเมทริกซ์สำหรับการเก็บผลลัพธ์การเปรียบเทียบ
for i in range(m):
        row=[]
        for j in range(n):
                if(x>mat[i][j]):                # เปรียบเทียบ x และข้อมูลในเมทริกซ์
                        row.append('>')
                elif (x<mat[i][j]):
                        row.append('<')
                else:
                        row.append('=')
        result.append(row)
print(result)
```


Thanks to: https://keep-calm.net/keep-calm-and-code-on.html

# Good luck