01076103, 01076104

Programming Fundamental

Programming Project

List, String, Loop

List

- List เป็นโครงสร้างข้อมูลของ Python ที่สามารถเก็บข้อมูลได้หลายตัว และสามารถเก็บข้อมูลต่างชนิดกันได้ด้วย
- List จะใช้เครื่องหมาย [ ] ในการกำหนดขอบเขตของ List
- การใช้งาน List อาจเก็บข้อมูลที่เป็นประเภทเดียวกัน เช่น

```
# a list of programming languages
['Python', 'C++', 'JavaScript']
```

- หรือต่างชนิดกัน หรือ ไม่มีข้อมูลอยู่เลยก็ได้

```
# empty list
my_list = []

# list with mixed data types
my_list = [1, "Hello", 3.4]
```

List

- นอกจากนั้นข้างใน List อาจจะมี List อื่นๆ อีกก็ได้ เรียกว่า Nested List

```
# nested list
my_list = ["mouse", [8, 4, 6], ['a']]
```

- List สามารถชี้โดยตัวแปรตัวเดียวกัน แต่จะหมายถึงข้อมูลทั้งชุด สำหรับการอ้างถึงข้อมูลแต่ละตัว สามารถทำได้โดยการอ้าง Index โดยข้อมูลตัวแรก จะอยู่ที่ Index 0
- เช่น จากโปรแกรมข้างต้น
  - my_list[0] จะหมายถึง “mouse”
  - my_list[1] จะหมายถึง [8, 4, 6]
  - my_list[2] จะหมายถึง [‘a’]
  - my_list [1][0] จะหมายถึง 8

List

```
my_list = ['p', 'r', 'o', 'b', 'e']

# first item
print(my_list[0])  # p

# third item
print(my_list[2])  # o

# fifth item
print(my_list[4])  # e

# Nested List
n_list = ["Happy", [2, 0, 1, 5]]

# Nested indexing
print(n_list[0][1])

print(n_list[1][3])

# Error! Only integer can be used for indexing
print(my_list[4.0])
```

List

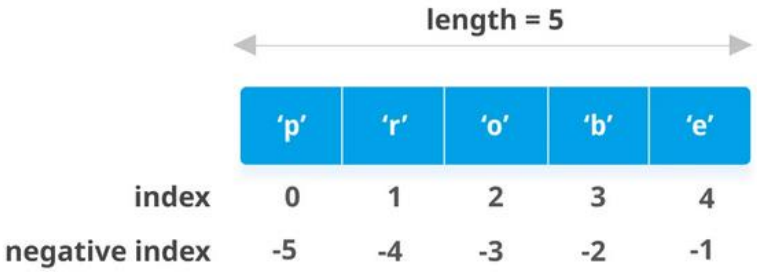
- Index สามารถใช้เป็นลบได้ด้วย โดย -1 จะหมายถึง สมาชิกตัวสุดท้ายของ List และ -2 หมายถึงสมาชิกตัวรองสุดท้ายของ List พูดง่ายๆ คือ การนับจากหลังมาข้างหน้า

```
# Negative indexing in lists
my_list = ['p','r','o','b','e']

# last item
print(my_list[-1])

# fifth last item
print(my_list[-5])
```

- จะได้ผลลัพธ์ บรรทัดแรก คือ e
- และ บรรทัดที่ 2 คือ p



List

- นอกจากการระบุสมาชิกของ List ตัวใดตัวหนึ่งแล้ว สามารถระบุสมาชิกเป็นช่วงได้ด้วย เรียกว่า Slicing

main.py ×

```
1  # List slicing in Python
2
3  my_list = ['c','o','m','e','n','g']
4
5  # elements from index 2 to index 4
6  print(my_list[2:5])
7
8  # elements from index 5 to end
9  print(my_list[5:])
10
11 # elements beginning to end
12 print(my_list[:])
```

Console Shell

```
['m', 'e', 'n']
['g']
['c', 'o', 'm', 'e', 'n', 'g']
>
```

List

- การทำ Slicing ยังสามารถเพิ่ม Step คือ การเลือกสมาชิกแบบกระโดดได้ด้วย

main.py

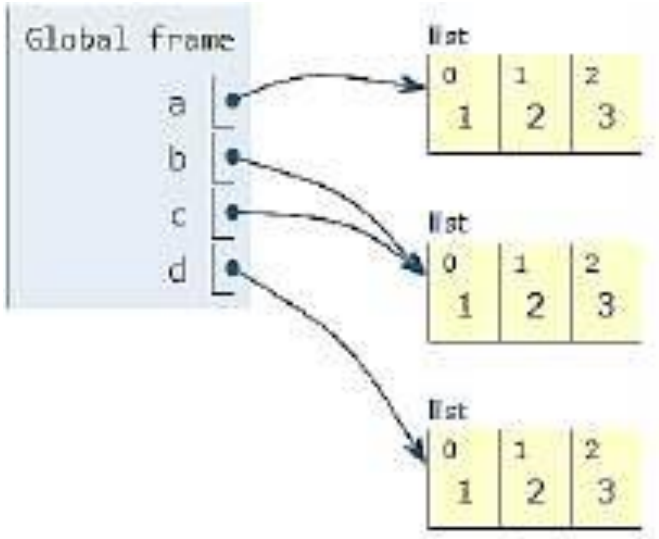
```
1  list1 = ['physics', 'chemistry',
   'calculus', 'biology'];
2  list2 = [1, 2, 3, 4, 5, 6, 7];
3  print("list1[0]:", list1[0])
4  print("list1[-1]:", list1[-1])
5  print("list2[3]:", list2[3])
6  print("list2[-4]:", list2[-4])
7  print("list2[1:5]:", list2[1:5])
8  print("list2[::2]:", list2[::2])
9  print("list2[2::2]:",
   list2[2::2])
10 print("list2[2:7:2]:",
   list2[2:7:2])
11 print("list2[:7]:", list2[:7])
12 print("list2[4:]:", list2[4:])
```

Console Shell

```
list1[0]: physics
list1[-1]: biology
list2[3]: 4
list2[-4]: 4
list2[1:5]: [2, 3, 4, 5]
list2[::2]: [1, 3, 5, 7]
list2[2::2]: [3, 5, 7]
list2[2:7:2]: [3, 5, 7]
list2[:7]: [1, 2, 3, 4, 5, 6, 7]
list2[4:]: [5, 6, 7]
>
```

List

- คำสั่ง list จะใช้สร้าง list
- x = list('abcde’) จะได้ x = ['a', 'b', 'c', 'd', 'e’]
- a = [1,2,3]; b = [1,2,3]; c = b; d = list(b) หลังจากทำคำสั่งจะได้



List

- การเพิ่มข้อมูลเข้าไปใน List สามารถใช้ได้หลายวิธี
- โดยการ +

main.py

```
1  flowers = ['Rose', 'Lily', 'Tulip']
2  flowers += ['Jasmine']
3
```

Console Shell

```
['Rose', 'Lily', 'Tulip', 'Jasmine']
```

- แต่หากเขียนโปรแกรมแบบนี้ ผลจะเป็นอีกแบบหนึ่ง

main.py

```
1  flowers = ['Rose', 'Lily', 'Tulip']
2  flowers += 'Jasmine'
3
```

```
['Rose', 'Lily', 'Tulip', 'J', 'a', 's', 'm', 'i', 'n', 'e']
```

List

- การเพิ่มโดยใช้ append

main.py

```
1  # Appending and Extending lists in Python
2  odd = [1, 3, 5]
3
4  odd.append(7)
5
6  print(odd)
```

Console Shell

```
[1, 3, 5, 7]
```

- การเพิ่มโดยใช้ extend (ข้อมูลที่ extend ต้องเป็น list เช่นกัน)

main.py

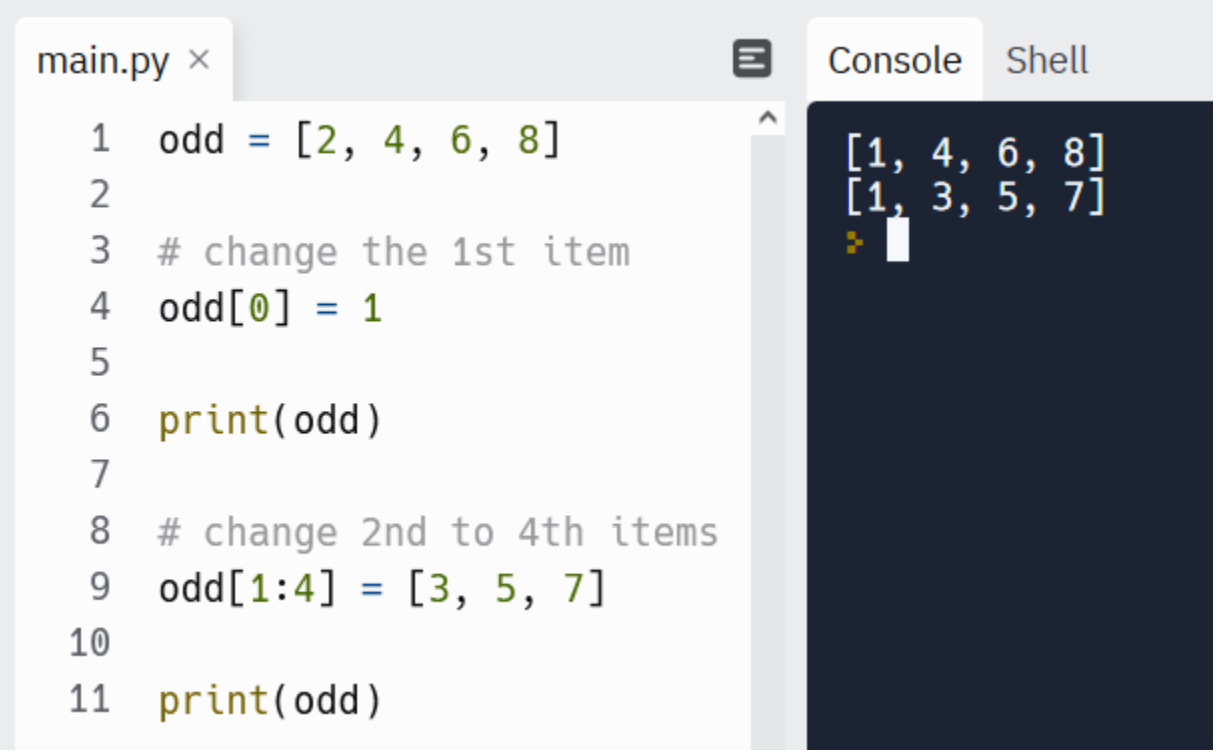
```
1  odd = [1, 3, 5]
2  odd.extend([9, 11, 13])
3  print(odd)
```

Console Shell

```
[1, 3, 5, 9, 11, 13]
```

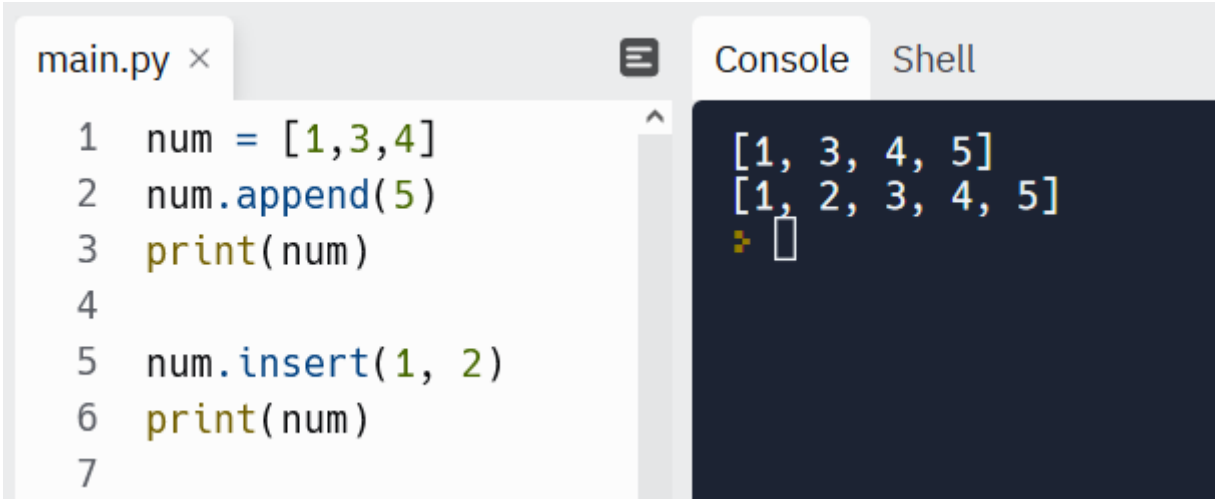
List

- การแก้ไขข้อมูลใน List จะเป็นการกำหนดค่าใหม่ให้กับแต่ละตำแหน่งที่อยู่ใน List โดยตรง



List

- สามารถแทรกข้อมูลเข้าไปใน List ได้ด้วย (insert(pos, value))



- สามารถใช้ Operator * กับ List ได้

List

- สำหรับการลบข้อมูลจาก List สามารถทำได้หลายวิธีเช่นกัน

main.py

```
1  flowers = ['Rose', 'Lily', 'Tulip', 'Sunflower']
2  print(flowers)
3
4  del flowers[2]
5  print(flowers)
6
7  flowers.pop(0)
8  print(flowers)
9
10 flowers.remove('Lily')
11 print(flowers)
12
13 flowers.clear()
14 print(flowers)
15
```

Console Shell

```
['Rose', 'Lily', 'Tulip', 'Sunflower']
['Rose', 'Lily', 'Sunflower']
['Lily', 'Sunflower']
['Sunflower']
[]
```

- คำสั่ง del จะต้องระบุตำแหน่ง (ใช้ slicing ได้) , คำสั่ง pop จะคล้ายกับ del แต่สามารถ return ตัวที่ลบออกมาได้ (ถ้าไม่ระบุจะได้ตัวสุดท้าย), คำสั่ง remove จะต้องระบุข้อมูล

List

- ฟังก์ชันที่เกี่ยวกับ List อื่นๆ
  - x.sort() ทำให้ข้อมูลในลิสต์ x เรียงจากน้อยไปมาก คำสั่งนี้ไม่มี return คืนกลับมา
  - sorted(x) คืนลิสต์ที่มีค่าเหมือนกับใน x แต่เรียงลำดับข้อมูลจากน้อยไปมากให้ (x ไม่เปลี่ยนแปลง)
  - sum(x) คืนผลรวม (summary) ของตัวเลขที่อยู่ในลิสต์ x
  - max(x) คืนค่ามากสุดในลิสต์ x, min(x) คืนค่าน้อยสุดในลิสต์ x
  - x.count(e) คืนจำนวนครั้งที่ e ปรากฏในลิสต์ x

List

- เราสามารถตรวจสอบได้ว่ามีสมาชิกนั้นอยู่ใน List นั้นหรือไม่ (Membership)

main.py

```
1  my_list = ['p', 'r', 'o', 'b', 'l', 'e', 'm']
2
3  # Output: True
4  print('p' in my_list)
5
6  # Output: False
7  print('a' in my_list)
8
9  # Output: True
10 print('c' not in my_list)
11
```

Console Shell

```
True
False
True
>
```

List

- เราสามารถเปรียบเทียบ List ได้ โดยวิธีการเปรียบเทียบจะเริ่มจากเปรียบเทียบสมาชิกตัวที่ 1 ของ List ทั้งสอง ถ้าสมาชิกตัวแรกมากกว่าหรือน้อยกว่า ก็จะใช้ผลลัพธ์นั้นเป็นคำตอบ แต่หากมีค่าเท่ากัน ก็จะเลื่อนไปเปรียบเทียบสมาชิกอันดับถัดไป

main.py

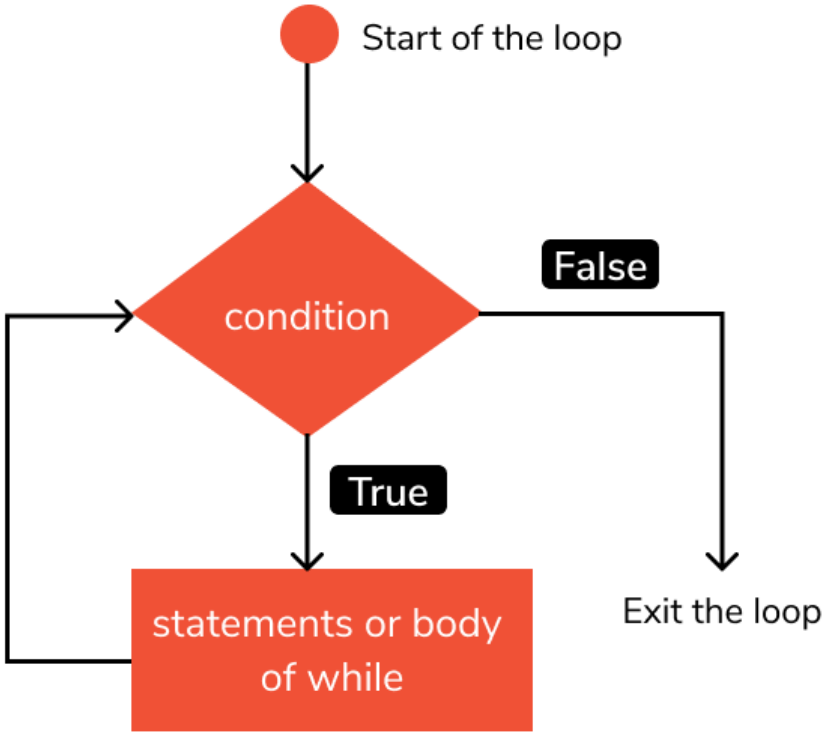
```
1  lst1 = [1, 2, 3, 4, 5]
2  lst2 = [9, 8, 7, 6, 5]
3  lst3 = [9, 8, 7, 6, 5]
4  lst4 = [8, 7]
5  print("lst1 < lst2 :", lst1 < lst2)
6  print("lst1 > lst2 :", lst1 > lst2)
7  print("lst2 >= lst1 :", lst2 >= lst1)
8  print("lst2 == lst3 :", lst2 == lst3)
```

Console Shell

```
lst1 < lst2 : True
lst1 > lst2 : False
lst2 >= lst1 : True
lst2 == lst3 : True
```

While Loop

- Loop เป็นการทำงานแบบวนซ้ำ คือมีการทำงานไปเรื่อยๆ จนกว่าจะตรงกับเงื่อนไขที่กำหนด จึงออกจาก Loop
- สำหรับ While Loop เป็น Loop แบบหนึ่งที่มีการใช้งานมาก ลักษณะ คือ จะตรวจสอบเงื่อนไข หากเป็น False ก็จะไม่ทำใน Code Block แต่หากเป็นจริงก็จะทำใน Code Block ต่อไป ตามรูป
- รูปแบบ Syntax มีดังนี้




```
while test_expression:
    Body of while
```

While Loop

- ตัวอย่างโปรแกรม บวกตัวเลขตั้งแต่ 1-n : sum = 1+2+3+...+n

main.py

```
1  n = int(input("Enter n: "))
2
3  # initialize sum and counter
4  sum = 0
5  i = 1
6
7  while i <= n:
8      sum = sum + i
9      i = i+1    # update counter
10
11 print("The sum is", sum)
12
```

Console Shell

```
Enter n: 10
The sum is 55
```

While Loop

- While Loop อาจใช้ในหลายๆ งาน เช่น ถามให้ทำงานซ้ำ

main.py

```
1  again = 'y'
2  while again == 'y':
3      height = int(input("Enter Height :"))
4      base = int(input("Enter Base :"))
5      print("Triangle Area :", height * base * 0.5)
6
7      again = input("Again (y/n) :")
8
```

Console Shell

```
Enter Height :10
Enter Base :5
Triangle Area : 25.0
Again (y/n) :y
Enter Height :5
Enter Base :5
Triangle Area : 12.5
Again (y/n) :
```

- ใช้ในการรับข้อมูลจนกว่าจะได้ ตามที่กำหนด

main.py

```
1  n = int(input("Enter a number > 0: "))
2
3  while n <= 0:
4      n = int(input("Enter a number > 0: "))
5
```

Console Shell

```
Enter a number > 0: -5
Enter a number > 0: 0
Enter a number > 0: 5
```

While Loop

- **Exercise** ให้เข้าไปที่ Web https://reeborg.ca/reeborg.html?lang=en&mode=python&menu=worlds%2Fmenus%2Freeborg_intro_en.json&name=Hurdle%201&url=worlds%2Ftutorial_en%2Fhurdle1.json
- แล้วเขียนโปรแกรมเพื่อพา Robot ไปที่ธง โดยใช้แค่ move() และ turn_left()
  - move() เคลื่อนที่ไปข้างหน้า
  - turn_left() เลี้ยวซ้าย
- จากนั้นให้เลือก Hurdle2 ซึ่งธงจะ random แสดง มีฟังก์ชัน at_goal() เพิ่มให้ใช้ (hint : ใช้ while loop)
- จากนั้นให้เลือก Hurdle3 กำแพงจะหายไปบางอัน แต่จะมีฟังก์ชัน front_is_clear() และ wall_in_front() (hint : ใช้ if)
- จากนั้นให้เลือก Hurdle4 กำแพงจะสูงไม่เท่ากัน จะมีฟังก์ชัน wall_on_right() และ right_is_clear() เพิ่มให้ใช้งาน
- จากนั้นให้เลือก Maze

While Loop

- ในการวน Loop เพื่อทำสิ่งใดสิ่งหนึ่ง บางครั้งเมื่อตรงกับเงื่อนไขที่กำหนดและต้องการออกจาก Loop เราจะใช้คำสั่ง break ซึ่งจะทำให้การทำงานหลุดจาก Loop

main.py

```
1  # break the loop as soon it sees 'e' or 's'
2  i = 0
3  str1 = 'computer engineering'
4
5  while i < len(str1):
6      if str1[i] == 'e' or str1[i] == 's':
7          i += 1
8          break
9
10     print('Current Letter :', str1[i])
11     i += 1
12
```

Console Shell

```
Current Letter : c
Current Letter : o
Current Letter : m
Current Letter : p
Current Letter : u
Current Letter : t
```

While Loop

- ในบางครั้งเราต้องการข้อยกเว้น ไม่ทำงานใน Block Code สำหรับบางเงื่อนไข เราสามารถใช้ continue เพื่อให้ข้ามกลับไปที่ต้น Loop ใหม่

main.py

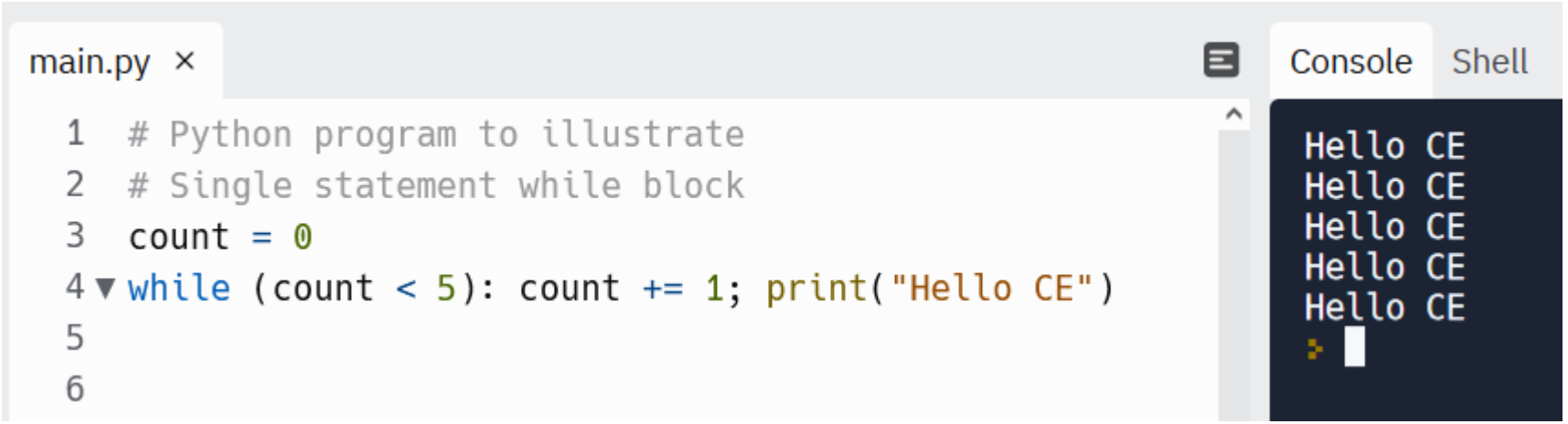
```
1  # Prints all letters except 'e' and 's'
2  i = 0
3  str1 = 'computer engineering'
4
5  while i < len(str1):
6      if str1[i] == 'e' or str1[i] == 's':
7          i += 1
8          continue
9
10     print('Current Letter :', str1[i])
11     i += 1
12
13
14
```

Console Shell

```
Current Letter : c
Current Letter : o
Current Letter : m
Current Letter : p
Current Letter : u
Current Letter : t
Current Letter : r
Current Letter :
Current Letter : n
Current Letter : g
Current Letter : i
Current Letter : n
Current Letter : r
Current Letter : i
Current Letter : n
Current Letter : g
```

While Loop

- กรณีที่เราต้องการใช้ While Loop โดยการทำงานสั้นๆ เพียงคำสั่งเดียว สามารถเขียนลดรูปแบบบรรทัดเดียวได้



While Loop

- While Loop ของ Python จะมีความพิเศษที่ไม่มีในภาษาอื่น คือ สามารถใช้ Else ได้
- โดย Else Block จะทำงานหากไม่มีการ break เกิดขึ้นใน While Loop

main.py ×

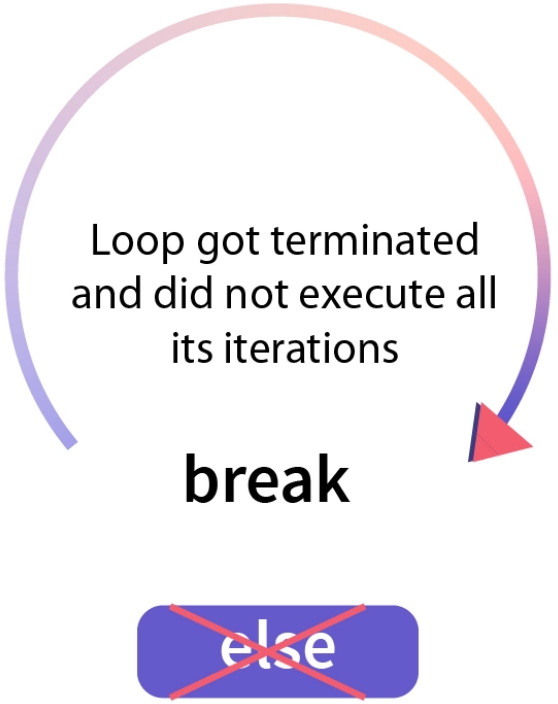
```
1  # Python program to demonstrate
2  # while-else loop
3
4  i = 0
5  while i < 4:
6      i += 1
7      print(i)
8  else:  # Executed because no break in for
9      print("No Break\n")
```

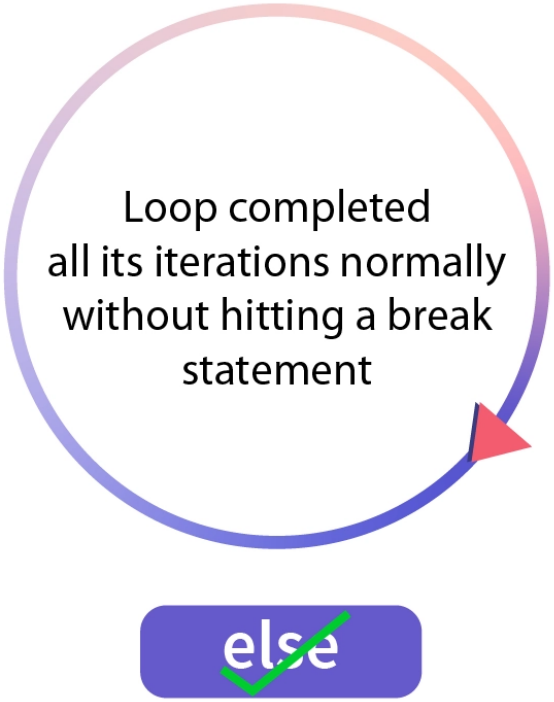
Console Shell

```
1
2
3
4
No Break
```

While Loop

- Else Block จะทำงานหากไม่มีการ break เกิดขึ้นใน While Loop





While Loop

- ตัวอย่าง While Loop ที่มี Else

main.py

```
1  # list of fruits
2  my_list =
   ["papaya","banana","pineapple","mango","grapes"]
3
4  size = len(my_list) #length/size of the list
5  i=0
6
7  # iterating through the fruit list
8  while i<size:
9      if my_list[i] == 'mango':
10         print("mango found!")
11         break
12     i+=1
13 else:
14     print("mango not found!")
```

Console Shell

```
mango found!
```

While Loop

- **ตัวอย่าง** โยนลูกบอลจากตึกสูง h เมตร หลังจากลูกบอลกระทบพื้นจะเด้งขึ้นมา จะเด้งขึ้นมาเท่ากับ 90 เปอร์เซ็นต์ของความสูงเดิม จากนั้นจะตกลงไปและเด้งขึ้นมาอีกเท่ากับ 90 เปอร์เซ็นต์ของความสูงเดิม เป็นแบบนี้ไปเรื่อยๆ จนกว่าความสูงน้อยกว่า 0.1 เมตร จึงจะหยุดเด้ง ให้หาว่าบอลจะเด้งทั้งหมดกี่ครั้ง
- วิเคราะห์โจทย์ รายละเอียดการทำงานจะเป็นดังนี้
  - บอลอยู่ที่ความสูง h
  - ลูกบอลตกลงกระทบพื้น และ เด้งกลับไปที่ 0.9h
  - ลูกบอลตกลงกระทบพื้น และ เด้งกลับไปที่ 0.9h ของ 0.9h
  - เป็นแบบนี้ไปเรื่อยๆ
  - เมื่อความสูงน้อยกว่า 0.1 เมตร ให้หยุด

While Loop

- สามารถเขียนเป็นโค้ดเทียมได้ดังนี้
  - รับค่า Input ความสูง h
  - กำหนดค่าตัวนับรอบ = 0
  - ถ้า h > 0.1 ให้ เปลี่ยนค่าความสูง h = 0.9 * h และเพิ่มค่าตัวนับ
  - แสดงผลค่าตัวนับ

While Loop

- **Exercise 3.1** จากข้อมูลใน List ให้หาผลบวกเฉพาะตัวเลขคู่ โดยใช้ While Loop

- **Exercise 3.2** จากข้อมูลใน List ให้หาตัวเลขที่มีค่ามากที่สุดเป็นอันดับ 2 ใน List โดยใช้ While Loop

- **Exercise 3.3** เขียนโปรแกรมรับค่า n แล้วคำนวณค่าต่อไปนี้

  — $1/1^2 + 1/2^2 + 1/3^2 + 1/4^2 + 1/5^2 + \ldots + 1/n^2$

String

- เราอาจมอง String คล้ายกับ List ได้ เช่น สามารถทำ Slicing ได้

main.py

```
1  #Accessing string characters in Python
2  str = 'computer'
3  print('str = ', str)
4
5  #first character
6  print('str[0] = ', str[0])
7
8  #last character
9  print('str[-1] = ', str[-1])
10
11 #slicing 2nd to 5th character
12 print('str[1:5] = ', str[1:5])
13
14 #slicing 6th to 2nd last character
15 print('str[5:-2] = ', str[5:-2])
```

Console Shell

```
str =  computer
str[0] =  c
str[-1] =  r
str[1:5] =  ompu
str[5:-2] =  t
```

String

- แต่ไม่สามารถเปลี่ยนแปลง หรือ ลบตัวอักษรออกจาก String เช่นเดียวกับที่ทำกับ List ได้
- String มี Function ให้ใช้มากมาย วิธีการใช้ Function ของ String จะใช้ .function หลังตัวแปร

main.py

```
1  # Convert to upper Accessing string characters in Python
2  name = "Ada Lovelace"
3  print(name.upper())
4  print(name.lower())
5
6  # all chars have been stripped from the end of the string
7  favorite_language = 'python '
8  print(favorite_language+':')
9  print(favorite_language.rstrip()+':')
10
11 # all chars have been stripped from the begin of the string
12 favorite_language = ' python '
13 print(':'+favorite_language)
14 print(':'+favorite_language.lstrip())
```

Console Shell

```
ADA LOVELACE
ada lovelace
python :
python:
: python
:python
```

String

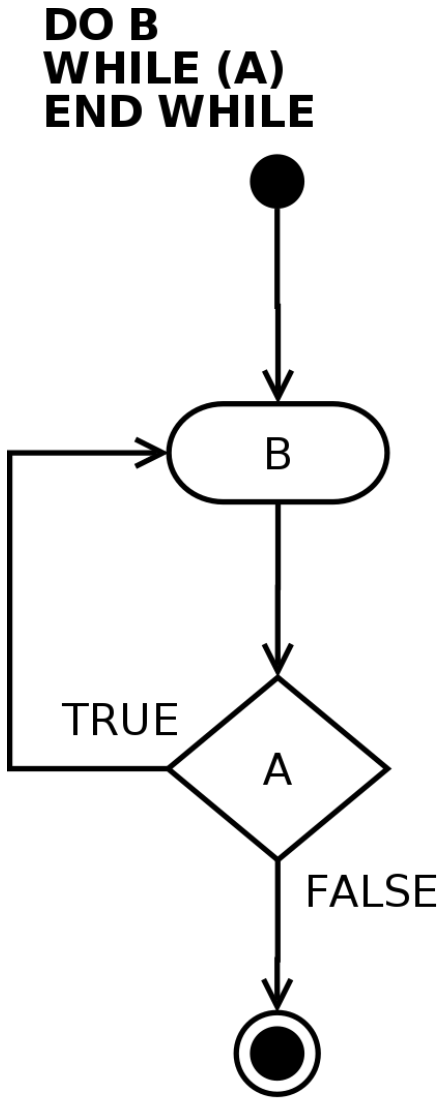
- ฟังก์ชันอื่นๆ
  - Isupper ตัวใหญ่หรือไม่
  - Islower ตัวเล็กหรือไม่
  - Isnumeric ตัวเลขหรือไม่
  - Isalpha ตัวอักษรหรือไม่
  - Isalnum Alphanumeric หรือไม่
  - Count นับจำนวนของอักษร
  - Find หาตำแหน่งตัวอักษรหรือชุด (หาตัวที่ n ได้ด้วย)
  - Startswith เริ่มต้นด้วย
  - Endswith ปิดท้ายด้วย
  - Replace เปลี่ยนข้อความ
  - Split แยก String และส่งกลับใน List
  - Join รวม List เป็น String (ข้อมูลใน List ต้องเป็น String)

String

- มีค่าคงที่ซึ่งเกี่ยวกับ String ที่ควรรู้ (ต้อง import string)
  - string.ascii_letters
    - ‘ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz’
  - string.ascii_lowercase
    - ‘abcdefghijklmnopqrstuvwxyz’
  - string.ascii_uppercase
    - ‘ABCDEFGHIJKLMNOPQRSTUVWXYZ’
  - string.digits
    - ‘0123456789’
  - Punctuation
    - !”#$%&'()*+,-./:;<=>?@[\]^_`{|}~

Do While Loop

- ในภาษา Programming อื่น จะมีการทำงานของ Loop ที่เรียกว่า Do While
- การทำงานจะเป็นไปตามรูป คือ จะมีการทำงานที่ Block B จำนวน 1 ครั้งก่อน จากนั้นจึงจะทดสอบเงื่อนไข A หากเงื่อนไขเป็น False ก็จะหยุดทำงาน แต่ถ้าเงื่อนไขเป็นจริง ก็จะกลับไปทำที่ต้น Loop อีก
- ใน Python ไม่มี Loop แบบ Do While



Do While Loop

- กรณีที่จำเป็นต้องใช้ Do While Loop ให้ใช้เทคนิคตามตัวอย่างนี้
- การใช้ While True จะทำให้เกิด Loop และใช้ break เพื่อออกจาก Loop

main.py

```
1  secret_word = "python"
2  counter = 0
3
4  while True:
5      word = input("Enter the secret word: ").lower()
6      counter = counter + 1
7      if word == secret_word:
8          break
9      if word != secret_word and counter > 7:
10         break
11
```

For Loop

- เป็น Loop อีกประเภทหนึ่งที่มีการใช้งานมาก โดยมักใช้กับ Loop ที่มีจำนวนการวนที่จำกัดที่ค่าใดค่าหนึ่ง (ในขณะที่ While จะไม่จำกัดจำนวนครั้ง แต่อยู่ที่เงื่อนไข)
- Loop For มักใช้กับข้อมูลที่เป็นชุด เช่น List

Range

- ฟังก์ชันที่มักใช้คู่กับ For Loop คือ range ซึ่งเป็นฟังก์ชันที่คืนค่าเป็นลำดับเลข โดยมีรูปแบบการใช้งานดังนี้

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

main.py

```
1  # numbers from 0 to 3 (4 is not included)
2  numbers = range(4)
3  print(list(numbers))    # [0, 1, 2, 3]
4
5  # if 0 or negative number, we get an empty sequence
6  numbers = range(-4)
7  print(list(numbers))    # []
```

Console Shell

```
[0, 1, 2, 3]
[]
```

Range

- การใช้งานแบบ 2 พารามิเตอร์

main.py

```
1  # numbers from 2 to 4 (5 is not included)
2  numbers = range(2, 5)
3  print(list(numbers))    # [2, 3, 4]
4
5  # numbers from -2 to 3 (4 is not included)
6  numbers = range(-2, 4)
7  print(list(numbers))    # [-2, -1, 0, 1, 2]
8
9  # returns an empty sequence of numbers
10 numbers = range(4, 2)
11 print(list(numbers))    # []
12
```

Console Shell

```
[2, 3, 4]
[-2, -1, 0, 1, 2, 3]
[]
```

Range

- การใช้งานแบบ 3 พารามิเตอร์

main.py

```
# numbers from 2 to 10 with increment 3 between
numbers
numbers = range(2, 10, 3)
print(list(numbers))    # [2, 5, 8]

# numbers from 4 to -1 with increment of -1
numbers = range(4, -1, -1)
print(list(numbers))    # [4, 3, 2, 1, 0]

# numbers from 1 to 4 with increment of 1
# range(0, 5, 1) is equivalent to range(5)
numbers = range(0, 5, 1)
print(list(numbers))    # [0, 1, 2, 3, 4]
```

Console Shell

```
[2, 5, 8]
[4, 3, 2, 1, 0]
[0, 1, 2, 3, 4]
>
```

For Loop

- การใช้ range ร่วมกับ For

For Loop

- โปรแกรมก่อนหน้านี้ (ดอกไม้) สามารถเขียนโดยใช้ range ได้ดังนี้
- หาความยาวของ List โดยใช้ Len แล้วนำมาใช้กับ range จากนั้นจึงอ้างถึง List โดยผ่าน Index (ข้อดี คือ สามารถอ้างลำดับจาก Index)

main.py

```
1 flowers = ['calla', 'lily', 'jasmine', \
2              'forget me not', 'sunflower',\
3              'ivy', 'gypso']
4
5 for i in range(len(flowers)):
6     print(flowers[i])
7
8
```

Console Shell

```
calla
lily
jasmine
forget me not
sunflower
ivy
gypso
```

For Loop

- ยังมีอีกวิธีเรียกว่า Enumerate วิธีการนี้จะสามารถได้ค่าคืนมาทั้ง Index และ ข้อมูล

main.py ×

```
1 ▼ flowers = ['calla', 'lily', 'jasmine', \
2                'forget me not', 'sunflower',\
3                'ivy', 'gypso']
4
5 ▼ for i, e in enumerate(flowers):
6       print(f"{i+1}. {e}")
7
8
```

Console Shell

```
1. calla
2. lily
3. jasmine
4. forget me not
5. sunflower
6. ivy
7. gypso
```

For Loop

- การใช้ for กับ list อาจทำกับบางส่วนของ list โดยใช้ slicing ก็ได้
- ฟังก์ชัน title() จะคืนค่า string โดยตัวแรกของทุกคำจะเป็นตัวใหญ่

main.py

```
1 players = ['charles', 'martina', 'michael', \
2              'florence', 'eli']
3
4 print("First three players on my team:")
5 for player in players[:3]:
6     print(player.title())
```

Console Shell

```
First three players on my team:
Charles
Martina
Michael
```

For Loop

- โปรแกรมแสดงสูตรคูณ

main.py

```
1  num = int(input("Enter th number of
   multiplication = "))
2  # use for loop to iterate 12 times
3  for i in range(1, 13):
4      print(f'{num} x {i:2} = {num*i:3}')
5
6
7
8
9
10
11
```

Console Shell

```
Enter th number of multiplication = 5
5 x  1 =   5
5 x  2 =  10
5 x  3 =  15
5 x  4 =  20
5 x  5 =  25
5 x  6 =  30
5 x  7 =  35
5 x  8 =  40
5 x  9 =  45
5 x 10 =  50
5 x 11 =  55
5 x 12 =  60
```

For Loop

- **Exercise 3.4** จากข้อมูลใน List ให้ตรวจสอบว่า ข้อมูลในลิสต์ s เรียงลำดับจากน้อยไปมากหรือไม่ ตอบเป็น True, False

- **Exercise 3.5** ให้ x เป็นลิสต์เก็บคะแนน ต้องการปรับช่องที่มีค่าน้อยกว่า 30 ให้มีค่าเพิ่มอีก 10% แสดงผลลัพธ์ เป็น List

- **Exercise 3.6** ต้องการหาผลรวมของคะแนนที่เก็บในลิสต์ x โดยขอไม่รวมคะแนนที่น้อยสุดอันดับ 2

For Loop

- โปรแกรมนี้ทำอะไร

main.py

```
1  n = int(input("Enter integer number: "))
2  Sum = 0
3  for i in range(2, n + 1):
4      if i % 2 != 0:
5          continue
6      else:
7          Sum = Sum + i
8  print("Sum  = ", Sum)
9
```

For Loop

- โปรแกรมนี้ทำอะไร

main.py ×

```
1  n = int(input("Enter number :"))
2  for i in range(2, n):
3      if n % i == 0:
4          print(n,"is not a prime")
5          break
6      else:
7          print(n, "is a prime")
8
```

For Loop

- **Exercise 3.7** จากข้อมูลใน List ให้แสดง string ที่มีความยาวมากกว่า 2 และมีอักษรตัวแรกกับตัวสุดท้ายเหมือนกัน แสดงบรรทัดละ 1 ตัว
- **Exercise 3.8** จากข้อมูลใน List ให้ลบข้อมูลที่มีซ้ำกันให้เหลือเพียงตัวเดียว ให้แสดงผลเป็น list เช่น [‘a’,’a’] => [‘a’]
- **Exercise 3.9** จากข้อมูลใน List ให้แสดงว่าแต่ละข้อมูลใน List มีจำนวนเท่าไร คือ มีซ้ำกันกี่ตัวใน List ให้แสดงบรรทัดละตัว เช่น

  [1, 2, 3, 4, 1, 2, 3, 1, 2, 1]

  1 = 4

  2 = 3

  3 = 2

  4 = 1

List แบบหลายมิติ

- List สามารถเก็บข้อมูลอะไรก็ได้ รวมทั้ง List เอง อาจเรียกว่า List ซ้อน List หรือ List หลายมิติ สำหรับเก็บข้อมูลที่เป็น Matrix หรือข้อมูลที่มีลักษณะเป็นชุด
- การประมวลผล List หลายมิติ จะต้องใช้ Loop ซ้อน Loop ตามจำนวนมิติ

```
a = [ [2, 4, 6, 8 ],
      [ 1, 3, 5, 7 ],
      [ 8, 6, 4, 2 ],
      [ 7, 5, 3, 1 ] ]

for i in range(len(a)) :
    for j in range(len(a[i])) :
        print(a[i][j], end=" ")
    print()
```

List แบบหลายมิติ

main.py

```python
stds_height =  [[125, 130, 142, 135, 145],   # year 1
                [132, 137, 155, 154, 158],
                [150, 154, 155, 153, 160],
                [152, 153, 156, 151, 160],
                [153, 154, 156, 157, 162],
                [155, 156, 154, 160, 162]]

Total_average = 0
Total_year = 0
Average_each_year = []

for i in stds_height:   # list of each year
    sum = 0
    for j in i:         # each student [125, 130, 142, 135, 145]
        sum = sum + j
    average = sum / len(i)
    Average_each_year.append(average)
    Total_year = Total_year + 1
    Total_average = Total_average + average

print("Total number of years =",Total_year)
for i in range(0, Total_year):
    print("Student height of year",i+1, "=", Average_each_year[i])
print("Overall of student height = %.2f" %(Total_average/Total_year))
```

Console Shell

```
Total number of years = 6
Student height of year 1 = 135.4
Student height of year 2 = 147.2
Student height of year 3 = 154.4
Student height of year 4 = 154.4
Student height of year 5 = 156.4
Student height of year 6 = 157.4
Overall of student height = 150.87
```

List แบบหลายมิติ

- โปรแกรมคูณ Matrix

main.py

```
a = [[4, 1, 7],
     [2, 4, 8],
     [3, 7, 1]]

b = [[6, 8, 1],
     [9, 7, 5],
     [2, 4, 3]]

c = [[0, 0, 0],
     [0, 0, 0],
     [0, 0, 0]]

for i in range(len(a)):
    for j in range(len(b[0])):
        for k in range(len(b)):
            c[i][j] += a[i][k] * b[k][j]

for r in c:
    print(r)
```

Console Shell

```
[47, 67, 30]
[64, 76, 46]
[83, 77, 41]
```

For your attention