More on Defining a function

Recap

- การนิยามฟังก์ชัน คือการตั้งชื่อให้ กลุ่มของคำสั่ง เพื่อที่เราจะสามารถนำกลุ่มของคำสั่งนั้นกลับมาใช้ได้อีก
- ในภาษา Python สามารถทำได้โดยใช้ syntax ต่อไปนี้

```
def functionName(parameter1, parameter2, ...):
        process1
        process2
          . . .
        processN
```

กลุ่มคำสั่ง

- ส่วนของ parameter จะมีหรือไม่มีก็ได้ แล้วแต่หน้าที่ของฟังก์ชัน
- Process แต่ละตัวที่อยู่ในฟังก์ชันต้องทำการย่อหน้าให้ถูกต้อง

Variable scope

- Global variable

  หมายความถึง ตัวแปรที่ประกาศไว้ **นอก** function ซึ่งฟังก์ชันทุกตัวสามารถเรียกใช้งานได้

- Local variable

  หมายความถึง ตัวแปรที่ประกาศไว้ **ใน** function *รวมถึงตัวแปรที่ทำหน้าที่เป็น argument ด้วย* จะมีแต่ฟังก์ชันที่เป็นเจ้าของเท่านั้นที่เรียกใช้งานได้

โปรแกรมต่อไปนี้ทำงานได้หรือไม่

```
globalGift = "Diamond"


def HappyBirthday(person):
    localGift = "Rock"
    if (person.lower() == 'je'):
        return globalGift
    elif (person.lower() == 'yoon'):
        return localGift
    else:
        return "Nothing"

gift = HappyBirthday('je')
print("I get ", gift, " as a gift")

print("Yoon will get ", localGift, "as a gift")
```

โปรแกรมต่อไปนี้ทำงานได้หรือไม่

```
1  globalGift = "Diamond"
2
3
4  def HappyBirthday(person):
5      localGift = "Rock"
6      if (person.lower() == 'je'):
7          return globalGift
8      elif (person.lower() == 'yoon'):
9          return localGift
10     else:
11         return "Nothing"
12
13 gift = HappyBirthday('je')
14 print("I get ", gift, " as a gift")
15
16 print("Yoon will get ", localGift, "as a gift")
```

local

local

global

เรียกใช้ไม่ได้เพราะเป็น local

What happens to function argument?

- จะเกิดอะไรขึ้นกับค่าตัวแปรที่ส่งเป็น argument เข้าไปในฟังก์ชัน แล้วมีการเปลี่ยนแปลงค่าของ argument ภายในฟังก์ชัน

Experiment 1: โปรแกรมถูกต้องไหม

```
1  def HappyBirthday(person, age):
2      print("Happy Birthday my dear ", person)
3      age = age + 1
4
5  name = input("What's your name?: ")
6  age  = int(input("How old are you? "))
7
8  HappyBirthday(name, age)
9
10 print(name, ", you are now", age, "year-old")
```

Experiment 2: แล้วโปรแกรมนี้ล่ะ

```python
def getMarried(brideName, groomName):
    brideName[1] = groomName[1]

# Input name in the form 'Firstname Lastname'
# What we get from split(' ') is a list in the form ['firstname', 'lastname']
brideName = input("What's the name of the bride: ").split(' ')
groomName = input("What's the name of the groom: ").split(' ')

getMarried(brideName, groomName)

# ' '.join(brideName) means joining elements of brideName with ' ' (space)
print("Congratulations ", ' '.join(brideName), " and ", ' '.join(groomName))
```

ตัวแปรใน Python มีสองแบบ

Immutable

- การเปลี่ยนแปลงต่อ argument ที่เกิดขึ้นใน function จะคงอยู่เฉพาะในฟังก์ชันเท่านั้น
- ตัวอย่างของตัวแปร Immutable
  - Int, float, long, complex (ใน experiment 1)
  - Str
  - Tuple
  - Bytes

Mutable

- การเปลี่ยนแปลงต่อ argument ที่เกิดขึ้นใน function จะส่งผลต่อภายนอกด้วย
- ตัวอย่างของตัวแปร mutable
  - List (ใน experiment 2)
  - Set
  - Dict

Exercise 1

- จงเขียนฟังก์ชัน min_of_three(a,b,c) ซึ่งรับ argument เป็นตัวเลขจำนวนเต็ม สามตัว แล้ว return ตัวเลขที่มีค่าน้อยที่สุดกลับคืนมา
- ตัวอย่าง ผลลัพธ์

```
>>> min_of_three(7,3,4)
3
>>> min_of_three(10,100,-3)
-3
>>> min_of_three(10,0,-9)
-9
```

template

```
def min_of_three(a,b,c):
    ……
    ……
    return result

n1=int(input(“Input number1: ”))
n2=int(input(“Input number2: ”))
n3=int(input(“Input number3: ”))
ans= min_of_three(n1,n2,n3)
print(ans)
```

Exercise 2

- โปรแกรม word processing อย่าง Microsoft office มีฟังก์ชันที่ผู้ใช้สามารถจัดย่อหน้าในสวยงามได้ทั้ง จัดชิดซ้าย จัดชิดขวา และจัดกึ่งกลาง แบบฝึกหัดนี้ ให้เขียนฟังก์ชันชื่อ right_justify(string) เพื่อรับข้อความหนึ่งข้อความ (a string) เข้ามา แล้วทำการจัดชิดขวา ให้สวยงาม กำหนดให้หน้ากระดาษมีความกว้าง 50 whitespaces และข้อความที่เข้ามามีความยาวไม่เกิน 50 อักขระ ฟังก์ชันนี้ไม่มีการ return ค่ากลับ

Exercise 2: Example

```
>>> right_justify("hello")
                                                  hello
>>> right_justify("longer than hello")
                                        longer than hello
>>> right_justify("longer than the longest hello")
                             longer than the longest hello
>>>
```

template

```
def right_justify(s):
    ……
    ……

st=input(“Input text: ”)
right_justify(st)
```

Exercise 2: String built-in functions

- string built-in functions ฟังก์ชันสำเร็จรูปในการจัดการร่วมกับสตริง ได้แก่ ฟังก์ชัน ljust(), rjust() และ center() โดยฟังกัชันแต่ละตัวมีรูปแบบไวยากรณ์ (syntax) ที่คล้ายกันดังนี้

  center(len, fillchar) , rjust(len, fillchar) , ljust(len, fillchar)

  โดยที่ len คือ ความกว้างของหน้ากระดาษที่ต้องการจัดตัวอักษร

  fillchar (เป็น option หรือตัวเสริมไม่ใช้ก็ได้) คือ อักขระที่ต้องการนำมาเติมเต็มช่องว่างที่เหลือเมื่อมีการจัดตำแหน่งข้อความที่ต้องการแล้ว

ตัวอย่างการใช้ฟังก์ชัน ljust() : string_in = “Hello Python”

string_in.ljust(20) ผลลัพธ์คือ ‘Hello Python ’

string_in.ljust(20,’#’) ผลลัพธ์คือ ‘Hello Python########## ’

References

- Hands-on Python tutorial
  - anh.cs.luc.edu/python/hands-on/3.1/handsonHtml/functions.html
- Think Python: How to Think Like a Computer Scientist