

Defining a function

Recap

- จากบทที่แล้ว เราเรียกใช้ฟังก์ชันเพียงอย่างเดียว
- บทนี้ เราจะมาเรียนรู้การเขียน (นิยาม) ฟังก์ชัน
- ทบทวน: สิ่งที่ต้องคำนึงถึงตอนเรียกใช้ฟังก์ชัน มี 3 ประการได้แก่
 - ชื่อ และหน้าที่ของฟังก์ชันที่ต้องการจะเรียก
 - ฟังก์ชันนั้นต้องการ **argument** อะไรบ้าง และมีลำดับของ argument อย่างไร
 - ฟังก์ชันนั้นมี **return value** ส่งกลับคืนมาหรือไม่
- ตัวอย่างการเรียกใช้ฟังก์ชัน เพื่อหาความยาวของสตริง (สังเกตสีด้วย)
 - **lengthOfString** = **len**("Howdy?")

Defining a function

- การนิยามฟังก์ชัน คือการตั้งชื่อให้ กลุ่มของคำสั่ง เพื่อที่เราจะสามารถนำกลุ่มของคำสั่งนั้นกลับมาใช้ได้อีก
- ในภาษา Python สามารถทำได้โดยใช้ syntax ต่อไปนี้

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

} กลุ่มคำสั่ง

- ส่วนของ parameter จะมีหรือไม่มีก็ได้ แล้วแต่หน้าที่ของฟังก์ชัน
- Process แต่ละตัวที่อยู่ในฟังก์ชันต้องทำการย่อหน้าให้ถูกต้อง

วันนี้วันเกิดเจ้

- เราอยากเขียนโปรแกรมไพธอนเพื่อร้องเพลงวันเกิดให้เจ้

```
1 print("Happy Birthday to You")
2 print("Happy Birthday to You")
3 print("Happy Birthday, my dear Je")
4 print("Happy Birthday to You")
5
6
```

Our very first function

- เราต้องการ ตั้งชื่อให้กลุ่มคำสั่งดังกล่าว เพื่อบอกว่าเป็นเพลงวันเกิดของเจ เราสามารถทำได้โดย

```
1 def HappyBirthdayJe():  
2     print("Happy Birthday to You")  
3     print("Happy Birthday to You")  
4     print("Happy Birthday, my dear Je")  
5     print("Happy Birthday to You")  
6
```

- หากลองพิมพ์ใน IDLE แล้วรันดู โปรแกรมจะไม่ปรี้นท์ค่าอะไรเลย
 - เพราะว่าเราแค่ define ฟังก์ชัน แต่ยังไม่ได้ทำการ call ฟังก์ชัน
 - ทำยังไงดี
-
- การเรียกใช้งานฟังก์ชันทำได้โดยการอ้างชื่อฟังก์ชัน
 - เมื่อฟังก์ชันถูกเรียกใช้มีผลให้กลุ่มคำสั่งของฟังก์ชันทำงาน 1 ครั้ง

Calling a function we've just defined

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 HappyBirthdayJe()
8
9
```

ลองทำตัวเป็น Python interpreter ดู

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 HappyBirthdayJe()
8
9
```

1. เริ่มจากบรรทัดที่ 1 เจอ คำว่า def แสดงว่า programmer กำลังจะนิยามฟังก์ชัน
2. อ่านบรรทัดถัดไปเรื่อยๆจนกว่าจะหมด indentation
3. สิ้นสุด indentation ที่บรรทัดที่ 6 แสดงว่าฟังก์ชัน ที่ programmer ต้องการนิยามคือกลุ่มคำสั่ง print 4 คำสั่งนี้ โดยเค้าอยากให้ชื่อว่า HappyBirthdayJe
4. จำไว้เฉยๆ แต่ยังไม่ทำการ print
5. อ่านมาเรื่อยๆเจอบรรทัดที่ 7 มีการอ้างอิงถึงฟังก์ชัน HappyBirthdayJe
6. กลับไปทบทวนว่าเราเคยเจอฟังก์ชันนี้มาก่อนรึเปล่า
7. เมื่อพบว่าเคยนิยามไว้ ก็ทำการ run ฟังก์ชันจริงๆ โดยเริ่มจากบรรทัดที่ 2 ถึง 5
8. จบการทำงาน

ระว่างการเว้นวรรคด้วย

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 HappyBirthdayJe()
8
```

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5 print("Happy Birthday to You")
6
7 HappyBirthdayJe()
8
```

อยากร้องเพลงให้สองรอบ

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 HappyBirthdayJe()
8 HappyBirthdayJe()
9
```

อยากร้องเพลงให้สิรอบ

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 HappyBirthdayJe()
8 HappyBirthdayJe()
9 HappyBirthdayJe()
10 HappyBirthdayJe()
11
```

อยากร้องเพลงให้เ้ารอบ

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 for i in range(1,10):
8     HappyBirthdayJe()
9
```

อยากร้องให้ยูนบ้าง

- ก็สร้างฟังก์ชันให้ยูนสิ

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 def HappyBirthdayYoon():
8     print("Happy Birthday to You")
9     print("Happy Birthday to You")
10    print("Happy Birthday, my dear Yoon")
11    print("Happy Birthday to You")
12
13 HappyBirthdayJe()
14 HappyBirthdayYoon()
15
```

Functions within a function

- เราสามารถเรียกฟังก์ชัน ภายในฟังก์ชันตัวอื่นที่เรานิยามได้ เช่น

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6
7 def HappyBirthdayYoon():
8     print("Happy Birthday to You")
9     print("Happy Birthday to You")
10    print("Happy Birthday, my dear Yoon")
11    print("Happy Birthday to You")
12
13 def HappyBirthdayJeAndYoon():
14     HappyBirthdayJe()
15     HappyBirthdayYoon()
16
17 HappyBirthdayJeAndYoon()
18
```

แล้วพีสุชาติล่ะ?

- เราสามารถนิยามฟังก์ชันเพลงวันเกิดให้พีสุชาติอีกได้
- แล้ว ของหมอฮิม ของไก่ ของพีเป้ง ของ.....อีกหลายคนล่ะ
- แทนที่จะเขียนฟังก์ชันเพลงวันเกิดให้แต่ละคน
- เราจะเขียนฟังก์ชันที่รับชื่อคนเข้าไปแล้วร้องเพลงให้คนนั้น

- ฟังก์ชันสามารถรับข้อมูลเข้าไปเพื่อประมวลผลเพื่อให้ได้ผลลัพธ์ที่ต้องการได้
- ในการรันโปรแกรมแต่ละครั้งเมื่อข้อมูลเข้าเปลี่ยนค่าไปจะทำให้ผลลัพธ์มีค่าเปลี่ยนแปลงไปในลักษณะที่สอดคล้องกัน
- ข้อมูลเข้าที่ส่งให้กับฟังก์ชันเรียกว่า พารามิเตอร์ (parameter) มีรูปแบบดังนี้
def ชื่อฟังก์ชัน(พารามิเตอร์1, พารามิเตอร์2, ...):
 กลุ่มของคำสั่ง

- ฟังก์ชันแต่ละฟังก์ชัน อาจจะมีจำนวนพารามิเตอร์ที่ต่างกัน การเรียกใช้ฟังก์ชันที่มีพารามิเตอร์ จะต้องมีการส่งค่าอาร์กิวเมนต์ (argument) เพื่อไปเป็นค่าพารามิเตอร์ของฟังก์ชัน
- อาร์กิวเมนต์ หมายถึงค่าที่ส่งผ่านจุดที่เรียกใช้คำสั่งหรือฟังก์ชันเข้าสู่การทำงานในคำสั่งหรือฟังก์ชันนั้นๆ

Function with one argument

```
1 def HappyBirthday(person):
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday,my dear ",person," ")
5     print("Happy Birthday to You")
6
7 HappyBirthday('Je')
8 HappyBirthday('Yoon')
9 HappyBirthday('PSuchart')
10
```

Function with one argument

```
1 def HappyBirthday(person):
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday,my dear ",person," ")
5     print("Happy Birthday to You")
6
7 HappyBirthday('Je')
8 HappyBirthday('Yoon')
9 HappyBirthday('PSuchart')
10
```

The diagram illustrates the flow of data in a function call. A green circle highlights the argument 'Je' in the function call on line 7. A green arrow points from this circle to the parameter 'person' in the function definition on line 1. A purple arrow points from the parameter 'person' in the function definition to the variable 'person' in the print statement on line 4, indicating how the argument is passed to the function and used within its body.

หรือรับชื่อจาก user

```
1 def HappyBirthday(person):
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday,my dear ",person," ")
5     print("Happy Birthday to You")
6
7 name = input("What's your name?: ")
8 HappyBirthday(name)
9
```

อยากร้องเพลงในโอกาสอื่นๆด้วย?

- เพิ่ม argument เข้าไปอีกซักตัวดีมั้ย

```
1 def HappyBirthday(person, occasion):
2     print("Happy ", occasion, " to You")
3     print("Happy ", occasion, " to You")
4     print("Happy ", occasion, " my dear ", person, " ")
5     print("Happy ", occasion, " to You")
6
7 name = input("What's your name?: ")
8 day  = input("In what occasion?: ")
9
10 HappyBirthday(name, day)
11
```

```
What's your name?: Suchart
In what occasion?: Monday
Happy Monday to You
Happy Monday to You
Happy Monday my dear Suchart
Happy Monday to You
```

ตัวอย่างฟังก์ชันเพื่อหาค่า BMI

```
def bmi_cal(name, weight, height):
```

```
    bmi = weight / height ** 2
```

```
    print('--Hello', name)
```

```
    print('Your BMI is',bmi)
```

```
print('BMI Calculation')
```

```
yourname = input('Enter your name')
```

```
yourweight = float(input('Enter your weight(kg)'))
```

```
yourheight = float(input('Enter your height(m)'))
```

```
bmi_cal(yourname,yourweight,yourheight)
```

การคืนค่าจากฟังก์ชัน

- ในการเรียกใช้งานฟังก์ชันบางครั้งเราต้องการให้ฟังก์ชันคืนค่าที่คำนวณได้มาให้ เพื่อเอาค่าที่คำนวณได้ไปใช้งานภายหลังสามารถทำได้โดยใช้รูปแบบดังนี้

return นิพจน์

```
def bmi_cal(weight, height):
```

```
    bmi = weight / height ** 2
```

```
    return bmi
```


Function with return value

- สามารถทำได้โดย ใช้ keyword: **return**

```
1 def HappyBirthdayJe():
2     print("Happy Birthday to You")
3     print("Happy Birthday to You")
4     print("Happy Birthday, my dear Je")
5     print("Happy Birthday to You")
6     return "Diamond"
7
8 gift = HappyBirthdayJe()
9 print("I get ", gift, " as a gift")
```

```
Happy Birthday to You
Happy Birthday to You
Happy Birthday, my dear Je
Happy Birthday to You
I get Diamond as a gift
```

So far so good

- แต่ถ้ามีคนเผลอเรียกใช้ฟังก์ชันของเราแบบนี้ล่ะ -- ระวังชนิดของ argument ด้วย

```
1 def HappyBirthday(person):
2     if (person.lower() == 'je'):
3         return "Diamond"
4     elif (person.lower() == 'yoon'):
5         return "MacBook"
6     else:
7         return "Nothing"
8
9 gift = HappyBirthday(2)
10 print("I get ", gift, " as a gift")
11
```

```
Traceback (most recent call last):
  File "rr.py", line 9, in <module>
    gift = HappyBirthday(2)
  File "rr.py", line 2, in HappyBirthday
    if (person.lower() == 'je'):
AttributeError: 'int' object has no attribute 'lower'
```

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