Lecture 14 Functions

15 ตุลาคม 2557 11:15

การเขียนโปรแกรมจะเริ่มมีขนาดใหญ่ขึ้น คือ จำนวนบรรทัดของโค้ดและมีความซับซ้อนมากขึ้น ดังนั้น เราจำเป็นต้องมีวิธีจัดการ การย่อยให้เป็นส่วนๆ เพื่อการดูแล ตรวจสอบและค้นหาได้ง่ายจึงมีความสำคัญ
เรามีวิธีจัดการเหล่านี้ในการเขียนโปรแกรมอยู่ 3 รูปแบบ ดังนี้

- Function ฟังก์ชั่น คือกลุ่มของโค้ด หรือที่เราเรียกว่า บล๊อค (block) ของโค้ด ที่เราสามารถเรียกใช้ได้หลายครั้งได้
- Objects อ็อปเจ็ค การห่อหุ้มข้อมูลและส่วนการจัดการข้อมูล
- Modules โมดูล การแยกไฟล์โค้ด

ในบทนี้เราจะสนใจเพียงแค่ฟังก์ชัน

ฟังก์ชัน - Function

คือการสร้างกลุ่มของโค้ดเพื่อให้ทำงานตามจุดประสงค์ที่ต้องการ เราสามารถใช้ส่วนเล็กๆ นี้ เพื่อนำมาประกอบเป็นโปรแกรมใหญ่ๆ เหมือนเวลาเราประกอบรถยนต์ เราอาจแบ่งออกเป็นชิ้นส่วนใหญ่ คือ เครื่องยนต์ และ โครงรถ และเครื่องยนต์จะประกอบไปด้วยชิ้นส่วนอื่นๆ ย่อยอีกที เกียว ลูกสูบ เหมือนกับการสร้างแบ่งส่วนแล้วนำมาประกอบกัน การแบ่งส่วนนี้เราเรียกว่าการสร้าง กลุ่มของคำสั่งหรือฟังก์ชัน เราสร้างส่วนย่อยของเราโดยใช้คำสั่ง def ย่อมาจาก define แปลว่าประกาศ และเราเรียกใช้ call โดยใช้ชื่อของฟังก์ชันที่เราประกาศ

การสร้างฟังก์ชัน Creating a function

สร้างฟังก์ชันพิมพ์ที่อยู่ บนหน้าจอ

```
def printMyAddress():
    print("Sarayut Gonwirat")
    print("153 Moo 5")
    print("Chumporn")
    print("Maeywadee")
    print("Roi-et")
    print("45250")

printMyAddress()
```

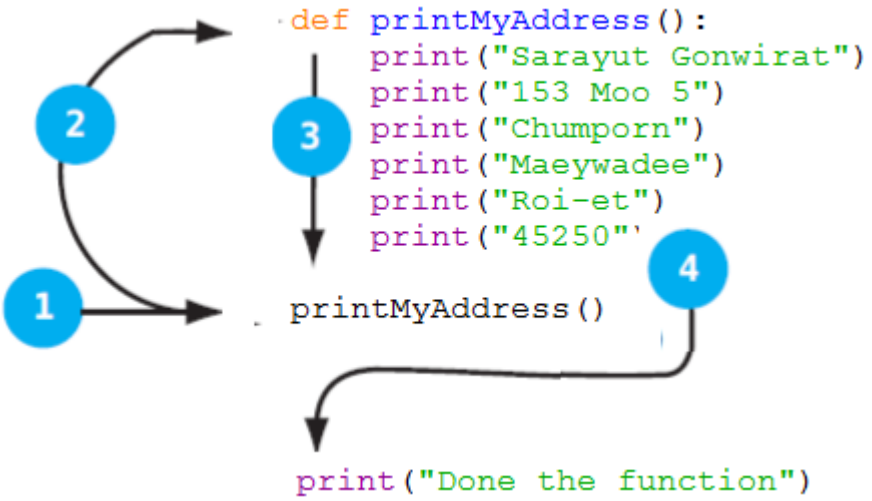
ผลลัพท์ที่ได้

```
>>>
Sarayut Gonwirat
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
```

เราสังเกตุได้ว่าจะมีโคลอล : เช่นเดียวกับ if , while และ for เพราะเป็นการสร้างกลุ่มของโค้ด block of codes
บรรทัดสุดท้ายเราเรียกใช้ function ที่ประกาศไว้ คือ

```
printMyAddress()
```

ทำให้ได้ผลลัพท์ออกบนหน้าจอ ถ้าไม่มีบรรทัดนี้จะไม่มีการแสดงผลใดๆ โค้ดที่เราประกาศฟังก์ชันจะไม่ทำงานใดๆ
จนกว่าเราจะเรียกใช้ (call)



ขั้นตอนการทำงาน

1. จุดเริ่มโปรแกรม การรันจะผ่านการประกาศ def มาและมาถึงจุดนี้
2. เมื่อเรียกใช้ฟังก์ชัน โปรแกรมจะย้อนไปยังฟังก์ชันที่เราประกาศตามชื่อไว้
3. จากนั้นจะเข้าไปรันในฟังก์ชันทีละบรรทัด
4. และเมื่อเสร็จในฟังก์ชัน จะย้อนกลับมาทำต่อจากจุดที่เราเรียกใช้

จากผลลัพท์ ถ้าเราทำการเรียกใช้หลายครั้ง ตัวอย่างเช่น

```
printMyAddress()
print("========")
printMyAddress()
print("========")
printMyAddress()
print("========")
print("Done the function")
```

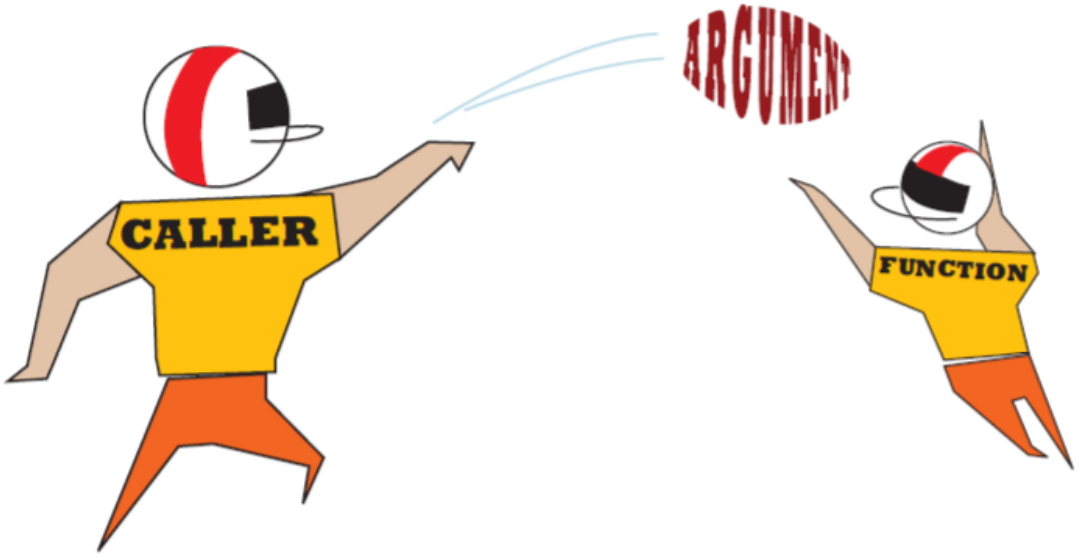
ผลลัพท์ที่ได้กลุ่มการพิมพ์ค่าหน้าจอจะซ้ำกันหลายครั้ง เนื่องจากเราเรียกใช้กลุ่มของโค้ดพิมพ์

```
>>>
Sarayut Gonwirat
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
========
Sarayut Gonwirat
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
========
Sarayut Gonwirat
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
========
Done the function
```

เราอาจจะใช้ loop แทนได้ในตัวอย่าง แต่ความจริงแล้วเราอาจจะไม่ได้วนในที่เดียวกันอย่างนี้ บางครั้งเราอาจเรียกใช้ครั้งเดียวในที่หลายครั้ง

ในตัวอย่างนี้เราแสดงให้เห็นการเรียกใช้สิ่งที่ให้ผลลัพท์เหมือนเดิมสามครั้งแต่ถ้าเราต้องการผลลัพท์ที่ต่างกันในแต่ละครั้งเราจะทำอย่างไร

การส่ง arguments ให้ฟังก์ชัน



อาร์กูเมนต์ คือ บางสิ่งที่เราส่งไปให้ฟังก์ชันนำไปใช้งานต่อ เราเรียกว่า ส่ง (pass) โดยเราจะเห็นตอนเรียกใช้เรามีวงเล็บ เราจะใช้วงเล็บนั้นช่วยครอบค่าทีเราต้องการส่ง ตัวอย่างที่แล้วเราไม่ได้ส่งค่าอะไรไป แต่ถ้าเราต้องฟังก์ชันที่พิมพ์ชื่อของสมาชิกในบ้านเดียวกัน คือมีที่อยู่เดียวกัน ต่างกันเพียงชื่อ เราจำเป็นจะต้องสร้างสามฟังก์ชันสำหรับสามคนหรือไม่ หรือเราสามารถสร้างอันเดียวแต่สามารถระบุชื่อเข้าไปด้วยได้อย่างไร ดูตัวอย่างต่อไปนี้

```
def printMyAddress(name):
    print(name)
    print("153 Moo 5")
    print("Chumporn")
    print("Maeywadee")
    print("Roi-et")
    print("45250")

printMyAddress("Sarayut Gonwirat")
print("Done the function")
```

จากโค้ด ในฟังก์ชันเราไม่ได้พิมพ์ชื่อตรงๆ ลงไป แต่เราพิมพ์ตัวแปร name ที่ประกาศต่อจากชื่อฟังก์ชันในวงเล็บ เวลาเราเรียกใช้เราส่งค่าชื่อเข้าไป "Sarayut Gonwirat" ซึ่งค่านี้จะถูกไปเก็บไว้ที่ตัวแปร name เมื่อเราพิมพ์ตัวแปร ค่านี้จะแสดงผลออกบนหน้าจอดังนี้

```
>>>
Sarayut Gonwirat
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
Done the function
```

เมื่อทดสอบเรียกใช้หลายครั้งและเปลี่ยนค่าที่ส่งเข้าไป จะให้ผลลัพท์ไม่เหมือนกัน ดังนี้

```
printMyAddress("Sarayut Gonwirat")
print("========")
printMyAddress("My dad")
print("========")
printMyAddress("My mom")
print("========")
print("Done the function")
```

```
>>>
Sarayut Gonwirat
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
========
My dad
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
========
My mom
153 Moo 5
Chumporn
Maeywadee
Roi-et
45250
========
Done the function
```

การส่งค่ามากกว่าสองค่า

ตอนนี้เราส่งเพียงแค่ชื่อให้ไปแสดงผลต่างกัน ถ้าโปรแกรมต้องการแสดงผลที่ต่างกันมากกว่าหนึ่งอย่าง หรือรับส่งค่าระหว่างผู้ใช้มากกว่าหนึ่งตัวแปร ตัวอย่างเช่น ให้สร้างฟังก์ชันแนะนำชื่อและอายุของตัวเองที่สามารถต่างกันได้ เราจะเขียนโปรแกรมดังนี้

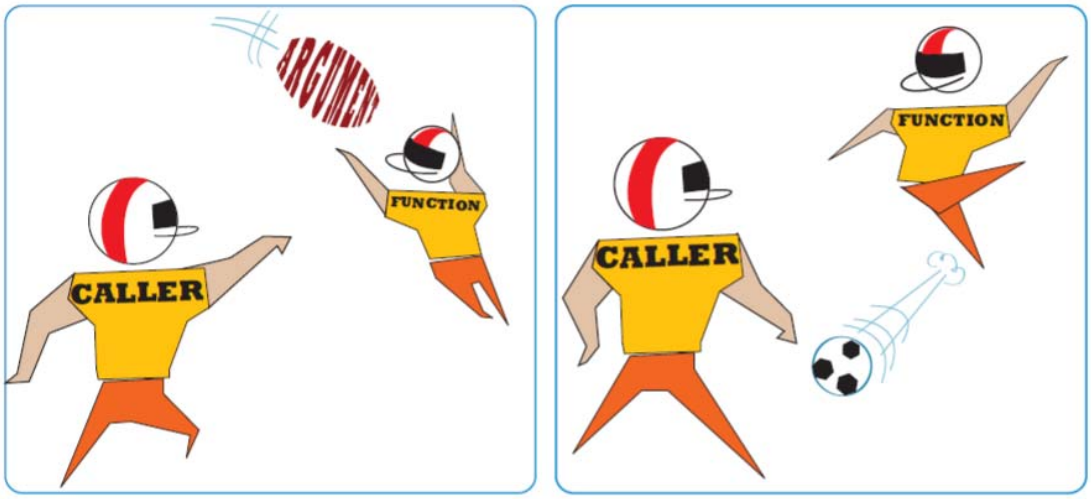
```
def printIntroduction(name, age):
    print("My name is",name)
    print("I 'm",age, "years old.")
    print("Nice to meet you.")

printIntroduction("Bank", 30)
print("============")
printIntroduction("Goft", 18)
print("============")
printIntroduction("Beer", 18)
```

จะเห็นได้ว่าเวลาประกาศจะใช้คอมม่าขั้นระหว่างตัวแปร และตอนเรียกใช้ ก็จะส่งค่าเพิ่มอีกตัวคืออายุ เพื่อให้ตรงกันระหว่างการส่งและการรับ จากนั้นโปรแกรมจะรับค่าที่เราเรียกใช้ ไปแสดงผลลัพท์ได้ดังนี้

```
>>>
My name is Bank
I 'm 30 years old.
Nice to meet you.
============
My name is Goft
I 'm 18 years old.
Nice to meet you.
============
My name is Beer
I 'm 18 years old.
Nice to meet you.
```

อีกคำศัพท์ที่เราต้องรู้คือ พารามิเตอร์ (parameter) คือตัวแปรที่อยู่ในฟังก์ชันไว้รับค่า argument ที่เราส่งเข้ามา เช่น
ในตัวอย่าง name และ age คือ พารามิเตอร์ที่รอรับค่าของ arguments ที่ส่งเข้ามา ส่วน "Bank" และ 30 คือ ค่าอาร์กิวเมนต์ที่ส่งไปให้ฟังก์ชันคำนวณ

การส่งค่ากลับ return a value



การเขียนฟังก์ชันจะมีประโยชน์มากขึ้นเมื่อเราส่งค่ากลับได้ (return) จากการคำนวณในฟังก์ชันรูปแบบใดรูปแบบหนึ่งจนได้ผลลัพท์ (result) เช่นการคำนวณหาส่วนลดดังนี้

```
def discount(price, percent):
    dis = price * (percent/100)
    return dis
```

เราเรียกใช้ ส่งค่าเข้าไปคำนวณ และสร้างตัวแปรมารับค่า

```
disPrice = discount(500, 10)
print("Discount 10% of 500 =", round(disPrice, 2))
```

ได้ผลลัพท์ดังนี้

```
>>>
Discount 10% of 500 = 50.0
```

เราอาจทดสอบเพิ่มเติม

```
>>> print(discount(100, 3))
3.0
>>> total = discount(500, 20) + discount(1000, 30)
>>> print(total)
400.0
```

ลองเขียนโปรแกรมให้เต็ม โดยรับค่าอินพุตราคาและส่วนลดดังนี้

```
def discount(price, percent):
    dis = price * (percent/100)
    return dis

price = float(input("Enter Price: "))
percentDis = float(input("Percent Discount: "))
disPrice = discount(price, percentDis)
print("Discount 10% of 500 =", round(disPrice, 2))
```

ผลลัพท์ที่ได้

```
>>>
Enter Price: 1000
Percent Discount: 10
Discount 10% of 500 = 100.0
```

ขอบเขตของตัวแปร Variable Scope

ตัวแปรที่สร้างอยู่ในฟังก์ชันจะมีอายุการใช้งานแค่ในฟังก์ชันเท่านั้น เพราะเป็นการประกาศในขอบเขตของบล็อคนี้ เมื่อออกจากบล็อกจะโดนทำลายทิ้ง เราเรียกตัวแปรพวกนี้ว่า ตัวแปรพื้นที่ **โลคอล(local variable)** เช่นตัวแปร dis ในตัวอย่างนี้
ถูกสร้างในฟังก์ชัน discount และถูกเรียกใช้ข้างนอกฟังก์ชัน

```
def discount(price, percent):
    dis = price * (percent/100)
    return dis

disPrice = discount(500, 10)
print("Discount 10% of 500 =", round(disPrice, 2))
print(dis)
```

ผลลัพท์ที่ได้คือ

```
>>>
Discount 10% of 500 = 50.0
Traceback (most recent call last):
  File "C:/Python33/ComLang/chapter 13.py", line
>
    print(dis)
NameError: name 'dis' is not defined
```

Error ที่เกิดขึ้นบอกว่า dis ไม่ได้ประกาศตัวแปร ทั้งที่เราสร้างตัวแปรนี้ในฟังก์ชัน แต่เนื่องจากเป็น local variable อายุจะอยู่เพียงด้านในของฟังก์ชันเท่านั้น เมื่อนำมาใช้ภายนอกจะมองไม่เห็นเนื่องจากถูกทำลายไปแล้ว

ตัวแปรโกลบอล (Global variables) อยู่ในขอบเขตด้านนอกเช่น my_price เมื่อใช้ในฟังก์ชัน จะสามารถมองเห็นได้ เนื่องจากเป็นตัวแปรที่อยู่ขอบเขตด้านนอก ด้านในสามารถมองเห็น ในทางกลับกันด้านนอกจะไม่สามารถมองเห็นตัวแปรด้านในได้เนื่องจากโดนทำลายทิ้งไปเมื่อจบการทำงานของ

ฟังก์ชัน แต่ด้านนอกตัวแปรจะยังคงมีชีวิตอยู่เมื่อเข้ามาในฟังก์ชันจะยังไม่โดนทำลายทิ้งตัวแปรโกบอลจะโดนทำลายทิ้งก็ต่อเมื่อปิดโปรแกรมหรือโปรแกรมจบการทำงาน

```
def discount(price, percent):
    dis = price * (percent/100)
    print("My price =", my_price)
    return dis
my_price = float(input("Enter price"))
disPrice = discount(500, 10)
print("Discount 10% of 500 =", round(disPrice, 2))
```

ผลลัพท์

```
>>>
Enter price100
My price = 100.0
Discount 10% of 500 = 50.0
```

ข้อควรระวัง

ถ้าเราต้องการเปลี่ยนตัวแปรโกลบอลในฟังก์ชันจะไม่ได้ผล เช่น

```
def discount(price, percent):
    dis = price * (percent/100)
    my_price = 1000
    return dis
my_price = float(input("Enter price"))
disPrice = discount(500, 10)
print("Discount 10% of 500 =", round(disPrice, 2))
print("My price =", my_price)
```

เราได้เปลี่ยน my_price = 1000 แต่ผลลัพท์ยังคงเหมือนเดิม ดังนี้

```
>>>
Enter price10
Discount 10% of 500 = 50.0
My price = 10.0
```

เนื่องจากเมื่อเราให้ค่า my_price = 1000 จะไม่ใช้การใช้ตัวแปรโกลบอล แต่เป็นเพียงการสร้างตัวแปรโลคอลใหม่ขึ้นมาและเมื่อให้ค่าใหม่เป็น 1000 ตัวแปรใหม่ที่มีเพียงชื่อเหมือนกันกับตัวแปรโกลบอลจะได้รับค่าใหม่ แต่ตัวแปรโกลบอลค่ายังเหมือนเดิม

แล้วเรามีวิธีไหนที่จะเปลี่ยนตัวแปรโกลบอลจากด้านในฟังก์ชันได้หรือไม่ เราจะเปลี่ยนได้เมื่อเราใช้คำว่า global ดังตัวอย่างนี้

```
def discount(price, percent):
    dis = price * (percent/100)
    global my_price
    my_price = 1000
    return dis
my_price = float(input("Enter price"))
disPrice = discount(500, 10)
print("Discount 10% of 500 =", round(disPrice, 2))
print("My price =", my_price)
```

ผลลัพท์คือ

```
>>>
Enter price10
Discount 10% of 500 = 50.0
My price = 1000
```

ทบทวน

- ฟังก์ชันคืออะไร การตั้งชื่อกลุ่มของโค้ด เพื่อเรียกใช้ได้ง่าย

- อาร์กิวเมนต์คืออะไร ค่าที่ส่งให้ฟังก์ชันคำนวณ
- การส่งค่าอาร์กิวเมนต์อย่างไร
- การส่างค่าอาร์กิวเมนต์หลายตัว
- เราส่งค่ากลับอย่างไร
- ขอบเขตของตัวแปร local และ global
- เราเปลี่ยนค่า global ในฟังก์ชันอย่างไร

แบบทดสอบความรู้

1. คำไหนที่ใช้ในการประกาศฟังก์ชัน
2. จงยกตัวอย่างการเรียกใช้ฟังก์ชัน
3. เราส่งข้อมูลให้ฟังก์ชันอย่างไร จงยกตัวอย่าง
4. จำนวนค่าที่ส่งให้ฟังก์ชันสูงสุดกี่ค่า
5. เรารับค่าคืนจากฟังก์ชันอย่างไร
6. จะเกิดอะไรขึ้นถ้าเราใช้ตัวแปรที่อยู่ในฟังก์ชัน ด้านนอกฟังก์ชัน

จงเขียนโปรแกรมต่อไปนี้

1. จงสร้างฟังก์ชันที่พิมพ์ชื่อเล่นตัวเองออกบนหน้าจอ ดังตัวอย่างต่อไปนี้

```
   CCCC        A        RRRRR  TTTTTTT  EEEEEE  RRRRR
  C    C      A A       R    R    T     E       R    R
 C           A   A      R    R    T     EEEE    R    R
 C          AAAAAAA     RRRRR     T     E       RRRRR
  C    C  A        A    R    R    T     E       R    R
   CCCC  A          A   R    R    T     EEEEEE  R    R
```

2. สร้างฟังก์ชันที่สามารถพิมพ์ชื่อ(name) บ้านเลขที่ (address) ตำบล(sub distinct) อำเภอ (distinct) จังหวัด(province) ออกบนหน้าจอ โดยจะป้อนค่าอะไร ลงไปแสดงก็ได้
3. จากตัวอย่างโค้ดเรื่องส่วนลดในเนื้อหา จงลองเปลี่ยนค่า my_price ในฟังก์ชัน ดังตัวอย่าง
4. จงสร้างฟังก์ชันที่สามารถคำนวณเงินถอนได้ แสดงผลโค้ดให้เหมือนด้านล่าง
   a. ถามราคาสินค้าของผู้ซื้อ (How much cost?)
   b. ถามจำนวนแบงค์ 100 (How many 100B?)
   c. ถามจำนวนแบงค์ 20 (How many 20B?)
   d. ถามจำนวนเหรียญ 10 ที่รับ (How many 10B?)
   e. บอกจำนวนเงินทอนไป

```
How much your cost:266
How many your 100B:2
How many your 20B:3
How many your 10B:1
Your change is 4.0 Bath
```