Function

Function คืออะไร

- ฟังก์ชันคือกลุ่มของคำสั่งที่มีการนิยามชื่อที่ใช้สำหรับอ้างอิงแทนกลุ่มคำสั่งนั้นเพื่อให้สามารถเรียกใช้งานได้ โดยกลุ่มของคำสั่งที่ประกอบกันเป็นฟังก์ชันจะทำงานอย่างใดอย่างหนึ่งโดยเฉพาะ

การสร้างฟังก์ชัน

- มีรูปแบบดังนี้

  def ชื่อฟังก์ชัน**():**

  กลุ่มของคำสั่ง

  - ชื่อของฟังก์ชันเป็นไปตามหลักการตั้งชื่อของ Python โดยต้องมีเครื่องหมายวงเล็บเปิดและวงเล็บปิดต่อท้าย
  - กลุ่มของคำสั่งหรือบล็อก เริ่มต้นด้วยการใช้เครื่องหมาย : ร่วมกับการเยื้องของบล็อก และสิ้นสุดบล็อกเมื่อมีการยุติการใช้ระยะเยื้องนั้น ทั้งนี้ระยะที่เยื้องเข้ามาอาจใช้การเคาะ spacebar อย่างน้อย 1 ครั้ง
  - โดยบล็อกเดียวกันระยะเยื้องต้องเท่ากัน

ตัวอย่างการสร้างฟังก์ชัน

```
def print_hello():
    print('--Hello from python')
    print('--Nice to meet you')
```

บรรทัดแรกเป็นจุดเริ่มต้นการสร้างฟังก์ชัน (def) ฟังก์ชันชื่อ print_hello สองบรรทัดถัดมาเป็นบล็อกของฟังก์ชันซึ่งประกอบไปด้วยฟังก์ชัน print() สองครั้งด้วยข้อความที่ต่างกัน

การเรียกใช้งานฟังก์ชัน

- คำสั่งในฟังก์ชันจะไม่ทำงานทันทีเมื่อเรารันโปรแกรม แต่คำสั่งในฟังก์ชันจะทำงานก็ต่อเมื่อมีการเรียกใช้ฟังก์ชันนั้นจากส่วนอื่นของโปรแกรม
- การเรียกใช้งานฟังก์ชันทำได้โดยการอ้างชื่อฟังก์ชัน
- เมื่อฟังก์ชันถูกเรียกใช้มีผลให้บล็อกของฟังก์ชันทำงาน 1 ครั้ง

Parameter ของฟังก์ชัน

- ฟังก์ชันสามารถรับข้อมูลเข้าไปเพื่อประมวลผลเพื่อให้ได้ผลลัพธ์ที่ต้องการได้
- ในการรันโปรแกรมแต่ละครั้งเมื่อข้อมูลเข้าเปลี่ยนค่าไปจะทำให้ผลลัพธ์มีค่าเปลี่ยนแปลงไปในลักษณะที่สอดคล้องกัน
- ข้อมูลเข้าที่ส่งให้กับฟังก์ชันเรียกว่า พารามิเตอร์ (parameter) มีรูปแบบดังนี้

def ชื่อฟังก์ชัน**(พารามิเตอร์1, พารามิเตอร์2, ...):**

    กลุ่มของคำสั่ง

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

```
def print_hello(name):
    print('--Hello ', name)
    print('--Nice to meet 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)
```

ลองเขียนฟังก์ชัน หาพื้นที่สามเหลี่ยม

การคืนค่าจากฟังก์ชัน

- ในการเรียกใช้งานฟังก์ชันบางครั้งเราต้องการให้ฟังก์ชันคืนค่าที่คำนวณได้มาให้ เพื่อเอาค่าที่คำนวณได้ไปใช้งานภายหลัง สามารถทำได้โดยใช้รูปแบบดังนี้

return นิพจน์

```
def bmi_cal(weight, height):
    bmi = weight / height ** 2
    return bmi
```

การออกแบบ function

ไม่รับ parameter
ไม่คืนค่า

ไม่รับ parameter
คืนค่า

รับ parameter
ไม่คืนค่า

รับ parameter
คืนค่า

ขอบเขตของตัวแปร

- ขอบเขตของตัวแปร(scope of variable) หมายถึงขอบเขตในการอ้างอิงตัวแปรที่นิยามขึ้นในโปรแกรม
- ขอบเขตของตัวแปรมี 2 ลักษณะได้แก่ตัวแปรเฉพาะที่(local variable) และตัวแปรส่วนกลาง(global variable)
- ตัวแปรทุกตัวมีขอบเขตอยู่ในบล็อก เริ่มต้นจากตำแหน่งที่แปรนั้นถูกนิยามขึ้น

- ตัวแปรเฉพาะที่เป็นตัวแปรที่นิยามอยู่ในบล็อกของฟังก์ชันนับจากตำแหน่งที่เริ่มต้นนิยามและจะถูกอ้างอิงได้ภายในบล็อกของฟังก์ชันดังกล่าวเท่านั้น ไม่สามารถอ้างอิงในบล็อกอื่น

- ตัวแปรส่วนกลางเป็นตัวแปรที่นิยามอยู่ในบล็อกของโปรแกรมหลัก และสามารถอ่างอิงในบล็อกของทุกฟังก์ชัน

```
y = 5

print(y)

print(x)

x = 10
```

```
def func():

    y = 5

    print(x,y)


x = 8

func()
```

โมดูลและเนมสเปซ

- โมดูล(module)หมายถึงไฟล์โปรแกรมไพทอนที่รวบรวมฟังก์ชัน คลาส ตัวแปรที่มีความสัมพันธ์กันให้เป็นหมวดหมู่เพื่อให้สะดวกต่อการอ้างอิงด้วยโปรแกรมอื่น
- โมดูลอาจประกอบด้วยโปรแกรมหลักที่สามารถประมวลผลด้วยตัวเองได้ ดังนั้นโปรแกรมที่เขียนขึ้น (*.py) ทุกโปรแกรมถือเป็นโมดูล โดยมีชื่อไฟล์โปรแกรมที่ไม่รวมส่วนขยาย .py เป็นชื่อโมดูล

- โปรแกรมสามารถอ้างถึงโมดูลโดยใช้คำสั่ง import พร้อมระบุชื่อโมดูลที่ต้องการ
- import ชื่อโมดูล1, ชื่อโมดูล2, ….

การเรียกใช้ฟังก์ชันในโมดูลมีรูปแบบดังนี้

ชื่อโมดูล.ชื่อฟังก์ชัน()

สมมติให้มีไฟล์ชื่อ hello.py ที่มีคำสั่งด้านในดังนี้

```
def print_hello(name):

        print('Hello',name)
```

โปรแกรมอื่นที่มีการอ้างถึงโมดูลชื่อ hello เพื่อเรียก print_hello สามารถทำได้ดังนี้

```
import hello

hello.print_hello('Jakarin')
```

- นอกจากการเรียกใช้โมดูลที่เขียนขึ้นใหม่แล้ว เรายังสามารถเรียกใช้โมดูลในไลบรารีมาตรฐานได้ในทำนองเดียวกัน เพื่อช่วยให้เขียนโปรแกรมได้รวดเร็วขึ้น
- ตัวอย่างโมดูลที่มีอยู่แล้วเช่น โมดูล math ซึ่งได้นิยามฟังก์ชันทางคณิตศาสตร์ต่างๆ ให้สามารถเรียกใช้งานได้โดยไม่ต้องเขียนเอง

```
import math


print('pi=', math.pi)
print('e=', math.e)
x=3.14
print('floor of ',x, ' is ',math.floor(x))
print('ceiling of ',x, ' is ',math.ceil(x))
print('factorial of 5 is ',math.factorial(5))
content = dir(math)
print(content)
```

- เราสามารถอ้างถึงโมดูลโดยกำหนดเฉพาะฟังก์ชันที่ต้องการใช้งานได้ในรูปแบบดังนี้
- from ชื่อโมดูล import ชื่อฟังก์ชัน1, ชื่อฟังก์ชัน2, …

```
from math import pi, e


print('pi=', pi)
print('e=', e)
```