程式設計

1082數位教材

單元5:函式及物件導向

主講老師：徐培倫

單元5 函式及物件導向

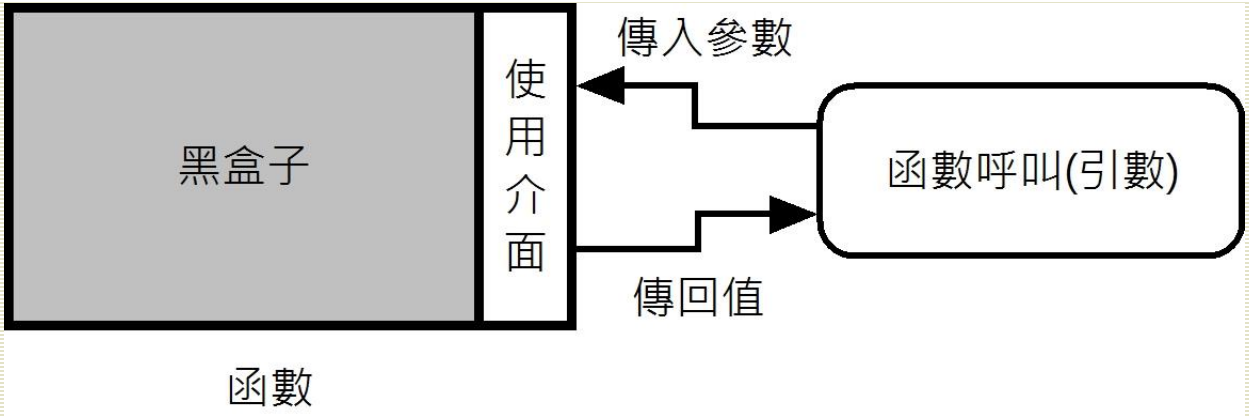
- 函式特性
- 函式定義
- 函式的引數
- 物件導向觀念
- 類別vs物件
- 應用範例

函式的特性

- 函式可以使程式的設計與維護更加容易
- 可提高程式的可讀性
- 函式可重複被呼叫，提高程式的再利用率
- 每一個函式有自己的任務，可按任務來規劃函式

函式是一個黑盒子

- 函式是一個獨立功能的程式區塊，如同是一個「黑盒子」（Black Box），我們根本不需要了解函式定義的程式碼內容，只要告訴我們如何使用此黑盒子的「介面」（Interface），就可以呼叫函式來使用函式的功能，如下圖所示：



Python 函式定義

- 不回傳值的函式
  - def 函式名稱(參數1,參數2, …):
  - 　　函式的敘述區塊
- 傳值的函式
  - def 函式名稱(參數1,參數2, …):
  - 　　函式的敘述區塊
  - 　　return 要傳回的變數或值

簡單例子（5-1def.py）

```
def hello():
    print('hello')

def circle(r):
    return r*r*3.14

hello()
print ("半徑=%d 圓面積=%.2f" % (10,circle(10)))
```

```
hello
半徑=10 圓面積=314.00
```

簡單例子（5-2sum.py）

```
 8 def sum(n):
 9     sss=0
10     for i in range(1,n+1):
11         sss+=i
12     return(sss)
13 print("1+2+...+%d=%d" % (5,sum(5)))
```

```
In [3]: runfile('D:/User
5-2factsum.py', wdir='D:
HW')
1+2+...+5=15
```

作業5-1

- 請利用 IDLE 及 Spyder (擇一)完成數位化學習平台之作業
- 將結果上傳 .py 及程式結果截圖

```
輸入 n, 利用sum及fact函數來計算 1+2+...+n 及n!

作業5-1 學號:xxx  姓名:xxx

n=5

1+2+...+5=15

5!=120
```

使用預設引數

- 依參數個數不同來區別的函式重載概念，在Python中可以使用預設引數（Argument）來解決
- def sum(a, b, c = 0):
- 　　return a + b + c
- print(sum(10, 20, 30))
- print(sum(10, 20))
- print(sum(c = 30, a = 10, b = 20))

```
60
30
60
```

不確定引數個數

- 定義函式的參數時使用*，表示該參數接受不定長度引數
- def sum(*nums):
- 　total = 0
- 　for number in nums:
- 　　total += number
- 　return total
- print(sum(1, 2))
- print(sum(1, 2, 3))
- print(sum(1, 2, 3, 4))

```
3
6
10
```

函式的引數使用tuple

- 使用tuple傳入，只要在傳入時加上*，則Tuple中每個元素會自動指定給各個參數
- def sum1(a, b, c):
- 　　return a + b + c
- numbers = (1, 2, 3)
- print(sum1(*numbers)) #答案=6

函式的引數使用dict

- 使用字典物件，只要在物件前加上**，則Python會依字典物件的鍵名稱，將值指定給對應名稱的參數。
- def sum2(a, b, c):
- 　　return a*2 + b + c
- args = {'b' : 4, 'a' : 5, 'c' : 6}
- print(sum2(**args)) #答案=20

Python-函式多回傳多值(tuple)

- 函式中回傳多個值以tupe方式來表示(5-3ret_tuple.py)
  - def mydata():
  - 　　name = "Alan"
  - 　　age = 35
  - 　　return name, age
  - print(mydata())
  - name1,age1=mydata()
  - print(name1,age1)

```
('Alan', 35)
Alan 35

In [5]:
```

lambda 運算式

- Python 提供匿名函式(anonymous function), 沒有函數名稱, 直接使用運算結果(不能使用區塊)為傳回值
- add = lambda x,y : x+y
- print (add(10,20))
- print (add(50,1.5))
- print (add("abc","de"))
- print (add(10,True)

```
30
51.5
abcde
11
```

變數的有效範圍

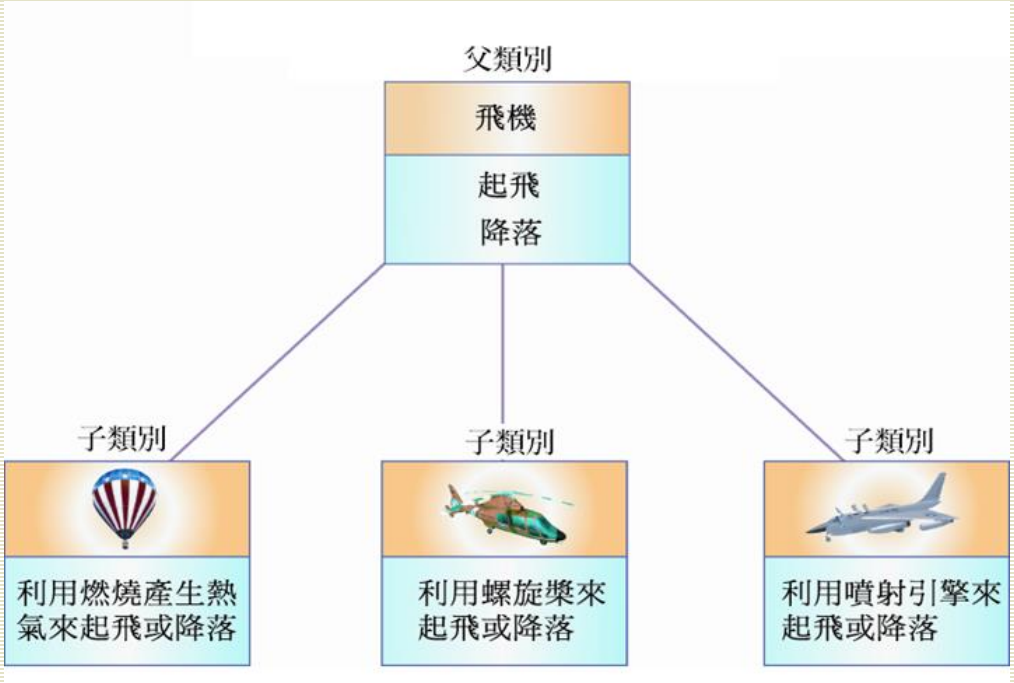
- 全域變數(global variable) : 所有敘述都可以存取
- 區域變數(local variable) : 區域內的敘述才可以存取
- def my():
- 　　x=1; print (x)
- print (x) #會發生錯誤,找不到 x 物件

物件導向

- 物件導向程式設計 (OOP，Object Oriented Programming) 主要有下列幾個特點：
  - **封裝** (encapsulation)：物件導向程式設計將資料與用來處理資料的函式放在一起成為一個類別，稱為「封裝」，著重於物件與物件之間的操作。

物件導向

- **繼承** (inheritance)：繼承指的是從既有的類別定義出新的類別，這個既有的類別叫做父類別 (parent class)，而這個新的類別則叫做子類別 (child class、subclass)。

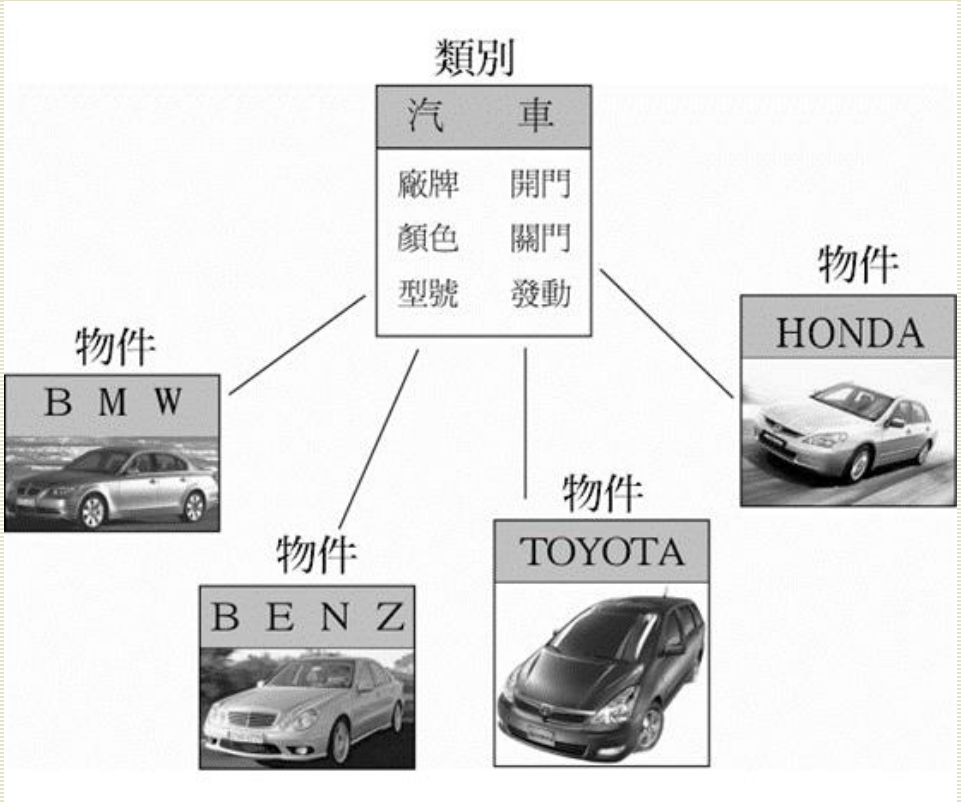


物件導向

- **多型** (polymorphism)：多型指的是當不同型態的物件收到相同的訊息時，會以各自的方法來做處理。
  - Python 是用鴨子型態的觀念來詮釋多型
  - 鴨子型態：某個人看見一隻鳥，
    - 只要這隻鳥走起路來像鴨子，
    - 游泳像鴨子，
    - 叫聲也像鴨子，
  - 那這個人就會叫那隻鳥為鴨子 (鳥=鴨子)

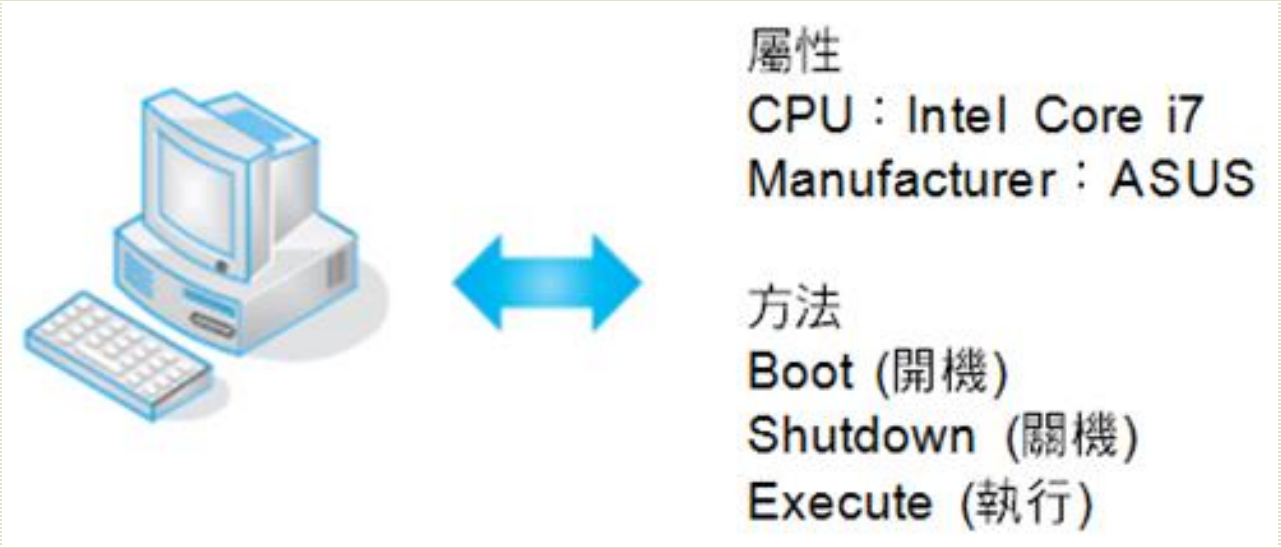
類別 vs 物件

- **類別** (class) 是物件的分類，就像物件的藍圖或樣板。
- **物件** (object) 或**實體** (instance) 是資料與程式碼的組合。



物件

- **屬性** (attribute) 或**成員變數** (member variable) 是用來描述物件的特質。
- **方法** (method) 或**成員函式** (member function) 是用來定義物件的動作。

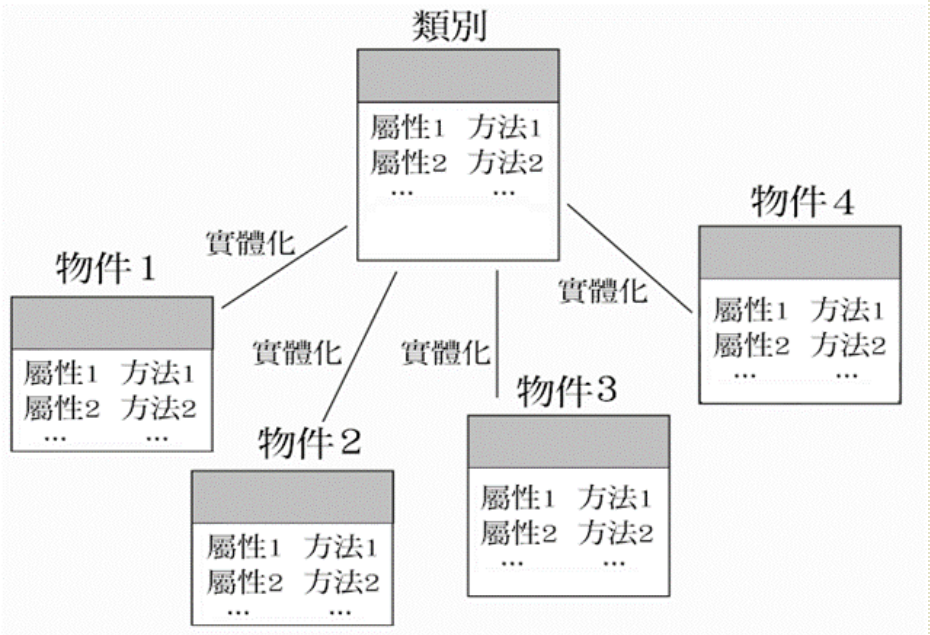


Python 物件

- Python中的所有資料都是物件 (object)，而物件的型別定義於類別 (class)
- 類別就像物件的藍圖或樣板，裡面定義了物件的資料，以及用來操作物件的函式，
- 前者稱為屬性 (attribute)，後者稱為方法 (method)。

Python 物件

- 物件是類別的實體(instance)，我們可以根據相同的類別建立多個物件，這個建立物件的動作稱為實體化(instantiation)。



Python 物件

- Python中的物件都有編號 (id)、型別 (type) 與值 (value)，我們可以透過下列函式取得這些資訊：
  - id(*x*)
  - type(*x*)
  - print(*x*)

```
12 x=123
13 print ("type=",type(x))
14 print ("id=",id(x))
15 print ("value=",x)
```

```
type= <class 'int'>
id= 1787657088
value= 123
```

定義類別

- 使用class關鍵字定義類別，其語法如下：

```
class 類別名稱:
    statement(s)
```

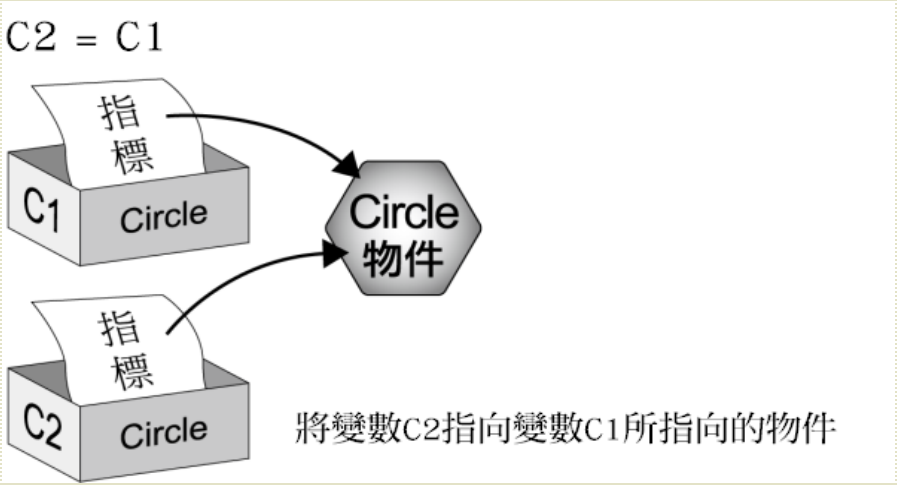
- 例子: 定義圓 Circle 類別

```
class Circle:
    PI = 3.14
    radius = 1

    def getArea(self):
        return self.PI * self.radius * self.radius
```

建立類別

- 根據類別建立物件，其語法如下：
- C1 = Circle()

- C2=C1



使用物件

- 在建立類別的物件後，就可以使用點運算子 (.) 存取物件的屬性與方法(5-4circle.py)

```
 8 class Circle:
 9     PI = 3.14
10     radius = 1
11     def getArea(self):
12         return self.PI * self.radius * self.radius
13     def my(self):
14         print (self)
15 C1=Circle()
16 C2=C1
17 C3=Circle()
18 print(C1.radius)
19 print(C2.PI)
20 print(C1.getArea())
21 print(C1.my())
22 print(C2.my())
23 print(C3.my())
```

```
1
3.14
3.14
<__main__.Circle object at 0x0000000008BCA438>
None
<__main__.Circle object at 0x0000000008BCA438>
None
<__main__.Circle object at 0x0000000008BDAB38>
None
```

__init__() 方法

- Python允許類別提供一個名稱為__init__()的特殊方法，在建立物件的時候，會自動呼叫這個方法將物件初始化(5-5circle1.py)

```
class Circle:
    PI = 3.14

    def __init__(self, r = 1):
        self.radius = r

    def getArea(self):
        return self.PI * self.radius * self.radius

C1 = Circle()
print("半徑=%2d 圓面積為=%.2f" %(C1.radius, C1.getArea()))

C2 = Circle(10)
print("半徑=%2d 圓面積為=%.2f" %(C2.radius, C2.getArea()))
```

```
半徑= 1 圓面積為=3.14
半徑=10 圓面積為=314.00
```

匿名物件

- Python允許我們在沒有將物件指派給變數的情況下存取物件，稱為匿名物件 (anonymous object) (5-5circle2.py)

```
class Circle:
    PI = 3.14

    def __init__(self, r = 1):
        self.radius = r

    def getArea(self):
        return self.PI * self.radius * self.radius

print("半徑=%2d 圓面積為=%.2f" %(Circle().radius, Circle().getArea()))

print("半徑=%2d 圓面積為=%.2f" %(Circle(10).radius, Circle(10).getArea()))
```

```
半徑= 1 圓面積為=3.14
半徑=10 圓面積為=314.00
```

私有成員 (私有屬性與方法)

私有屬性的名稱前面要加上兩個底線，但名稱後面則不能有底線，例如 __radius是私有屬性。(5-5circle3.py)

```
class Circle:
    PI = 3.14

    def __init__(self, r = 1):
        self.__radius = r

    def getRadius(self):
        return self.__radius

    def getArea(self):
        return self.PI * self.__radius * self.__radius

C1 = Circle(10)
print("C1的半徑=%d" % C1.getRadius())
print("C1的圓面積=%.2f" % C1.getArea())
```

```
C1的半徑=10
C1的圓面積=314.00
```

作業5-2

- 請利用 IDLE 及 Spyder (擇一)完成數位化學習平台之作業,將結果上傳 .py 及程式結果截圖

```
class Circle:
    PI = 3.14
    def __init__(self, r = 1):
        self.__radius = r
    def getRadius(self):
        return self.__radius
    def getArea(self):
        return self.PI * self.__radius * self.__radius
C1 = Circle(10)
print("C1的半徑=%d" % C1.getRadius())
print("C1的圓面積=%.2f" % C1.getArea())
```

請參考上述程式,完成下列畫面

作業5-2 執行畫面

請參考上述程式,完成下列畫面

```
aa = int(input("輸入直徑="))
my=Circle(aa)
半徑 => my.getRadius()
面積 => my.getArea()
周長 => my.getPeri()   請新增一個方法(函數)
```

作業5-2 學號:xxx 姓名:xxxx

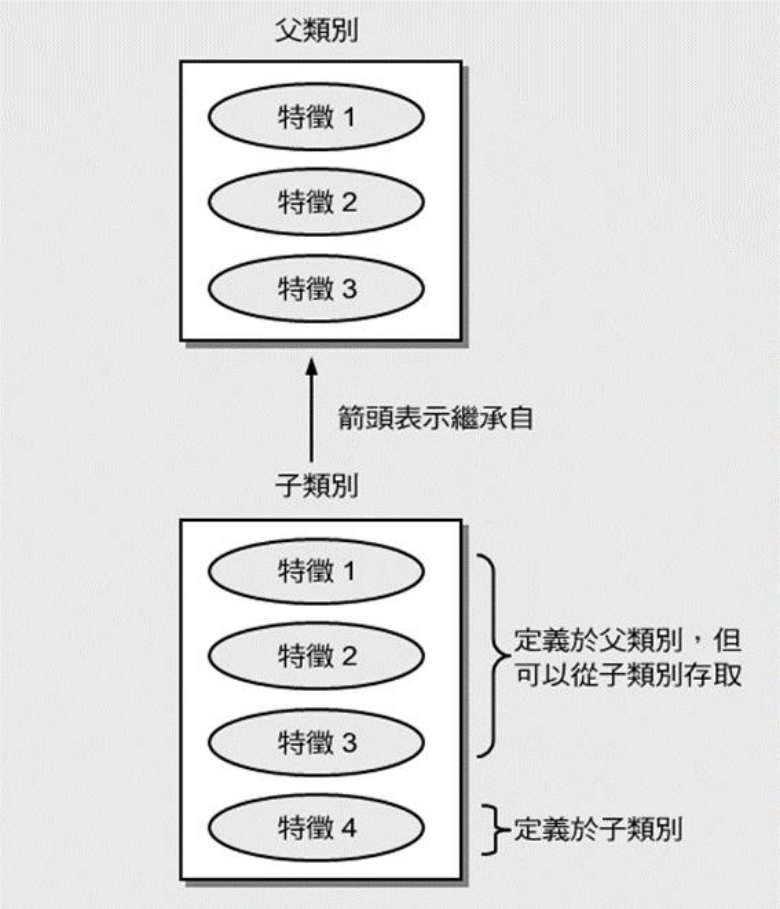
輸入直徑=10

周長=xx.x 公分

面積=xx.xx 平方公分

繼承

- 繼承 (inheritance) 是從既有的類別定義出新的類別，這個既有的類別叫做父類別 (parent class)，而這個新的類別則叫做子類別 (child class、subclass)。

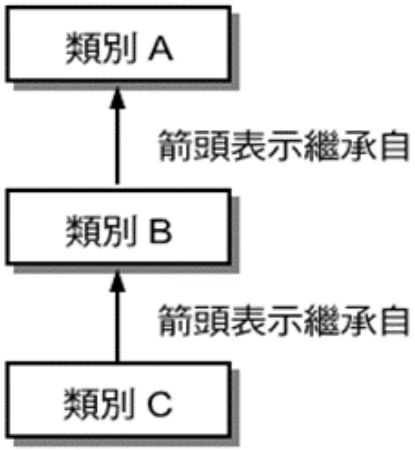


繼承-定義子類別

5-8inherit1.py

class 子類別**(** 父類別-1,父類別-2, ...**):**
statement(s)

```
class A:
    x = 1

class B(A):
    y = 'A'

class C(B):
    z = 3.14

my = C()
print("x屬性=", my.x)
print("y屬性=", my.y)
print("z屬性=", my.z)
```

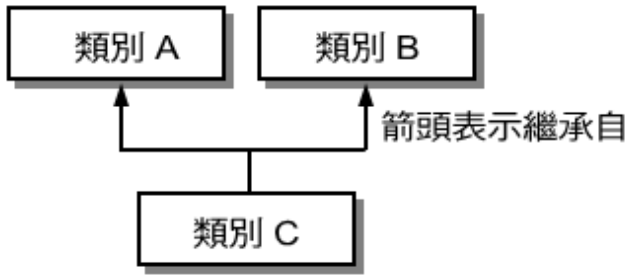



```
x屬性= 1
y屬性= A
z屬性= 3.14
```

多重繼承-定義子類別

5-9inherit2.py

class 子類別**(** 父類別-1,父類別-2, ...**):**
statement(s)



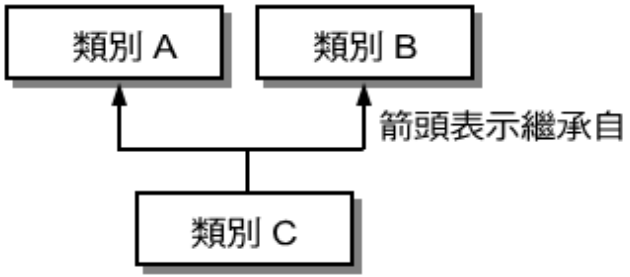

```
class A:
    x = 1

class B():
    y = 'A'

class C(A,B):
    z = 3.14

my = C()
print("x屬性=", my.x)
print("y屬性=", my.y)
print("z屬性=", my.z)
```

```
x屬性= 1
y屬性= A
z屬性= 3.14
```

覆蓋繼承的 getx( )方法

呼叫父類別內被覆蓋的方法 super() 5-10inherit3.py

```
class A:
    x = 1
    def getx(self):
        return self.x
class B:
    y = 'A'
    def gety(self):
        return self.y
class C(A,B):
    x = 3.14
    def getx(self):
        return self.x;
    def getxx(self):
        return super().x;
my = C()
print("x屬性=", my.getx()) ; print("y屬性=", my.gety())
print("A的x屬性=", my.getxx())
print("A的x屬性=", A().getx())
x = 3.14159
def getx():
    return x;
print (getx())
```




```
x屬性= 3.14
y屬性= A
A的x屬性= 1
3.14159
```

isinstance() 與issubclass()

- isinstance(obj, classA) : 物件obj 是否為 classA 或 ClassA子類別的物件 (True/False)
- issubclass(class, classA) : 類別class 是否為 ClassA 的子類別 (True/False)

```
>>> class A:
        x = 1
>>> class B(A):
        y = 2
>>> obj1 = A()
>>> obj2 = B()
>>> isinstance(obj1, A)
True
>>> isinstance(obj2, A)
True
>>> issubclass(B, A)
True
```

多型(polymorphism)

- 從(from)亂數模組(random)中，匯入(import)一個定義- 選擇(choice)方法 (5-11ploy.py)

```
from random import choice
class People:
    def say(self):
        print("大家好！")
class Student:
    def say(self):
        print("老師好！")
p1=People()
stu1=Student()
#通過choice方法我們可以隨機選擇列表中的某一項
obj=choice([p1,stu1])
print(type(obj))
obj.say()
```

```
In [26]: runfile('D:/Users/
PyNo5')
<class '__main__.People'>
大家好！

In [27]: runfile('D:/Users/
PyNo5')
<class '__main__.Student'>
老師好！

In [28]: runfile('D:/Users/
PyNo5')
<class '__main__.Student'>
老師好！

In [29]: runfile('D:/Users/
PyNo5')
<class '__main__.People'>
大家好！
```

應用範例(5-12classBMI.py)

- classBMI.py 建立類別 class BMI 提供計算 BMI 的方法
- BMI.py 計算個人 BMI

```
 8 #classBMI.py
 9 class BMI:
10     def __init__(self,h=0,w=0):
11         self.height=h
12         self.weight=w
13     def BMI(self):
14         return self.weight / ((self.height/100)**2)
```

應用範例 (5-13BMI.py)

- BMI.py 計算個人 BMI
- import classBMI #載入 classBMI.py 模組

```
 8 #BMI.py
 9 import classBMI
10 a = classBMI.BMI(170,70)
11 print("BMI=%.3f" % a.BMI())
```

```
In [6]: runfile(
BMI=24.221
```

應用範例

- 增加 BMIdata 方法

```
 8 #classBMI.py
 9 class BMI:
10     def __init__(self,h=0,w=0):
11         self.height=h
12         self.weight=w
13     def BMI(self):
14         return self.weight / ((self.height/100)**2)
15     def BMIdata(self,h=0,w=0):
16         if h==0:
17             return self.weight / ((self.height/100)**2)
18         else:
19             return w/(h/.100)**2
```

應用範例

- BMI.py 計算個人 BMIdata

```
 8 #BMI.py
 9 import classBMI
10 a = classBMI.BMI(170,70)
11 print("BMI=%.3f" % a.BMI())
12 print("BMI=%.3f" % a.BMIdata())
13 print("BMI=%.3f" % a.BMIdata(170,80))
```

```
In [6]: runfile
BMI=24.221
BMI=24.221
BMI=27.682
```

物件導向 – 繼承(BMI1.py)

- 女性(Woman)類別包含著人類(Human)類別的所有特性，女性(Woman)裡面具有三個屬性— 胸圍(bust)、腰圍(waist)、臀圍(hip)，兩個方法—初始化(__init__)與印出三圍(printBWH)

物件導向 – 繼承(5-14BMI1.py)

```
 8 class Human:
 9     def __init__(self,h=0,w=0):
10         self.height=h
11         self.weight=w
12     def BMI(self):
13         return self.weight / ((self.height/100)**2)
14
15 class Woman(Human):
16     def __init__(self,h,w,bust=0,waist=0,hip=0):
17         super().__init__(h,w)
18         self.bust=bust
19         self.waist=waist
20         self.hip=hip
21     def printBWH(self):
22         print("胸圍=%d,腰圍=%d,臀圍=%d" % (self.bust,self.waist,self.hip))
23
24 a = Woman(165,54,83,64,84)
25 print("BMI=%.2f" % a.BMI())
26 a.printBWH()
```

```
BMI=19.83
胸圍=83,腰圍=64,臀圍=84
```

私有成員範例 (5-15BMI2.py)

- 加入 __height , __BMI 是私有成員方式

```
 8 class Human:
 9     def __init__(self,h=0,w=0):
10         self.__height=h
11         self.__weight=w
12     def __BMI(self):
13         return self.__weight / ((self.__height/100)**2)
14     def getBMI(self):
15         return self.__BMI()
16     def getHeight(self):
17         return self.__height
18     def getWeight(self):
19         return self.__weight
20     def setWeight(self,w):
21         self.__weight=w
22
23 a = Human(180,80)
24 print("身高=%d 體重=%d " %(a.getHeight(),a.getWeight()))
25 print("BMI=%.2f" %a.getBMI())
26 a.setWeight(60)
27 print("身高=%d 體重=%d " %(a.getHeight(),a.getWeight()))
28 print("BMI=%.2f" %a.getBMI())
```

```
In [13]: runfile
身高=180 體重=80
BMI=24.69
身高=180 體重=60
BMI=18.52
```

作業5-3

- 請利用 IDLE 及 Spyder (擇一)完成數位化學習平台之作業
- 將結果上傳 .py 及程式結果截圖
- 請參考 BMI 測試

  http://health99.hpa.gov.tw/OnlinkHealth/Onlink_BMI.aspx

作業5-3

h=int(input("身高="))
w= int(input("體重="))
輸入身高及體重
可以得到 你的BMI 及 建議

作業5-8 學號:xxxx 姓名:xxxx
身高=xxx
體重=xxx
BMI=xx.xx 建議:健康體位

作業5-3 學號:xxxx 姓名:xxxx
身高=xxx
體重=xxx
BMI=xx.xx 建議:過重

參考如下:

```
if a.BMI()<18.5:
    print("體重過輕")
elif 18.5<=a.BMI()<24:
    print("健康體位")
elif 24<=a.BMI()<27:
    print("過重")
elif 27<=a.BMI()<30:
    print("輕度肥胖")
elif 30<=a.BMI()<35:
    print("中度肥胖")
else:
    print("重度肥胖")
```