Python 程式設計作業

範圍： 類別、模組的應用

銘傳大學電腦與通訊工程系

班　　級	電通四乙
姓　　名	陳昱叡
學　　號	04052474
作業成果	應繳作業共 7 題，前 3 題每題 20 分，後 4 題每題 10 分，滿分為 100 分 我共完成 7 題，應得 100 分
授課教師	陳慶逸

■ 請確實填寫自己寫完成題數，並且計算得分。填寫不實者(如上傳與作業明顯無關的答案，或是計算題數有誤者)，本次作業先扣 50 分。

■ 確實填妥封面的內容，完成後請上傳 pdf 檔。

EX 1: 試寫出一個命名為 strSplitClass 的 Python 類別，該類別裡有一個 StrSplit 方法(methods)可將輸入的字串進行切割，得到以字(word)組成的串列。

例如:

```
str1 = strSplitClass()
str1.strSplit()
```

```
machine learning .ipynb
```

```
Out[42]: ['machine', 'learning', '.ipynb']
```

圖 5-16　輸入與輸出結果

```
class strSplitClass:
    def __init__(self):
        self.str = ""
        print(self.str)
    def strSplit(self):
        self.str=input()
        self.str=self.str.split(" ")
        print(self.str)
str1=strSplitClass()
str1.strSplit()
```

```
In [8]: class strSplitClass:
            def __init__(self):
                self.str = ""
                print(self.str)
            def strSplit(self):
                self.str=input()
                self.str=self.str.split(" ")
                print(self.str)
        str1=strSplitClass()
        str1.strSplit()
```

```
machine learning .ipynb
['machine', 'learning', '.ipynb']
```

EX 2: 試寫出一個命名為 cartesian 的 Python 類別，該類別的輸入參數為 x 和 y 兩個數值(為平面座標中的一個點(x,y))，而類別裡的方法(distanceToOrigin)可以計算出(x,y)這個點離原點的距離。

例如:

```
a = cartesian(3,3)
a.distanceToOrigin()
```

```
4.242640687119285
```

圖 5-167　輸入與輸出結果

```
class cartesian:
    def __init__(self, x, y):
        self.x=x
        self.y=y
    def distanceToOrigin(self):
        print(((self.x-0)**2+(self.y-0)**2)**0.5)
a=cartesian(3,3)
a.distanceToOrigin()
```

```
In [10]: class cartesian:
             def __init__(self, x, y):
                 self.x=x
                 self.y=y
             def distanceToOrigin(self):
                 print(((self.x-0)**2+(self.y-0)**2)**0.5)
         a=cartesian(3,3)
         a.distanceToOrigin()

         4.242640687119285
```

EX 4: 試寫出一個命名為 Shape 的 Python 類別，該類別由計算矩形面積(area)、矩形周長(perimeter)和尺度大小改變(scaleSize)等三個方法(methods)所構成。

例如:

```
rectangle = Shape(4, 8)

print(rectangle.area())  # finding the area
print(rectangle.perimeter())  #finding the perimeter

rectangle.scaleSize(0.5)  #making the rectangle 50% smaller
print(rectangle.area())  #re-printing the new area of the rectangle
print(rectangle.perimeter())  #re-printing the new perimeter of the rectangle
```

```
32
24
8.0
12.0
```

圖 5-19　輸入與輸出結果

```
class Shape:
    def __init__(self,width,height):
        self.width=width
        self.height=height
    def area(self):
        return self.width*self.height
    def perimeter(self):
        return self.width*2+self.height*2
    def scalesize(self,size):
        self.width=self.width*size
        self.height=self.height*size
rectangle=Shape(4,8)

print(rectangle.area())
print(rectangle.perimeter())

rectangle.scalesize(0.5)
print(rectangle.area())
print(rectangle.perimeter())
```

In [6]:
```
class Shape:
    def __init__(self,width,height):
        self.width=width
        self.height=height
    def area(self):
        return self.width*self.height
    def perimeter(self):
        return self.width*2+self.height*2
    def scalesize(self,size):
        self.width=self.width*size
        self.height=self.height*size
rectangle=Shape(4,8)

print(rectangle.area())
print(rectangle.perimeter())

rectangle.scalesize(0.5)
print(rectangle.area())
print(rectangle.perimeter())
```

```
32
24
8.0
12.0
```

EX 5: 試寫出一個命名為 IOString 的 Python 類別，該類別由兩個兩個方法(methods)所構成，其中 getString 負責接收使用者所輸入的字串，而另一個方法 printString 則是將字串改成大寫後列印出來。

例如:

```
str1 = IOString()
str1.getString()
str1.printString()
```

```
block
BLOCK
```

圖 5-20　輸入與輸出結果

```
class IOString:
    def __init__(self):
        self.string=""
    def getString(self):
        self.string=input()
    def printString(self):
        self.string=self.string.upper()
        print(self.string)
str1=IOString()
str1.getString()
str1.printString()
```

In [4]:
```
class IOString:
    def __init__(self):
        self.string=""
    def getString(self):
        self.string=input()
    def printString(self):
        self.string=self.string.upper()
        print(self.string)
str1=IOString()
str1.getString()
str1.printString()
```

```
block
BLOCK
```

EX 6: 試寫出一個命名為 Employee 的 Python 類別，該類別由兩個兩個方法(methods)所構成，其中 displayEmployee 可以列印受僱員的名字和薪水，displayCount 則是輸出受僱者的總數。

例如:

```
emp1 = Employee("Zara", 2000)
emp2 = Employee("Manni", 5000)
emp3 = Employee("Amy", 5000)
emp1.displayEmployee()
emp2.displayEmployee()
emp3.displayEmployee()
print("Total Employee %d" % Employee.empCount)
```

```
Name :  Zara , Salary:  2000
Name :  Manni , Salary:  5000
Name :  Amy , Salary:  5000
Total Employee 3
```

圖 5-21　輸入與輸出結果

```
class Employee:
    empCount=0
    def __init__(self,Name,Salary):
        self.Name=Name
        self.Salary=Salary
        Employee.empCount=Employee.empCount+1
    def displayEmployee(self):
        print("Name : ",self.Name,",","Salary : ",self.Salary)
    def displayCount(self):
        print("Total Employee %d" % Employee.empCount)
emp1=Employee("Zera",2000)
emp2=Employee("Manni",5000)
emp3=Employee("Amy",5000)
emp1.displayEmployee()
emp2.displayEmployee()
emp3.displayEmployee()
emp3.displayCount()
```

In [21]:
```
class Employee:
    empCount=0
    def __init__(self,Name,Salary):
        self.Name=Name
        self.Salary=Salary
        Employee.empCount=Employee.empCount+1
    def displayEmployee(self):
        print("Name : ",self.Name,",","Salary : ",self.Salary)
    def displayCount(self):
        print("Total Employee %d" % Employee.empCount)
emp1=Employee("Zera",2000)
emp2=Employee("Manni",5000)
emp3=Employee("Amy",5000)
emp1.displayEmployee()
emp2.displayEmployee()
emp3.displayEmployee()
emp3.displayCount()
```

```
Name :  Zera , Salary :  2000
Name :  Manni , Salary :  5000
Name :  Amy , Salary :  5000
Total Employee 3
```

EX 7: 對於一個運動學公式: $S = V_0t + 1/2\ gt^2$，其中 V_0 代表初速，t 為時間，g = 9.8。試寫出一個命名為 kinematic 的 Python 類別，該類別的輸入參數為初速 V_0，而類別裡的方法(disFormula)在給予時間 t 這個參數後，可以計算出移動距離 S。

例如:

```
computDis = kinematic(4)  # V0 = 4 m/s
computDis.disFormula(2)   # t = 2s
```

```
27.62
```

圖 5-22　輸入與輸出結果

```
class kinematic:
    def __init__(self,v):
        self.v=v
    def disFormula(self,t):
        S=self.v*t+9.8*t*t*0.5
        print(S)
computDis=kinematic(4)
computDis.disFormula(2)
```

```
In [2]: class kinematic:
            def __init__(self,v):
                self.v=v
            def disFormula(self,t):
                S=self.v*t+9.8*t*t*0.5
                print(S)
        computDis=kinematic(4)
        computDis.disFormula(2)

        27.6
```

EX 8: 試建立一個命名為 calculation 的 Python 模組，該模組提供加、減、乘、除等功能的運算。

例如:

```
import calculation

print(calculation.add(4,8))
print(calculation.sub(4,8))
print(calculation.mul(4,8))
print(calculation.div(4,8))
```

```
12
-4
32
0.5
```

圖 5-23　輸入與輸出結果

Calculation.py

```
def add(x,y):
    return x+y
def sub(x,y):
    return x-y
def mul(x,y):
    return x*y
def div(x,y):
    return x/y
```

ex8.ipynb

```
import calculation

print(calculation.add(4,8))
print(calculation.sub(4,8))
print(calculation.mul(4,8))
print(calculation.div(4,8))
```

In [2]:
```
import calculation

print(calculation.add(4,8))
print(calculation.sub(4,8))
print(calculation.mul(4,8))
print(calculation.div(4,8))
```

```
12
-4
32
0.5
```