

Python 程式設計

範圍： Set、Dictionary 的應用

銘傳大學電腦與通訊工程系

班 級	電通四甲
姓 名	趙啟翔
學 號	05050130
成 績	應繳作業共 <u>8</u> 題，前六題每題 10 分，後兩題每題 20 分，滿分為 100 分 共完成 <u>8</u> 題，應得 <u>100</u> 分
授課教師	陳慶逸

※直接將你的程式碼貼在指定的欄位裡，並且執行題目要求的輸入參數

※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。

EX 1: 針對下面的字典 Adict，試寫幾行 python 程式碼，增加'Jake'這個名字，且其電話為 928544322；並且刪除 Jake 這筆資料。

```
Adict = {'John': 938457566, 'Peter': 928377464, 'Mary': 941662781,
        "Jill" : 928662781}

# write your code here

Adict["Jake"] = 928544322

del Adict['Jill']

# testing code
if "Jake" in Adict:
    print("Jake is listed in the Adict.")
if "Jill" not in Adict:
    print("Jill is not listed in the Adict.")
```

```
In [6]: Adict = {'John': 938457566, 'Peter': 928377464, 'Mary': 941662781, "Jill" : 928662781}

# write your code here

Adict["Jake"] = 928544322

del Adict['Jill']

# testing code
if "Jake" in Adict:
    print("Jake is listed in the Adict.")
if "Jill" not in Adict:
    print("Jill is not listed in the Adict.")

Jake is listed in the Adict.
Jill is not listed in the Adict.
```

EX 2: 若有一個字典 dict_double 的內容如下，試利用程式產生之；並在該字典下面接著寫一個 isKeyPresent(x)函式，該函式的功能可檢查所輸入的鍵值 x 是否存在於 Adict 之中。

dict_double 的內容如下：

{0: 0, 1: 2, 2: 4, 3: 6, 4: 8, 5: 10, 6: 12, 7: 14, 8: 16, 9: 18, 10: 20, 11: 22, 12: 24, 13: 26, 14: 28, 15: 30, 16: 32, 17: 34, 18: 36, 19: 38}

例如：

<code>isKeyPresent(5)</code>	Key is present in the dictionary
<code>isKeyPresent(9)</code>	Key is present in the dictionary
<code>isKeyPresent(25)</code>	Key is not present in the dictionary

我的作答：

請在下面欄位貼上程式碼：

```
def isKeyPresent(i):  
    dict_double = {0: 0, 1: 2, 2: 4, 3: 6, 4: 8, 5: 10, 6: 12, 7: 14,  
8: 16, 9: 18,  
10: 20, 11: 22, 12: 24, 13: 26, 14: 28, 15: 30, 16: 32, 17: 34, 18:  
36, 19: 38}  
    if i in dict_double:  
        print("Key is present in the dictionary")  
    if i not in dict_double:  
        print("Key is not present in the dictionary")  
isKeyPresent(5)  
isKeyPresent(9)  
isKeyPresent(25)
```

執行結果擷圖：

```
In [5]: 1 def isKeyPresent(i):
2         dict_double = {0: 0, 1: 2, 2: 4, 3: 6, 4: 8, 5: 10, 6: 12, 7: 14, 8: 16,
3         10: 20, 11: 22, 12: 24, 13: 26, 14: 28, 15: 30, 16: 32, 17: 34, 18: 36, 19: 38,
4         20: 40, 21: 42, 22: 44, 23: 46, 24: 48, 25: 50, 26: 52, 27: 54, 28: 56, 29: 58,
5         30: 60, 31: 62, 32: 64, 33: 66, 34: 68, 35: 70, 36: 72, 37: 74, 38: 76, 39: 78,
6         40: 80, 41: 82, 42: 84, 43: 86, 44: 88, 45: 90, 46: 92, 47: 94, 48: 96, 49: 98,
7         50: 100, 51: 102, 52: 104, 53: 106, 54: 108, 55: 110, 56: 112, 57: 114, 58: 116,
8         59: 118, 60: 120, 61: 122, 62: 124, 63: 126, 64: 128, 65: 130, 66: 132, 67: 134,
9         68: 136, 69: 138, 70: 140, 71: 142, 72: 144, 73: 146, 74: 148, 75: 150, 76: 152,
10        77: 154, 78: 156, 79: 158, 80: 160, 81: 162, 82: 164, 83: 166, 84: 168, 85: 170,
11        86: 172, 87: 174, 88: 176, 89: 178, 90: 180, 91: 182, 92: 184, 93: 186, 94: 188,
12        95: 190, 96: 192, 97: 194, 98: 196, 99: 198, 100: 200}
13        if i in dict_double:
14            print("Key is present in the dictionary")
15        else:
16            print("Key is not present in the dictionary")
17        isKeyPresent(5)
18        isKeyPresent(9)
19        isKeyPresent(25)
```

Key is present in the dictionary
Key is present in the dictionary
Key is not present in the dictionary

EX 3: 針對下面的字典 Adict，試以下面型式輸出 Adict 的內容。

Member: John -> Tel: 938457566

Member: Peter -> Tel: 928377464

Member: Mary -> Tel: 941662781

Member: Jill -> Tel: 928662781

```
Adict = {'John': 938457566, 'Peter': 928377464, 'Mary': 941662781,
        "Jill" : 928662781}
for name, number in Adict.items():
    print("Member: %s -> Tel: %d" % (name, number))
```

執行結果擷圖：

```
In [6]: 1 Adict = {'John': 938457566, 'Peter': 928377464, 'Mary': 941662781, 'Jill': 928662781}
        2 for name, number in Adict.items():
        3     print("Member: %s -> Tel: %d" % (name, number))
        4
```

Member: John -> Tel: 938457566
Member: Peter -> Tel: 928377464
Member: Mary -> Tel: 941662781
Member: Jill -> Tel: 928662781

EX 4: 試寫一 python 函式 generateDict(n)，它會根據我們所給予的 n 值，回傳一個內容為{1:2, 2:8, 3:18, 4: 32, ..., n: 2*n*n}的字典。例如：

generateDict(3)	{1: 2, 2: 8, 3: 18}
generateDict(10)	{1: 2, 2: 8, 3: 18, 4: 32, 5: 50, 6: 72, 7: 98, 8: 128, 9: 162, 10: 200}

```
def generateDict(n):
    d = {i: 2*i**2 for i in range(1,n+1)}
    return d

print(generateDict(3))
print(generateDict(10))
```

執行結果擷圖：

```
In [8]: 1 def generateDict(n):
        2     d = {i: 2*i**2 for i in range(1,n+1)}
        3     return d
        4
        5 print(generateDict(3))
        6 print(generateDict(10))
        7
```

{1: 2, 2: 8, 3: 18}
{1: 2, 2: 8, 3: 18, 4: 32, 5: 50, 6: 72, 7: 98, 8: 128, 9: 162, 10: 200}

EX 5: 若 `myDict = { 'data1':160,'data2':-254,'data3':1247, 'data4':2247}`，試寫一 python 程式來得到字典裡所有值(value)連乘的結果。以本例而言，其輸出值應為 -1333800。

```
myDict = { 'data1':160,'data2':-254,'data3':1247, 'data4':2247}
d=1
for name, number in myDict.items():
    d=d*number
print(d)
```

執行結果擷圖：

```
In [14]: 1 myDict = { 'data1':160,'data2':-254,'data3':1247, 'data4':2247}
          2 d=1
          3 for name, number in myDict.items():
          4     d=d*number
          5 print(d)

-113873645760
```

EX 6: `zip()` 是 Python 的一個內建函數，它接受一系列可迭代的對象作為參數，將對象中對應的元素打包成一個個 `tuple`。例如：

```
a = [1, 2, 3]
b = ['bird', 'dog', 'cat']
c = dict(zip(a,b)) # c = {1: 'apple', 2: 'dog', 3: 'cat'}
```

現有字串 `str = 'hogen'`，試將之轉成一個由該字串裡的字母所構成的串列 A；再建立一個由這些字母對應的 ASCII 碼所構成的串列 B（使用 `ord()` 函式可將英文字母轉換為 ASCII 碼），最後列印出由 A 和 B 建立的字典 C。C 的內容如下：

C = {'h': 104, 'o': 111, 'g': 103, 'e': 101, 'n': 110}

```
str = 'hogen'
A = list(str)
B = []
for i in A:
    B.append(ord(i))
C = dict(zip(A,B))
print(C)
```

執行結果擷圖：

```
In [10]: 1 str = 'hogen'
          2 A = list(str)
          3 B = []
          4 for i in A:
          5     B.append(ord(i))
          6 C = dict(zip(A,B))
          7 print(C)
          8
```

```
{'h': 104, 'o': 111, 'g': 103, 'e': 101, 'n': 110}
```

EX 7: 現在一字典 `num = {'n1': [2, 3, 7, 9, 1], 'n2': [5, 1, 2, 8, 13], 'n3': [3, 2, 4, 6, 9]}`，試寫一 python 程式來將字典裡所有由串列所構成的值(value)都進行排序。例如本例最後應輸出：

```
{'n1': [1, 2, 3, 7, 9], 'n2': [1, 2, 5, 8, 13], 'n3': [2, 3, 4, 6, 9]}
```

以 for-loop 寫法：

```
num = {'n1': [2, 3, 7, 9, 1], 'n2': [5, 1, 2, 8, 13], 'n3': [3, 2, 4, 6, 9]}
for i in num:
    num[i] = sorted(num[i])
print(num)
```

```
In [11]: 1 num = {'n1': [2, 3, 7, 9, 1], 'n2': [5, 1, 2, 8, 13], 'n3': [3, 2, 4, 6, 9]}
          2 for i in num:
          3     num[i] = sorted(num[i])
          4 print(num)
          5
          {'n1': [1, 2, 3, 7, 9], 'n2': [1, 2, 5, 8, 13], 'n3': [2, 3, 4, 6, 9]}
```

以解析式 for-loop 寫法：

```
num = {'n1': [2, 3, 7, 9, 1], 'n2': [5, 1, 2, 8, 13], 'n3': [3, 2, 4, 6, 9]}
num = {i: sorted(num[i]) for i in num}
print(num)
```

執行結果擷圖：

```
In [12]: 1 num = {'n1': [2, 3, 7, 9, 1], 'n2': [5, 1, 2, 8, 13], 'n3': [3, 2, 4, 6, 9]}
          2 num = {i: sorted(num[i]) for i in num}
          3 print(num)
          4
          {'n1': [1, 2, 3, 7, 9], 'n2': [1, 2, 5, 8, 13], 'n3': [2, 3, 4, 6, 9]}
```

EX 8: 找出同時出現在兩個不同的字典中的鍵-值對，

例如：

```
x = {'key1': 1, 'key2': 3, 'key3': 2}
y = {'key1': 1, 'key2': 2}
```


Sample output: ('key1', 1)

```
x = {'key1': 1, 'key2': 3, 'key3': 2}
```

```
y = {'key1': 1, 'key2': 3}
```

Sample output:

```
('key2', 3)
```

```
('key1', 1)
```

```
x = {'key1': 1, 'key2': 3, 'key3': 2}
y = {'key1': 1, 'key2': 3}
z = x.items() & y.items()
for i in z:
    print(i)
```

執行結果擷圖：

```
In [13]: 1 x = {'key1': 1, 'key2': 3, 'key3': 2}
          2 y = {'key1': 1, 'key2': 3}
          3 z = x.items() & y.items()
          4 for i in z:
          5     print(i)
          6
```

```
('key1', 1)
```

```
('key2', 3)
```

