

101 學年度 Assignment 5

Description

- 本作業必須以「Microsoft Visual Studio 2010 Professional」完成，利用其它軟體完成者將不予計分。
- 開啟「Microsoft Visual Studio 2010」，新增一個「專案」，以你的學號及作業的題號作為專案名稱。例如你的學號為 s123456 且要寫的作業為 Assignment 1 的第 3 題，則你的專題名稱為「s123456_Assignment1_3」。
- 你的專案目錄可能被儲存在"C:\Documents and Settings\Administrator\My Documents\Visual Studio 2010\Projects\s123456_Assignment1_3" in XP 作業系統 or "C:\Users\Administrator\Documents\Visual Studio 2010\Projects\s123456_Assignment1_3" in Windows 7 作業系統。
- 在完成程式撰寫後，完成存檔並關閉 Microsoft Visual Studio 2010 Professional。重複上述動作，進行下一題的作業。
- 當完成所有作業，回到「Projects」目錄，選擇所有要上傳的目錄，例如「s123456_Assignment1_1」、「s123456_Assignment1_2」、「s123456_Assignment1_3」等，並將滑鼠壓在這些目錄上並按滑鼠右鍵，以「傳送到」選項下的壓縮功能進行壓縮，壓縮後將得到此一作業的壓縮檔，例如 s123456_Assignment1_1.zip。之後將此一壓縮檔的檔名改為 s123456.zip，並上傳該檔至虛擬教室。
- 若繳交的內容(含檔案命名方式，目錄名稱)與指定的內容不合，將不被評分。

1. 假設有一鐵路系統共有四個車站(台北、桃園、台中、高雄)，而各站間的票價如下表所示，例如台北到桃園的票價為 140 元、台北到台中的票價為 480 元、高雄到台中的票價為 500 元等。注意，去程與回程的票價可能不同。

| 車站 | 台北 | 桃園 | 台中 | 高雄 |
|----|-----|-----|-----|-----|
| 台北 | 0 | 140 | 480 | 950 |
| 桃園 | 130 | 0 | 550 | 880 |
| 台中 | 520 | 430 | 0 | 520 |
| 高雄 | 980 | 870 | 500 | 0 |

在你的程式中，上述的票價表將以 price 二維陣列方式儲存，而車站將以 station 一維陣列儲存，如下所示。

```
double[,] price = { { 0, 140, 480, 950 }, { 130, 0, 550, 880 }, { 520, 430, 0, 520 }, { 980, 870, 500, 0 } };  
string[] station = { "台北", "桃園", "台中", "高雄" };
```

試發展一個 C# 程式，可查詢「查詢最便宜票價」，此一程式將至 price 二維陣列中查詢，得知 130 元為最便宜票價，並將其起始站與終點站利用 station 一維陣列的資料印出，如下圖所示。

```
最便宜票價為：130
起始站為：桃園
終點站為：台北
請按任意鍵繼續 . . .
```

Note: 在「查詢最便宜票價」時，要將相同的起始站與終點站情況避開(即票價為 0 的情況)，例如起始站為台北且終點站為台北不能列入最便宜票價。

Note: 你的程式必須是一般性的寫法，並非只針對目前提供的資料而已。也就是，若任意修改你程式中的 price 二維陣列的內容及長度，則你的程式必須仍能正確執行。

2. 呈上題，利用相同的資料(price 及 station 陣列)，以 C#發展一個票價調整的程式。若原始票價小於 300 元，則調幅為 40%；若原始票價大於或等於 300 元且小於 600 元，則調幅為 20%；若原始票價大於或等於 600 元，則不予調整。調整完票價後，將所有可能起站、終站、及調整後的票價依序列出，如下圖所示。例如桃園至台北的調整後票價為 $130 \times 1.4 = 182$ 、台中至高雄的調整後票價為 $520 \times 1.2 = 624$ 、而高雄至台北的調整後票價為 $980 \times 1 = 980$ 。

```
起站 終站 票價
台北 桃園 196
台北 台中 576
台北 高雄 950
桃園 台北 182
桃園 台中 660
桃園 高雄 880
台中 台北 624
台中 桃園 516
台中 高雄 624
高雄 台北 980
高雄 桃園 870
高雄 台中 600
請按任意鍵繼續 . . .
```

Note: 在「票價調整」時，相同的起始站與終點站情況也要避開(即票價為 0 的情況)。

3. 利用第 1 題的資料，試發展一個 C#程式，先詢問使用者「起點站的站名」，如下：

請輸入起站站名：

及「終點站的站名」，如下：

請輸入終站站名：

之後顯示其票價如下：

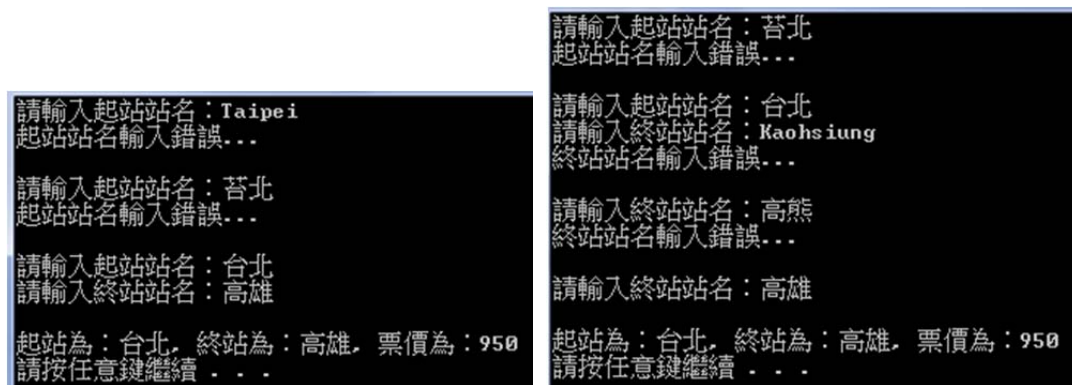
起站為：，終站為：，票價為：

以下為兩個範例的畫面：

```
請輸入起站站名：桃園
請輸入終站站名：高雄
起站為：桃園，終站為：高雄，票價為：880
請按任意鍵繼續 . . .
```

```
請輸入起站站名：高雄
請輸入終站站名：台北
起站為：高雄，終站為：台北，票價為：980
請按任意鍵繼續 . . .
```

然而，若使用者在「起點站的站名」或「終點站的站名」輸入錯誤的話(也就是找不到此一站名)，則程式則會顯示錯誤訊息並一直詢問，直到使用者輸入正確的站名。例如



Note: 你的程式必須是一般性的寫法，並非只針對目前提供的資料而已。也就是，若任意修改你程式中的 `price` 二維陣列的內容及長度，則你的程式必須仍能正確執行。

4. 在第 1 題中，price 及 station 陣列的資料為事先給定。試發展一個 C# 程式，price 及 station 陣列的資料為動態輸入，也就是藉由詢問使用者下列問題而得到 price 及 station 陣列的資料。首先，

請問總共有幾個站：

之後詢問每一站的站名，

請輸入站名：

之後詢問站與站之間的票價

| | | | |
|------|------|------|---------------------------------|
| 起站：A | 終站：B | 票價為： | <input type="text" value="XX"/> |
| 起站：B | 終站：A | 票價為： | <input type="text" value="XX"/> |
| 起站：A | 終站：C | 票價為： | <input type="text" value="XX"/> |
| 起站：C | 終站：A | 票價為： | <input type="text" value="XX"/> |
| 起站：B | 終站：C | 票價為： | <input type="text" value="XX"/> |
| 起站：C | 終站：B | 票價為： | <input type="text" value="XX"/> |

注意，詢問完起站為 A 終站為 B 的票價後，接下來要立刻詢問起站為 B 終站為 A 的票價，依此原則問完所有以 A 為起站後，再詢問以 B 為起站的問題。兩個範例如下所示。

```

請問總共有幾個站：3
請輸入站名：東京
請輸入站名：大阪
請輸入站名：名古屋

請輸入票價
起站：東京 終站：大阪 票價為： 3000
起站：大阪 終站：東京 票價為： 2800
起站：東京 終站：名古屋 票價為： 1800
起站：名古屋 終站：東京 票價為： 2200
起站：大阪 終站：名古屋 票價為： 800
起站：名古屋 終站：大阪 票價為： 900

```

```

請問總共有幾個站：4
請輸入站名：台北
請輸入站名：桃園
請輸入站名：台中
請輸入站名：高雄

請輸入票價
起站：台北 終站：桃園 票價為： 140
起站：桃園 終站：台北 票價為： 130
起站：台北 終站：台中 票價為： 480
起站：台中 終站：台北 票價為： 520
起站：台北 終站：高雄 票價為： 950
起站：高雄 終站：台北 票價為： 980
起站：桃園 終站：台中 票價為： 550
起站：台中 終站：桃園 票價為： 430
起站：桃園 終站：高雄 票價為： 880
起站：高雄 終站：桃園 票價為： 870
起站：台中 終站：高雄 票價為： 520
起站：高雄 終站：台中 票價為： 500

```

之後程式將各站間的資料列印出，如下所示

```

起站 終站 票價
台北 桃園 140
台北 台中 480
台北 高雄 950
桃園 台北 130
桃園 台中 550
桃園 高雄 880
台中 台北 520
台中 桃園 430
台中 高雄 520
高雄 台北 980
高雄 桃園 870
高雄 台中 500
請按任意鍵繼續 . . .

起站 終站 票價
東京 大阪 3000
東京 名古屋 1800
大阪 東京 2800
大阪 名古屋 800
名古屋 東京 2200
名古屋 大阪 900
請按任意鍵繼續 . . .

```

5. 假設微積分要進行課堂報告，但由於時間限制無法讓所有的人報告，因此要用抽籤方式決定人選。目前共有七人修課，其姓名以陣列方式儲存於程式中，如下所示

```
string[] Name = { "王一", "黃二", "張三", "李四", "陳五", "丁六", "鄭七" };
```

試發展一個 C# 程式，詢問要報告的人數，如下所示

```
請輸入人數：XX
```

之後以亂數的方式，抽出並顯示同學姓名，範例結果如下。

```

請輸入人數： 3
丁六
鄭七
王一
請按任意鍵繼續 . . .

```

```

請輸入人數： 3
鄭七
丁六
李四
請按任意鍵繼續 . . .

```

```

請輸入人數： 3
鄭七
王一
王一
請按任意鍵繼續 . . .

```

```

請輸入人數： 5
鄭七
丁六
陳五
黃二
陳五
請按任意鍵繼續 . . .

```

```

請輸入人數： 5
陳五
王一
李四
張三
鄭七
請按任意鍵繼續 . . .

```

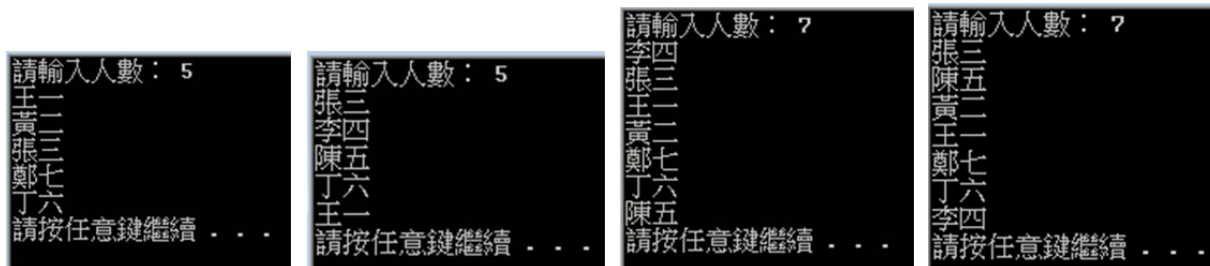
```

請輸入人數： 5
黃二
鄭七
王一
黃二
丁六
請按任意鍵繼續 . . .

```

Note: 本題可以允許同一個同學被抽超過一次(sampling with replacement)。

6. 承上題，微積分老師發現你的程式有問題，要你發展一個不會重複抽樣的程式(sampling without replacement)，範例結果如下。



7. 將七位同學的姓名及其成績分別以陣列方式儲存於程式中，如下所示

```
string[] Name = { "王一", "黃二", "張三", "李四", "陳五", "丁六", "鄭七" };  
int[] Chinese = { 80, 45, 60, 90, 20, 50, 70};
```

試發展一個 C# 程式，將同學姓名依其成績由高到低顯示出來。注意，你必須自己撰寫泡泡排序(Bubble Sort)方法，不得使用 C# 提供的任何排序方法。泡泡排序法的原理是將一組數字中的第一位與後一位相比較，若後一位數字較大，則位置對調，再將第二位數與第三位數做比較，若後一數字較大，再對調位置。上述資料的結果將如下所示。

