夏季學院通識計算機程式設計期末考

8/14/2020

試題共 8 題，兩面印製 10 頁，滿分 103

1. 撰寫一或數個C#敘述達成下列要求: (假設**`using System;`**敘述已經包含於程式中) 。

(a) 宣告整數常數 **`N = 10`**，再宣告一個長度為 **`N`** 的一維整數陣列 **`fib`** (3%)

(b) 寫一個迴圈，將 **`fib`** 的每個元素都設為 **`0`** (3%)

(c) 設定 **`fib[1]`** 為 **`1`**，再寫一個迴圈，迴圈變數 **`n`** 從 **`2`** 開始，在迴圈內設定 **`fib[n]`** 為其前兩項 **`fib[n-1]`**、**`fib[n-2]`** 之和 (3%)

(d) 宣告並設定一個整數變數 **`match`** 之值為 **`21`**，利用 C# 提供的函式 **`Array.IndexOf`** 在陣列 **`fib`** 中尋找與 **`match`** 相同之元素的索引。如果沒找到，此一索引值會是 **`-1`** (3%)

(e) 利用 C# 提供的函式 **`Array.Reverse`**，將陣列 **`fib`** 的元素倒過來排列 (3%)

2. 撰寫一或數個C#敘述達成下列要求: (假設**`using System;`**敘述已經包含於程式中) 。

(a) 宣告一個二維陣列 **`trans`**，同時設定初值，用以表示如 表1 所示： The Peach Blossom Spring Town、Gotham City 兩地區，今年第一季(一到三月)的房地產交易值 (3%)

表1. 今年第一季(一到三月)兩地區的房地產交易值(單位:億元)

	The Peach Blossom Spring Town	Gotham City
January	135	246
February	95	183
March	120	212

(b) 宣告整數變數 **`maxi`** 和 **`maxj`** ，並利用 C# 提供的函數 **`GetUpperBound`** ，分別設定 **`maxi`** 和 **`maxj`** 為陣列 **`trans`** 對應的列索引(row index)和行索引(column index)最大值 (3%)

(c) 宣告整數陣列 **`rowSum`**，以迴圈設定其第 **`i`** 個元素為第 **`i`** 個月中，兩個地區的交易量總和 (3%)

(d) 宣告整數陣列 **`colSum`**，以迴圈設定其第 **`j`** 個元素為第 **`j`** 個地區在第一季三個月的交易量總和 (3%)

(e) 宣告整數變數 **`total`** ，以 **`foreach`** 迴圈敘述，設定其值為兩個地區第一季的所有房地產交易量總和 (3%)

3. 撰寫一或數個C#敘述達成下列要求: 丟擲一般桌遊常見正八面體骰子(octahedron dice) 10,000次後，分別累計點數 1 到 8 的出現次數。程式執行後，螢幕輸出截圖如圖1。(令 **`using System;`** 敘述已經包含於程式中)

(a) 宣告一個亂數產生器 **`rand`**，其種子數 **`seed`** 為 **`777`** (3%)

(b) 宣告一個 **`static int`** 函數 **`ThrowOctahedronDice`**，以一個亂數產生器 **`rand`** 為引數，大括弧中的內容空白 (3%)

(c) 完成函式 **`ThrowOctahedronDice`** 的內容：以引數 **`rand`** 產生一個隨機正整數，求此正整數除以 **`8`** 的餘數加 **`1`**，就可以得到一個介於 **`1`** 到 **`8`** 的整數，作為傳回值，代表某次丟擲八面骰子時，向上正面顯現的點數 (3 %)

(d) 在主程式 **`Main`** 中，宣告整數常數 **`N_VALUES = 8`** 代表正面點數的個數。宣告長度為 **`N_VALUES`** 的整數陣列 **`accum`**，用來累計各個點數出現的次數。再其次宣告整數常數 **`N_TRIALS = 10000`** (3%)

(e) 寫一個迴圈，呼叫 **`ThrowOctahedronDice`** 函式 **`N_TRIALS`** 次，在陣列**`accum`**的對應元素累計各點數出現的次數。完成迴圈後，用 **`Console.WriteLine`** 印出結果如圖 1 (3%)

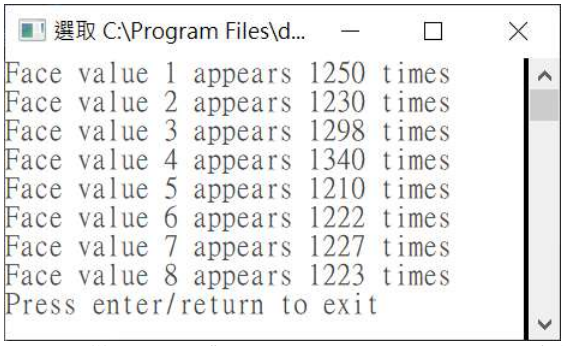
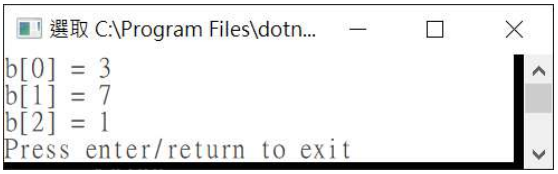


圖1. 丟擲正八面體骰子(octahedron dice) 10,000次後，點數 1 到 8 的出現次數分布

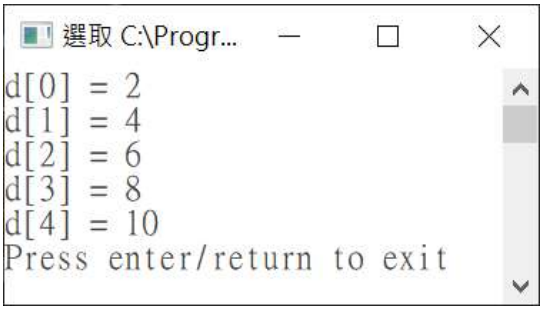
4. 指出以下程式片段之錯誤，並在盡量保持原先程式碼之前提下，予以更正。假設**`using System;`**敘述已經包含於程式中。同時請忽略螢幕最後一行文字: “Press enter/return to exit”。

(a) (3%) (一個語義錯誤) 執行時螢幕應顯示



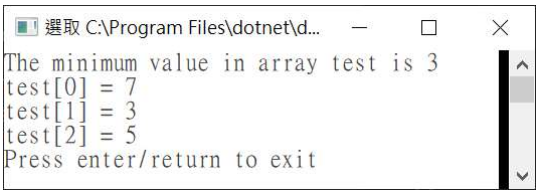

```
int[] a = {3, 7, 1};
int[] b = new int[3];
b = a;
Array.Reverse(a);
for(int idx = 0; idx < 3; ++idx)
{
   Console.WriteLine("b[" + idx + "] = " +  b[idx]);
}
```

(b) (3%) (一個語義錯誤) 執行時螢幕應顯示



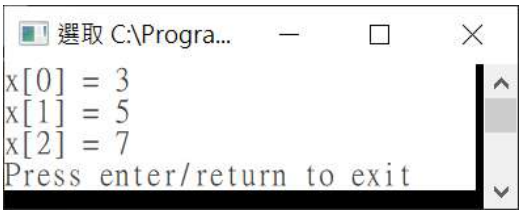

```
int[] d = {1, 2, 3, 4, 5};
for(int idx = 0; idx <= d.Length; ++idx)
{
  d[idx] *= 2;
  Console.WriteLine("d[" + idx + "] = " + d[idx]);
}
```

(c) (3%) (一個語義錯誤) 執行時螢幕應顯示




```
static int Minimum(int[] x)
{
   Array.Sort(x);
   return x[0];
}
static void Main(string[] args)
{
   int[] test = {7, 3, 5};
   int min = Minimum(test);
   Console.WriteLine(
     "The minimum value in array test is " + min);
   for(int idx = 0; idx < 3; ++idx)
   {
     Console.WriteLine("test[" + idx + "] = " + test[idx]);
   }
}
```

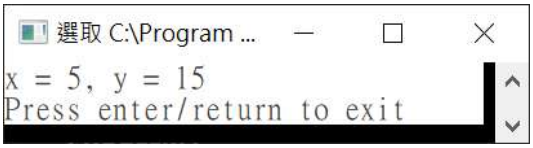
(d) (3%) (一個語義錯誤) 執行時螢幕應顯示如下:




```
static void Increment(int x)
{
   ++x;
}

static void Main(string[] args)
{
   int[] s = {2, 4, 6};
   for(int idx = 0; idx < 3; ++idx)
   {
      Increment(s[idx]);
      Console.WriteLine("x[" + idx +  "] = " + s[idx]);
   }
}
```

(e) (3%) (一個語法錯誤) 執行時螢幕應顯示如下:




```
static void TripleX(int x, int y)
{
   y = 3*x;
}

static void Main(string[] args)
{
  int x = 5;
  int y;
  Triple(x, y);
  Console.WriteLine("x = {0}, y = {1}", x, y);
}
```

5. 試寫出下列程式的螢幕輸出。 (5 %)

```
using System;
namespace Problem5
{
    class Program
    {
        static void Main(string[] args)
        {
            const int N = 25;
            const int M = 7;
            int n = 0;
            for(int i = 1; i < N; ++i)
            {
                if(i / M * M == i) ++n;
                Console.WriteLine("i = {0}, n = {1}", i, n);
            }

            Console.WriteLine("final n = " + n);

            Console.WriteLine("Press enter/return to exit");
            Console.ReadLine();
        }
    }
}
```

6. 試寫出下列程式的螢幕輸出 (修改自

https://www.eximiaco.tech/en/2019/11/17/computing-the-levenshtein-edit-distance-of-two-strings-using-c/) (10 %)

```
using System;

namespace Problem6
{
    class Program
    {
```

```
static int Minimum(int e1, int e2, int e3)
{
    int r1 = Math.Min(e1, e2);
    int result = Math.Min(r1, e3);
    return result;
}
static int ComputeTheDistanceBetween(string s1, string s2)
{
    char[] first = s1.ToCharArray();
    char[] second = s2.ToCharArray();
    if (first.Length == 0)
    {
        return second.Length;
    }
    if (second.Length == 0)
    {
        return first.Length;
    }
    int[,] d = new int[first.Length + 1, second.Length + 1];
    for (int i = 0; i <= first.Length; i++)
    {
        d[i, 0] = i;
    }
    for (int j = 0; j <= second.Length; j++)
    {
        d[0, j] = j;
    }
    for (int i = 1; i <= first.Length; i++)
    {
        for (int j = 1; j <= second.Length; j++)
        {
            int cost = (second[j - 1] == first[i - 1]) ? 0 : 1;
            d[i, j] = Minimum(
                d[i - 1, j] + 1,
                d[i, j - 1] + 1,
                d[i - 1, j - 1] + cost
            );
        }
```

```
            }
            return d[first.Length, second.Length];
        }
        static void DisplayResults(string s1, string s2, int d)
        {
            Console.WriteLine("distance between {0} and {1}: {2}",
                s1, s2, d);
        }
        static void Main(string[] args)
        {
            int d = 0;
            string s1 = "";
            string s2 = "";

            s1 = "";
            s2 = "aunt";
            d = ComputeTheDistanceBetween(s1, s2);
            DisplayResults(s1, s2, d);

            s1 = "kitten";
            s2 = "";
            d = ComputeTheDistanceBetween(s1, s2);
            DisplayResults(s1, s2, d);

            s1 = "ant";
            s2 = "aunt";
            d = ComputeTheDistanceBetween(s1, s2);
            DisplayResults(s1, s2, d);

            s1 = "kitten";
            s2 = "sitting";
            d = ComputeTheDistanceBetween(s1, s2);
            DisplayResults(s1, s2, d);
            Console.WriteLine("Press enter/return to exit");
            Console.ReadLine();
        }
    }
}
```

7. 時間序列(time series)是一組按照時間發生先後順序進行排列的數據點序列，常用於財務金融、經濟分析、醫學公衛、企業銷售、社會政治、電機資訊的訊號分析等等(所以學經濟的台大管中閔校長，也可能在國際電機電子工程師學會的期刊發表論文)。時間序列隨時間的改變，既有長時間的趨勢，也有短時間或者週期性的變化。尋求長期趨勢的方法之一，稱為移動平均(moving average)。以下用圖 2 所示，美國佛羅里達州新冠肺炎(COVID-19)於 2020 年 3 月到 7 月的每日死亡人數統計，說明移動平均概念(參看 https://statisticsbyjim.com/time-series/moving-averages-smoothing/)。圖 2 的數據顯示死亡率每天都有快速的變動，比較不容易找出長期變化的趨勢。要除去這些短期改變，最直接的想法，便是選擇一個適當的期間(稱為「窗框」(window)，這裡令為 7 天)，計算這一段時間的平均值，把短期變化消除；接著將窗框移動一天，重新計算平均，再移動一天，再算一次平均，以此類推，可得到如圖 2 中的紅色曲線。COVID-19 死亡率先上升、拉平、下降、再爬升的趨勢就很容易觀察到。這種方法，也稱為平滑化(smoothing)方法。

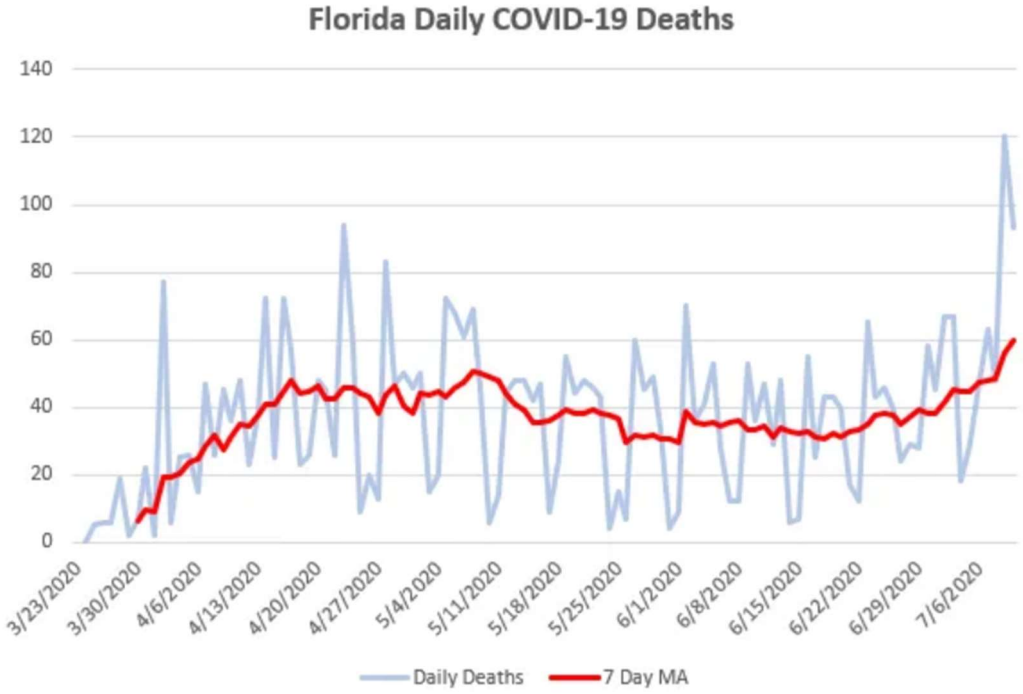


圖2. 美國佛羅里達州新冠肺炎(COVID-19)於2020年3月到7月的每日死亡人數統計
取自 https://statisticsbyjim.com/time-series/moving-averages-smoothing/

基本上，計算移動平均有兩種方法，第一種將窗框平均值，放在窗框最後一點對應的時間點，稱為單邊移動平均(one-sided moving average)，亦即 $MV_t = (X_{t-6} + X_{t-5} + X_{t-4} + X_{t-3} + X_{t-2} + X_{t-1} + X_t)/7$，其中$X_t$代表原先時間序列(圖2中的淺藍色線)，$MV_t$則是在時間$t$算出的單邊移動平均(圖2中的紅色線)。有了移動平均，可以算出短期變動(variation) $V_t = X_t - MV_t$。

第二種方法，則是將窗框平均值放在窗框中心，稱為中心移動平均(centered moving average)。計算方式為 $MV_t = (X_{t-3} + X_{t-2} + X_{t-1} + X_t + X_{t+1} + X_{t+2} + X_{t+3})/7$。對應的短期變動也可由 $V_t = X_t - MV_t$ 算出。

這兩種移動平均的方法，都是數位訊號處理(digital signal processing，簡稱DSP)領域中，Finite Impulse Response (簡稱FIR) 數位濾波器(digital filter)的特例，有相關理論可分析其特性。

本題希望你能夠寫一個程式，計算美國佛羅里達州新冠肺炎(COVID-19)於2020年3月22日到4月27日，共36天，每日死亡人數統計的單邊移動平均及中心移動平均，以及對應的短期變動。令窗框大小均為 7。程式執行的螢幕截圖如圖3。

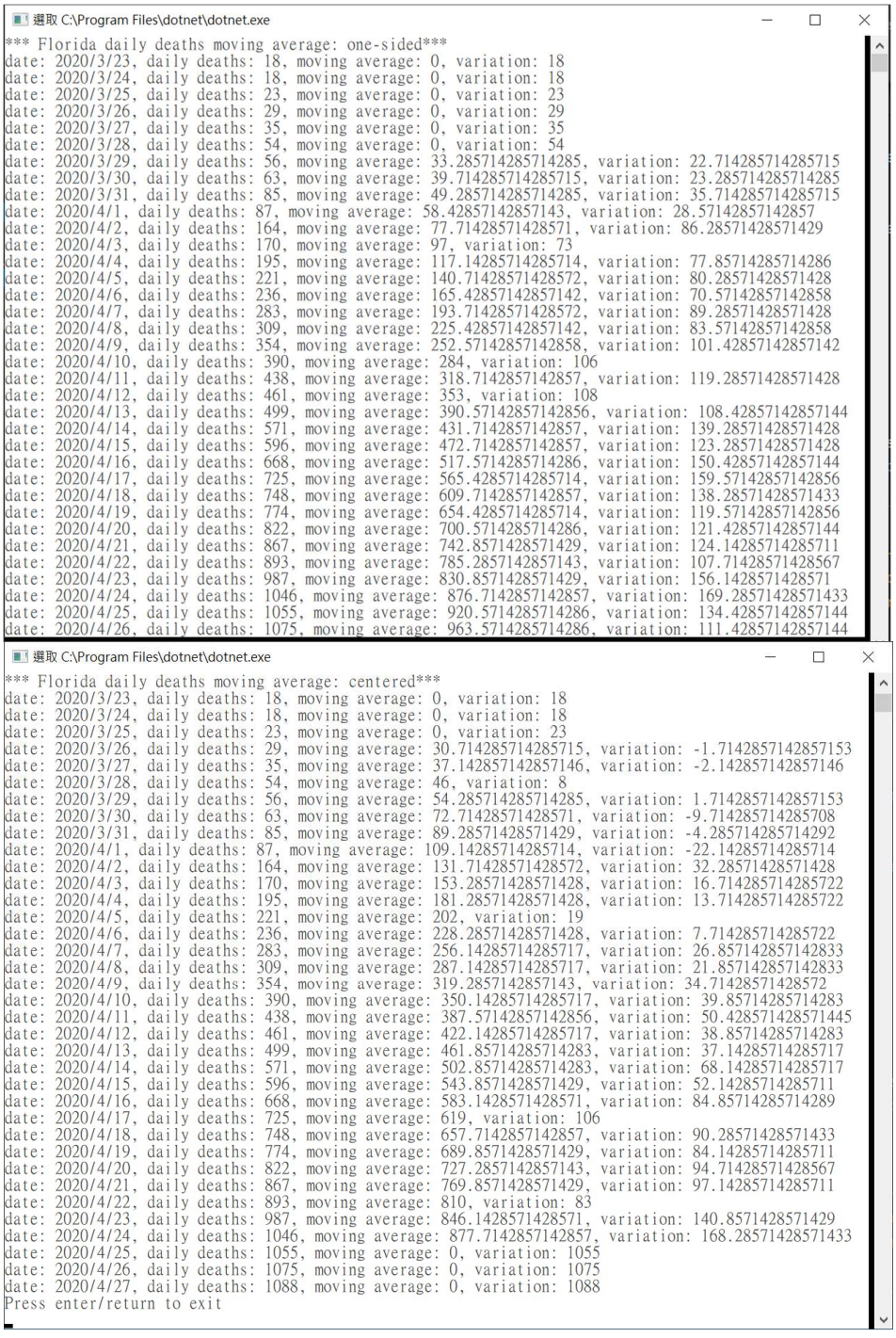


圖3．佛羅里達州 COVID-19 於 3/22/2020 至 4/27/2020 的單邊移動平均及中心移動平均計算結果螢幕截圖。

本題所需的數據資料，假定已寫在主程式中如下，可以直接應用：

```
const int N_DAYS = 36;
string[] date = {
  "2020/3/23", "2020/3/24", "2020/3/25", "2020/3/26",
  "2020/3/27", "2020/3/28", "2020/3/29", "2020/3/30",
  "2020/3/31", "2020/4/1",  "2020/4/2",  "2020/4/3",
  "2020/4/4",  "2020/4/5",  "2020/4/6",  "2020/4/7",
  "2020/4/8",  "2020/4/9",  "2020/4/10", "2020/4/11",
  "2020/4/12", "2020/4/13", "2020/4/14", "2020/4/15",
  "2020/4/16", "2020/4/17", "2020/4/18", "2020/4/19",
  "2020/4/20", "2020/4/21", "2020/4/22", "2020/4/23",
  "2020/4/24", "2020/4/25", "2020/4/26", "2020/4/27"
};

int[] dailyDeaths = {
  18,         18,         23,         29,
  35,         54,         56,         63,
  85,         87,         164,        170,
  195,        221,        236,        283,
  309,        354,        390,        438,
  461,        499,        571,        596,
  668,        725,        748,        774,
  822,        867,        893,        987,
  1046,       1055,       1075,       1088
};
```

本題滿分 25 分，全部程式集中寫成一個大 Main 函式，不區分函式者，最高得 20 分；善用函式，乃至尚未教到的物件導向程式設計(object-oriented programming)者，最高得 23 分；能利用虛擬碼或流程圖思考，適當劃分函式或類別(class)者，最高得 25 分(使用虛擬碼)或 24 分(使用流程圖)。(25%)

8. 請寫下本課程教學「待改進」之處及改進方法建議。(3%)