# 夏季學院通識計算機程式設計期末考

8/13/2021

試題共 4 大題，滿分 100

注意：本次考試可以自行參考各種書籍與網路資源，協助答題，但須於答案中敘明相關訊息來源。同時，不可以給予或接受其他人的幫助。答案中沒有註明課外重要參考資訊來源，或發現有答題內容顯然與其他同學雷同，將視為違反學術倫理，依照課程規定，本次考試成績以零分計，並通知同學學校。

## 壹. 選出每一小題最適合的答案 (30%)

1. 設定一維陣列 a 的元素之初值何者正確？

   (A) a = (1, 2, 3, 4, 5);

   (B) a = [1, 2, 3, 4, 5];

   (C) a = {1, 2, 3, 4, 5};

   (D) a = <1, 2, 3, 4, 5>;

2. 如何取得一維陣列之長度？

   (A) a.length

   (B) a.Length

   (C) a.length()

   (D) a.getLength()

3. 關於 k 的敘述何者正確？

```
Random rand = new Random();
int k = rand.Next();
```

(A) Int32.MinValue 到 0 之間的整數

(B) 0 到 Int32.MaxValue 之間的整數

(C) Int32.MinValue 到 Int32.MaxValue 之間的整數

(D) 無限定範圍的整數

4. 假設以下程式執行時間趨近於 0，下列敘述何者正確？

```
Random rand1 = new Random();
int i = rand1.Next();
Random rand2 = new Random();
int j = rand2.Next();
```

(A) i 必等於 j

(B) i 必不等於 j

(C) i 不一定等於或不等於 j

5. 將陣列 a 設值給陣列 b、陣列 c、陣列 d，下列敘述何者正確？

```
b = a;
Array.Reverse(b);

c = new int[N];
for (k = 0; k < N; ++k)
{
    c[k] = a[k];
}
Array.Reverse(c);

d = new int[N];
Array.Copy(a, d, N);
Array.Reverse(d);
```

(A) a 隨著 b 反轉

(B) a 隨著 c 反轉

(C) a 隨著 d 反轉

6. 資料型別分為**實質型別(Value Type)**與**參考型別(Reference Type)**，實質型別變數的記憶體空間存放的是**實際的值**，參考型別變數的記憶體空間存放的是值的**記憶體位置**，以下和者屬於參考型別？

(A) 布林值 (bool)

(B) 陣列 (array)

(C) 列舉型別 (enum)

(D) 結構型別 (struct)

7. 一個函數最多有幾個回傳值？

(A) 0

(B) 1

(C) 2

(D) 無限多個

8. 請問 score.GetUpperBound(0)的值為何？

```
int[,] score = { {90, 84},
                 {88, 86},
                 {86, 92} };
```

(A) 1

(B) 2

(C) 3

(D) 90

(E) {90, 84}

9. 請問下列 statement 的執行順序為何？

```
static void Function1()
{
    Function2();
    statement1;
}
static void Function2()
{
    statement2;
}
static void Main(string[] args)
{
    statement3;
    Function1();
    statement4;
}
```

(A) statement1→statement2→statement3→statement4

(B) statement3→statement1→statement2→statement4

(C) statement3→statement2→statement1→statement4

(D) statement1→statement3→statement4→statement2

10. 對於函式標頭為 void 的函式，下列敘述何者正確？

(A) 函式中必須 return 數值

(B) 函式中必須有 return，但不須有數值

(C) 函式中不能有 return

(D) 以上皆非

## 貳. 程式實作: 星期換算小幫手　(20%)

對於一個行程滿檔的人來說，行事曆就是一個非常重要的存在。

大家都知道，我們的柏森助教一年到頭總是非常地忙碌，甚至忙到連幾號是星期幾都搞不清楚了！因此拜託同學們協助柏森助教完成一個顯示小日曆的程式，讓他至少可以看著日期就知道當天是星期幾。 另外，貼心的助教已經幫同學寫好了一部分的程式碼（詳見圖 2.3 Unfinished Code），請接力完成他的程式碼，加入 **SetCalendar** 和 **PrintCalendar** 兩個函式來完成作答，未加入這兩個函式不予計分。

### Input

有兩個數字 **a**（**28 <= a<= 31**）和 **b**（**1<=b<=7**），
**a** 代表當月有幾天，
**b** 則代表第一天為星期幾（1=Sunday,2=Monday, …… ,7=Saturday）

### Output

印出小日曆，排版請參考範例輸出圖 2.1 及 圖 2.2。

### Sample Input and Output

```
31 1
   1   2   3   4   5   6   7
   8   9  10  11  12  13  14
  15  16  17  18  19  20  21
  22  23  24  25  26  27  28
  29  30  31
```

圖 2.1 輸入 **31 1**，輸出相對應的小日曆

```
30 5
                       1   2   3
   4   5   6   7   8   9  10
  11  12  13  14  15  16  17
  18  19  20  21  22  23  24
  25  26  27  28  29  30
```

圖 2.2　輸入 **30 5**，輸出相對應的小日曆

```
using System;
namespace Program
{
    class Program
    {
        static void Main(string[] args)
        {
            int[] input_params = Array.ConvertAll(Console.ReadLine().Split(), int.Parse);
            int date_in_month = input_params[0];
            int first_day = input_params[1];
            string[] calendar=new string[date_in_month+first_day-1];
            SetCalendar(ref calendar,date_in_month,first_day);//設定小日曆陣列
            PrintCalendar(ref calendar);//顯示小日曆
        }
    }
}
```

圖 2.3　Unfinished Code

## 參. 程式實作:

### 粒子群最佳化演算法(簡化版)-Particle Swarm Optimization **(25%)**

相信大家都聽過「深度學習」、「人工智慧」，
出題者認為，機器學習是一種跨領域知識的完美產物，可說是人類智慧的結晶！

但你是否有想過：「這些 cool 東西，到底背後的基礎學問是什麼呢？」

「最佳化演算法」正是其中一樣基礎知識，且 optimization algorithm 不只用於機器學習，也已普遍用於各種領域，舉凡 元件設計、水利系統、運輸物流系統、人體肌肉力學…等，都可以找到相關研究成果。

「粒子群最佳化」(PSO)正是其中一個類別，主要是透過擁有不同資料的「粒子」，彼此間的「資料交換」、融合，不斷的「產生新的粒子」或「改變現有粒子的狀態」，來找到我們欲解決問題的最佳解。

這種「粒子間的資料交換」、「產生新的粒子」、「改變粒子的狀態」的概念，正

是弱人工智慧的一種模型。

事實上早在 class11-Q3 的題目中，提到的勘根法，就可說是一種簡化的粒子群最佳化演算法，圖 3.1 是以「粒子演化」的邏輯來撰寫勘根法的虛擬碼。透過不斷更新的粒子，讓我們得以逐步逼近已知函數的根(問題的最佳解)。

```
// 使用勘根定理前，需先找到兩個點，使得f(x1)*f(x2) < 0
// 才能確保函數於x1~x2範圍內必定有根
Do
    於給定範圍內隨機創造2個particle, a與b
    給予a與b隨機初始值位置, xa與xb
While ( f(xa)*f(xb) > 0 ) // f就是你的代測函數(欲解決的問題)

Do
    // 產生1個子代particle
    於xa, xb區間算出xc // 可以直接取中間值，或用隨機挑選ab間的數

    // 讓particle c取代，代入f後會同號的particle (a或b)
    If ( f(xa) * f(xc) > 0 )
        xa = xc
    Else
        xb = xc
While ( abs(xa-xb) > 0.0001 ) // 0.0001是你容許x的誤差，可以隨需求設定
print("勘根法找出的root位置位於{xc}")
```

圖 3.1 勘根法的虛擬碼

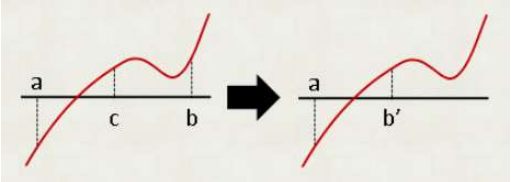



圖 3.2、勘根法的視覺化流程[1]

以下會以虛擬碼(圖 3.3)及視覺化流程(圖 3.,4)，介紹另一種簡化後的粒子群最佳化(PSO)模型，請同學用此模型來找出給定函數的極小值與極小值位置。

```
於給定範圍內創造N個paritcle
// 你可以使用Array來儲存所有particle的位置
給予particles各一個隨機初始位置x[i]: i = 0 ~ N-1
// 設定最大世代數
maxGeneration = 1000

for (generation=1; generation<=maxGeneration; generation++)
    // f 就是你的待測函數(欲解決的問題)
    // 計算每一個particle的y值同時，順便找出目前的最佳解(極小值)
    for (i=0; i<N; i++)
        y[i] = f(x[i])
        if (y[i] < ymin)
            ymin = y[i]
            xmin = x[i]

    // 取得當前世代的最小值後
    // 便可開始讓每一個particle朝有最小值的particle靠近
    for (i=0; i<N; i++)
        // particle移動的步長會是一個隨機值
        // 步長越大，會一口氣靠近最小值的點越多
        stepSize = (xmin - x[i]) * rand.NextDouble()
        x[i] += stepSize
print("easyPSO找到的函數極小值為{ymin}, 位於{xmin}")
```

圖 3.3、簡化版 PSO 的虛擬碼

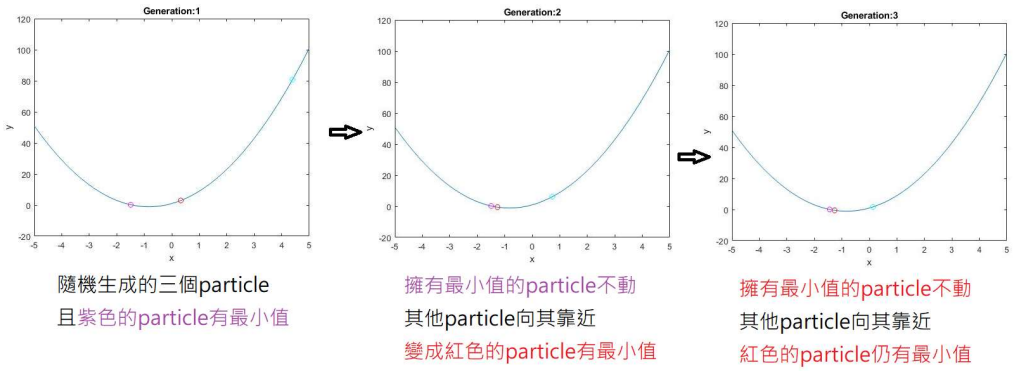



圖 3.4 簡化版 PSO 的視覺化流程

請同學根據圖 3.3 的虛擬碼，寫出一個實際能執行的 easyPSO，並用其找出 1-D Griewank's Function 的極小值，及其極小值位置。

$$f(x) = \frac{x^2}{4000} - \cos(x) + 1, x \in [-100,100]$$

建議：你可以將粒子數(N)設為 100，最大世代數(maxGeneration)設為 1000 範例輸出如圖 3.5 。

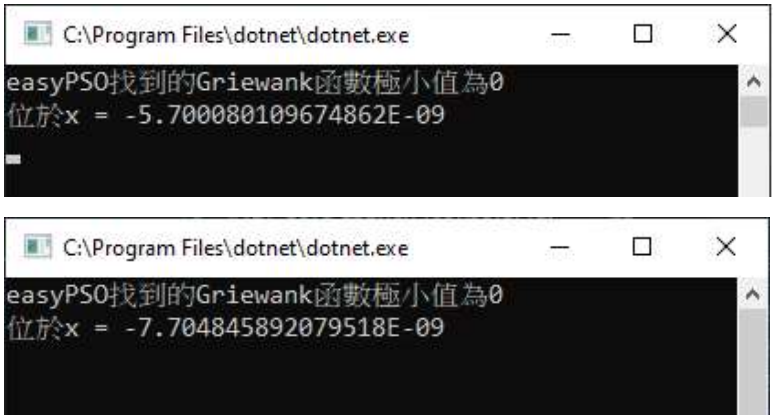



圖 3.5 輸出範例

參考文獻

[1] 台師大演算法筆記 http://web.ntnu.edu.tw/~algo/RootFinding.html

## 肆. 程式實作: 雙盲實驗設計 (25%)

近年來新冠肺炎疫苗的研發成為相當重要的課題，而疫苗抵抗病毒能力的有效程度更是成功的關鍵。疫苗效力的檢驗，需要嚴謹的實驗證實。為確保實驗能夠排除許多實驗設計上的人為以及系統性的偏差(bias)，雙盲實驗(blind experiments)是常見的方法。

根據 Wikipedia 的解說[1]：**雙盲**是科學方法的一種，目的是避免研究結果受安慰劑效應或觀察偏差所影響；廣泛應用於各個學術領域，尤其是藥物的研發。
雙盲實驗的過程其實相當繁複，網頁[2]顯示的流程就是一例。我們這裡只討論一個相當簡化的版本。

首先需要收集受試者的接受實驗同意書，通常稱為「收案」(enrollment)。接著得把所有受試者隨機分為實驗組(treatment group)，注射要實驗的疫苗，以及對照組(control group)，打安慰劑。分組的原則是讓受試者以及實驗主持者與執行者都不知道誰分到實驗組，誰打安慰劑，因此稱為雙盲實驗。如果連實驗數據分析者也都不曉得分組名單，那就是「三盲」實驗了。

為了達成雙盲的要求，通常會給每個受試者一個亂數編號(label)，由實驗者或電

[1] https://zh.wikipedia.org/wiki/%E9%9B%99%E7%9B%B2 及 https://en.wikipedia.org/wiki/Blinded_experiment

[2] Chrome - extension://efaidnbmnnnibpcajpcglclefindmkaj/viewer.html?pdfurl=https%3A%2F%2Fwww.onenessbio.com.tw%2Ffile%2Finvestors%2F202006%2Fon101_0612.pdf&clen=299971&chunk=true

腦只根據亂數編號，隨機決定哪一個編號屬於實驗組，哪一個編號分為對照組。分組對應表從前封在信封中，妥善保管，不讓任何人看。現在則以電腦加密，存在硬碟或光碟中。隨後由不知情人員，為每個編號準備外觀完全相同的藥劑或安慰劑。當然，還需要產生一個身分證字號(ID)和亂數編號的對照表(沒有分組訊息)。

完成受試者分組後，就開始讓受試者開始接受注射，實驗執行者依照受試者編號施打。注意，受試者可以自由選擇施打時間。所以，實際施打順序不會和分組對應表的排列順序相同。受試者接受注射後一段時間，由實驗室檢測其產生抗體的能力。

所有受試者的實驗室結果都產生後，進行解盲(unblind)：根據先前保存的分組對應表，將受試者的資料，分別分到實驗組與控制組的資料檔，由數據分析師依據統計學，判斷疫苗藥劑是否產生所需的效力。

本題希望你完成一個 **C#** 程式，能夠模擬雙盲實驗的流程。首先，假設給定受試者身分證字號，運用亂數產生器函式 **rand.Next()** 為每位受試者產生一個亂數編號，再用另一個亂數是否為偶數，決定受試者分配到控制組或實驗組。所完成的分組對應表如表 4.1 所示。

表 4.1 分組對應表

| ID_enrollment | label_enrollment | grouptypes_enrollment |
|---|---|---|
| A*102 | 614893002 | Control |
| G*406 | 209388840 | Treatment |
| B*503 | 643272181 | Treatment |
| F*205 | 1026471585 | Control |
| D*811 | 1103468486 | Control |
| C*709 | 761956083 | Treatment |

其次變動亂數編號順序，模擬受試者接受注射的先後順序，並且以亂數產生器函式 **rand.NextDouble()** 生成一個 **0** 與 **1** 之間的小數，代表疫苗效力得強弱。實驗的結果依照實驗順序，列出實驗結果表如表 4.2。

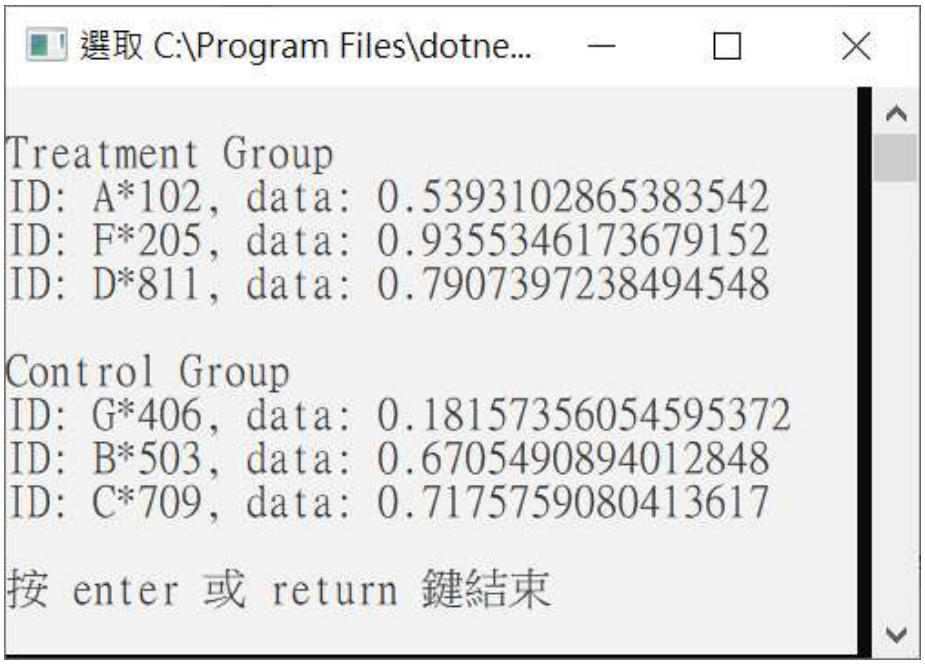
表 4.2 實驗結果表

| label_trial | data_trial |
|---|---|
| 1103468486 | 0.79073 |
| 643272181 | 0..667054 |
| 614893002 | 0.53931 |
| 1026471585 | 0.93553 |
| 761956083 | 0.717575 |
| 209388840 | 0.18157 |

然後進行解盲，其演算法可以用虛擬碼說明如下:

```
i_treatment = 0
i_control = 0
for 所有受試者依照收案順序
{
  由分組對應表決定受試者 ID, 亂數編號 label, 及分組別 group
  根據亂數編號找到實驗結果表的對應位置以及對應實驗數據 data
  if(group == Treatment)
  {
    將對應的 ID、data 分別放入陣列 IDs_treatment 和
      data_treatment 在 i_treatment 的位置
    ++i_treatment
  }
  else
  {
    將對應的 ID、data 分別放入陣列 IDs_control 和
      data_control 在 i_control 的位置
    ++i_control
  }
}
```

最後將陣列 **IDs_treatment**、**data_treatment**、**IDs_control**、**data_control** 顯示在主控台螢幕，如圖 4.1 即可。後續的工作，就交由統計學者完成。




注意程式只要處理以下程式敘述的受試者資料就可以:

```
string[] IDs_enrollment =
{
  "A*102", "G*406", "B*503",
  "F*205", "D*811", "C*709"
};
```

同時，可以呼叫如下兩個函式，完成實驗模擬:

```
static void ShuffleLabels(
  int[] labels_enrollment, int[] labels_trial)
{
  labels_trial[0] = labels_enrollment[4];
  labels_trial[1] = labels_enrollment[2];
  labels_trial[2] = labels_enrollment[0];
  labels_trial[3] = labels_enrollment[3];
  labels_trial[4] = labels_enrollment[5];
  labels_trial[5] = labels_enrollment[1];
}
```

```
static void ConductExperiment(Random rand,
  int[] labels_trial, double[] data_trial)
{
  int n = labels_trial.Length;
  for(int i = 0; i < n; ++i) // in experiment order
  {
     data_trial[i] = rand.NextDouble();
  }
}
```