通識計算機程式設計期中考

4/24/2020

試題共 7 題，兩面印製 12 頁，滿分 100

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 **bool** 變數 **success**， **int** 變數 **m**， **double** 變數 **r** (3%)

(b) 在螢幕顯示一行字，要求使用者輸入一個有小數點的實數 (3%)

(c) 自鍵盤讀入一個 **double** 數，並將其值存入已宣告之 **double** 變數 **r** (3%)

(d) 將 **r** 加 **0.5** 後，轉成整數，將結果存入 **int** 變數 **m** (3%)

(e) 檢查整數變數 **m** 數值是否為 **1**，將結果存入 **bool** 變數 **success** (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) 將已宣告設值之 **int** 變數 **n**，用算子 ***=**，計算 **n** 乘上他處已宣告設值的 **int** 變數 **factorial** 後，存入 **factorial** (3%)

(b) 宣告 **double** 變數 **theta** 及 **ct**。利用 C# 提供的數學函數，將π/2 設給 **theta**，並計算**theta**的餘弦函數值，把計算結果設為**ct** (3%)

(c) 宣告 **double** 變數 **threshold**，設定其初值為 **-80** 。利用三元運算子，使他處已宣告設值之 **double**變數 **v**，其數值大於等於 **threshold** 時，設定變數 **response** 的數值為 **1**，反之則令 **response** 值為 **0**。假設變數 **response** 已在他處宣告 (3%)

(d) 宣告字串變數 **s1**、**s2**，並且分別令其值為 **"MidTerm"** 及**"2020"**。接著宣告另一個字串變數 **testName**，設其值為 **s1** 和 **s2** 的 concatenation (連結為單一字串) (3%)

(e) 宣告變數 **lf** 為 **char** 型別，並令其值代表 new line (3%)

3. 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) 先以一個敘述宣告一個亂數產生器物件 **rand**，種子數為 **777** (3%)

(b) 宣告 **int** 常數 **LIST_SIZE**，並設其值 **100**。其次宣告一維 **int** 陣列 **grades**，長度為 **LIST_SIZE** (3%)

(c) 寫一個 **static int** 函式 **NewGrade**，其輸入參數為亂數產生器物件 **rand**。函式中利用 **rand.Next(100)** 得到一個 **0** 至 **100** 之間的隨機亂數，再將此一亂數數值傳回 (3 %)

(d) 寫一個 **for** 迴圈，迴圈控制變數 **i** 由 **0** 開始，每次加 **1**，遞增至

LIST_SIZE - 1。迴圈每次 iteration ，呼叫上一小題完成的 **static int** 函式 **NewGrade**，將傳回之值設為 **grades[i]**，**grades** 是(b)小題宣告的陣列 (3%)

(e) 將上一小題所得的 **grades** 陣列排序(sorting)，再顛倒順序排列(reverse)。此時 **grades[0]** 便是最高分，將其值設給他處宣告的整數變數 **topScore**，並輸出於螢幕 (3%)

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。

(a) (3%) (一個語義錯誤) 執行時螢幕應顯示


```
int s = 5;
int t = 5;
if(s < t && t > s++)
{
  t = 3;
}
else
{
  t = 5;
}
Console.WriteLine("s = {0}, t = {1}", s, t);
```

(b) (3%) (一個語義錯誤) 執行時螢幕應顯示

或


```
Console.Write("輸入成績: ");
int grade = Console.ReadLine();
if(grade > 60)
    Console.WriteLine("成績為: " + grade);
else
  Console.WriteLine("成績不及格");
```

(c) (3%) (一個語義錯誤) 執行時螢幕應顯示


```
int u = 3;
int v = 0;
if ( u >= 5 )
  if( u >= 7 )
    v = 1;
else
  v = -1;
Console.WriteLine("v =" + v);
```

(d) (3%) (一個語義錯誤) 執行時螢幕應顯示如下:


```
double b = 1.0;
double sum = 0.0;
do {
    sum += b;
    b /= 2;
} while (b <= 0.25);
Console.WriteLine("sum =" + sum);
```

(e) (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 x = 6;
    Increment(x);
    Console.WriteLine("x = " + x);
}
```

5. 試寫出下列程式的螢幕輸出 (5 %)

```
using System;
```

```
namespace Problem5
{
  class Program
  {
    static void GV()
    {
      double[ , ] vList = new double[ , ] {
        {0.0, 5.0, -5.0},
        {0.0, -5.0, -5.0},
        {0.0, -5.0, 5.0},
        {0.0, 5.0, 5.0},
        {10.0, 5.0, -5.0},
        {10.0, -5.0, -5.0},
        {10.0, -5.0, 5.0},
        {10.0, 5.0, 5.0}
      };

      double x = 0.0;
      double y = 0.0;
      double z = 0.0;
      for(int i = 0; i < vList.GetUpperBound(0)+1; ++i)
      {
        x = vList[i, 0];
        y = vList[i, 1];
        z = vList[i, 2];
        Console.WriteLine("v {0} {1} {2}", x, y, z);
      }
    }

    static void GN()
    {
      double[ , ] nList = new double[ , ] {
        {-1.0, 0.0, 0.0},
        {1.0, 0.0, 0.0},
        {0.0, -1.0, 0.0},
        {0.0, 1.0, 0.0},
        {0.0, 0.0, -1.0},
        {0.0, 0.0, 1.0}
      };

      double nx = 0.0;
      double ny = 0.0;
      double nz = 0.0;
      for(int i = 0; i < nList.GetUpperBound(0)+1; ++i)
      {
        nx = nList[i, 0];
        ny = nList[i, 1];
        nz = nList[i, 2];
        Console.WriteLine("vn {0} {1} {2}", nx, ny, nz);
      }
    }

    static void GF()
    {
      Console.WriteLine("f 1//1 2//1 3//1");
      Console.WriteLine("f 1//1 3//1 4//1");
      Console.WriteLine("f 5//2 7//2 6//2");
      Console.WriteLine("f 5//2 8//2 7//2");
      Console.WriteLine("f 2//3 6//3 7//3");
      Console.WriteLine("f 2//3 7//3 3//3");
      Console.WriteLine("f 1//4 8//4 5//4");
```

```
      Console.WriteLine("f 1//4 4//4 8//4");
      Console.WriteLine("f 1//5 5//5 2//5");
      Console.WriteLine("f 2//5 5//5 6//5");
      Console.WriteLine("f 3//6 7//6 8//6");
      Console.WriteLine("f 3//6 8//6 4//6");
    }

    static void Main(string[] args)
    {
      GV();
      GN();
      GF();

      Console.WriteLine("\n 按 enter 或 retuen 鍵結束");
      Console.ReadLine();
    }
  }
}
```

6. 試寫出下列程式的螢幕輸出[1] (10 %)

```
using System;

namespace Problem6
{
  class Program
  {
    static void LS (int source, int[,] w)
    {
      const int L = (int) 1.0e9;

      int n = w.GetUpperBound(0) + 1;
      int[] d = new int[n];
      int[] parent = new int[n];
      bool[] visited = new bool[n];
      for (int i=0; i<n; i++) visited[i] = false;

      d[source] = 0;
      parent[source] = source;
      visited[source] = true;
      for (int k=0; k<n-1; k++)
      {
        int a = -1;
        int b = -1;
        int min = L;
        for (int i=0; i<n; i++)
        {
          if (visited[i])
          {
            for (int j=0; j<n; j++)
            {
              if (!visited[j])
              {
                if (d[i] + w[i,j] < min)
                {
                  a = i;
                  b = j;
```

[1] 修改自 http://www.csie.ntnu.edu.tw/~u91029/Path.html#2

```
                min = d[i] + w[i,j];
              }
            }
          }
        }
      }
      if (a == -1 || b == -1) break;
      d[b] = min;
      parent[b] = a;
      visited[b] = true;
      Console.WriteLine("a = {0}, b = {1}, d = {2}", a, b, min);
    }
  }

  static void Main(string[] args)
  {
    const int L = (int) 1.0e9;
    int[,] w = new int[,] {
     {L, 5, L, 3, L, 2, L, L, L},
     {L, L, L, L, 3, L, L, L, L},
     {L, L, L, L, 1, L, L, L, L},
     {L, L, L, L, L, L, 5, L, L},
     {L, L, L, L, L, L, L, L, 5},
     {1, L, L, L, 0, L, L, L, L},
     {L, L, L, L, L, L, L, 3, L},
     {L, L, L, L, L, 1, L, L, L},
     {L, L, L, L, L, 2, L, 4, 2}
    };
    LS (0, w);
    Console.WriteLine("\n 按 enter 或 retuen 鍵結束");
    Console.ReadLine();
  }
 }
}
```

7. 能夠取代醫師的疾病自動診斷系統，一直是醫療資訊研究人員追求的目標。一九七 0 年代，史丹福大學團隊研發的細菌感染病自動診斷系統 MYCIN[2]，診斷正確率已經可以與受完整醫學訓練的醫師相提並論。MYCIN 的架構建立於許多推理規則與相關醫學數據之上，是第一個達到實用程度的人工智慧系統，結合醫學與電機資訊研究人員的努力，才得以完成：本校醫學系第一名畢業的傅立明[3]校友，是研發團隊的重要成員，並因此獲得史丹福大學電機工程博士，任教於美國知名大學的電機資訊系。

建造類似 MYCIN 的專家系統(expert system)，最重要的步驟是由教科書，以及訪談各個領域的專家，收集專家處理問題的過程及方法，從中萃取規則和數據。聽說當年有的名醫，發現自己數十年經驗累積的知識，不過就是數十條規則，頗有失落之感。

雖然類似 MYCIN 的系統，乃至近年來透過機器學習建構的人工智慧系統，診斷能力或許已經可以與人類醫師並駕齊驅，但是醫師看診還是有一些層面，電腦目前，或者在可知的未來，沒有辦法取代；例如：對病人外表言談舉止的觀察

[2] https://en.wikipedia.org/wiki/Mycin

[3] http://www.chineseoverseas.org/index.php/celebrity/show/id/309

與檢驗數據的整合，看診經驗累積的直覺等等。同時，實務上，在執業證照以及法律上的行為責任要求下，還是需要醫師下最後的診斷。因此，電腦資訊系統最好是提供客觀定量數據，推理建議可能的診斷，給醫師參考。電腦與人類專家的合作，勢必是未來許多領域，包括醫學，的主要發展趨勢(參見 Rangaraj M. Rangayyan, *Biomedical Signal Analysis*, 2nd ed., IEEE Press, 2015, pp. 55-57.)。

一般情況下，除了病人主訴的症狀外，如台北市長柯文哲在某個演講[4]中提到：醫師需要詢問病人病史等問題以取得必要資訊，從而利用貝氏定理，縮小問題範圍，做出正確的診斷。這是正確的說法，但是如果需要問的問題很多，問問題的順序，就變得很重要。問的好的話，只需幾個問題，就可以決定正確的診斷；問的沒效率的話，問再久也得不出所以然來。那要怎麼決定問題的先後呢？顯然鑑別率高的問題要先問，馬上可以排除許多可能。電腦科學中有一種利用大數據的機器學習方法，稱為決策樹(decision tree)[5]，正是分析收集到的資料，計算各種問題的分辨能力，建立起問題詢問順序的方法。

決策樹的建立，通常需要應用機率統計，也會用到略為複雜的資料結構，所以本題不要求同學寫出建構決策樹的程式(有興趣的同學可以參考網路資源[6])，而希望你能寫一個程式，根據已經建好的決策樹，詢問(inquiry)病人的症狀(symptom)，產生診斷給醫師參考。

我們在這一題的要求，是根據圖 1 的 COVID-19 問診流程，撰寫程式，自動依照病人的臨床條件、檢驗條件、流行病學條件，將病人分為極可能病例、高度懷疑之社區型肺炎患者、自主健康管理或隔離對象、正常或尚無症狀之感染者等四類。當然，此一決策樹並非由大數據建構，而是我們照常理推測的結果。

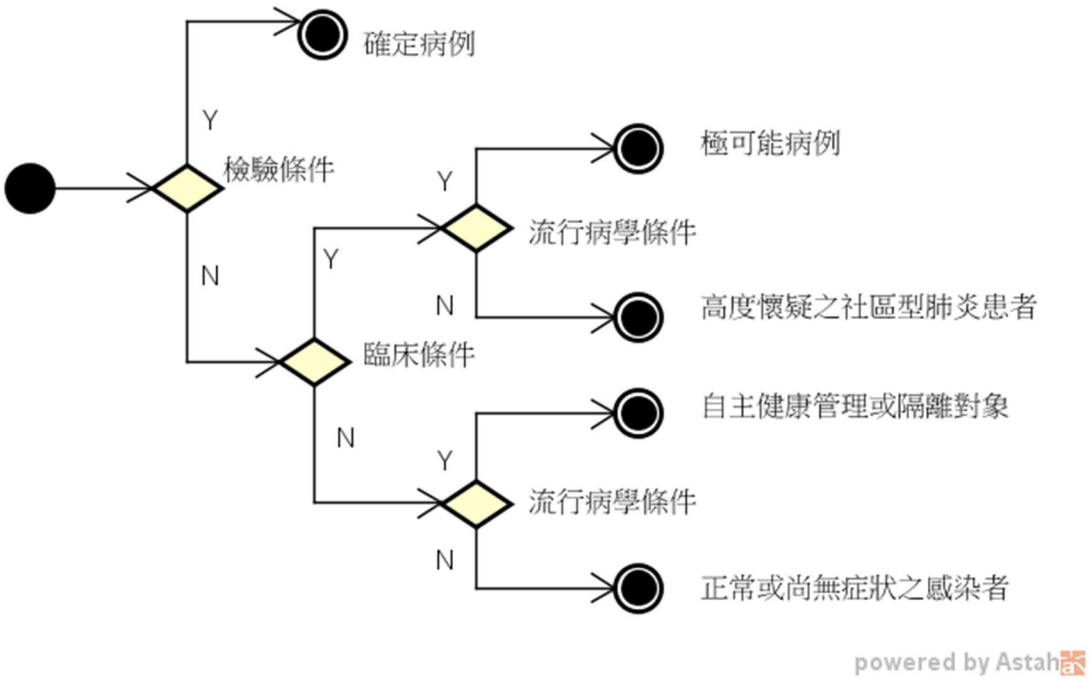

圖 1. COVID-19 問診流程決策樹

[4] http://zh.allreadable.com/d468M7Uo
[5] https://en.wikipedia.org/wiki/Decision_tree_learning
[6] 例如 https://visualstudiomagazine.com/articles/2020/01/21/decision-tree-classifier.aspx

此處提到的三種條件，依照中央流行疫情指揮中心的定義[7]，分別說明如下。

臨床條件:

1.發燒(≧38℃)或急性呼吸道感染或嗅、味覺異常。
2.臨床、放射線診斷或病理學上顯示有肺炎。
3.無明確旅遊史，醫師高度懷疑之社區型肺炎。

檢驗條件：

具有下列任一個條件

1. 臨床檢體如鼻咽或咽喉擦拭液、痰液或下呼吸道抽取液等，分離並鑑定出新型冠狀病毒。

2.臨床檢體新型冠狀病毒分子生物學核酸檢測陽性。

流行病學條件：

發病前14日內，具有下列任一個條件

1.有國外旅遊史或居住史，或曾接觸來自國外有發燒或呼吸道症狀人士。
2.曾經與出現症狀的極可能病例或確定病例有密切接觸(在無適當防護下提供照護、相處，或有呼吸道分泌物、體液之直接接觸)。
3.有群聚現象。

為簡化問題，以上三種條件，在本題中，不進一步考慮各種細節狀況。

圖1所示的決策樹，是計算機科學資料結構所討論之二元樹(binary tree)的特例。圖2是一般的完全二元樹，共有三層(layer)條件(C0至C6)，一層決策結果(D0至D7)。每一個個案在各個條件為真(True, T)或偽(False, F)或不需回答(Null, N)的結果，可以依照資料結構的討論，表示為如圖3的一維陣列。圖3陣列顯示：C0為真、C1為偽、C4為真，代表可沿圖2 中，C0、C1、C4路徑(粗箭線所示)，將個案歸入分類D2。

回到圖1的COVID-19決策樹，某個「高度懷疑之社區肺炎患者」所對應的一維陣列(可稱為症狀(symptom)陣列)，可能如圖4所示：檢驗條件NO (False)、臨床條件YES (True)、流行病學條件NO(False)，其他條件無須測試(NULL)。對應問診過程如圖5中的粗箭線所示。圖1中各個測試條件，及診斷的名稱，則用相同安排方式，分別顯示為圖6和圖7的陣列，方便檢視及應用。

[7] https://www.msn.com/zh-tw/news/living/%E6%8B%92%E7%B5%95%E6%BC%8F%E7%B6%B2%E4%B9%8B%E9%AD%9A%EF%BC%81%E6%8C%87%E6%8F%AE%E4%B8%AD%E5%BF%83%E6%93%B4%E5%A4%A7%E3%80%8C%E6%96%B0%E5%86%A0%E8%82%BA%E7%82%8E%E3%80%8D%E6%8E%A1%E6%AA%A2%E6%A8%99%E6%BA%96/ar-BB11ZXrx

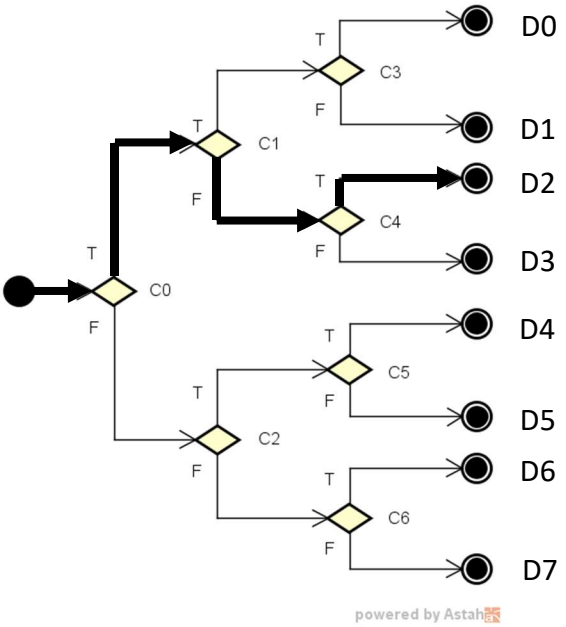

圖2. 完全二元樹，粗箭線代表某一個案的詢問及決策過程

0	1	2	3	4	5	6
T	F	N	N	T	N	N

圖3. 圖2二元樹的個案一例

0	1	2	3	4	5	6
NO	NULL	YES	NULL	NULL	NO	NULL

圖4. 圖1COVID-19決策樹的個案一例

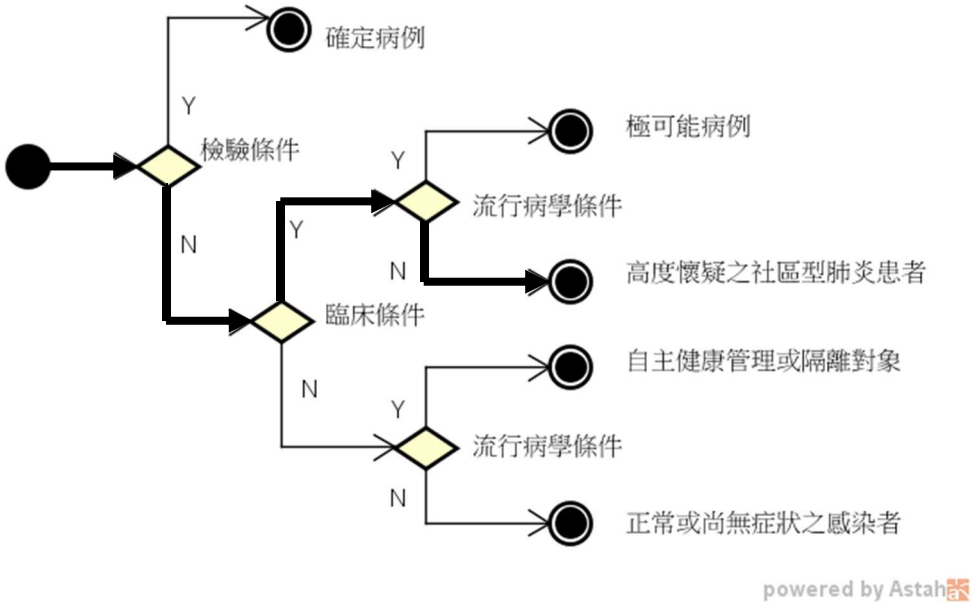

圖5. 圖4症狀陣列對應的問診過程

0	1	2	3	4	5	6
檢驗條件	NULL	臨床條件	NULL	NULL	流行病學條件	流行病學條件

圖6. 圖1決策樹對應的測試條件名稱陣列

0	1	2	3	4	5	6
NULL	確定病例	NULL	NULL	NULL	NULL	NULL

7	8	9	10	11	12	13
NULL	NULL	NULL	NULL	極可能病例	高度懷疑之社區肺炎患者	自主健康管理或隔離對象

14
正常或無症狀之感染者

圖7. 圖1決策樹對應的診斷名稱陣列

一旦得到如圖4的個案症狀陣列，資料結構的定理告訴我們：可以用如下(圖8)的演算法虛擬碼，依照每個測試條件，進行分類診斷:

```
/*
 * symptoms 是症狀陣列，如圖4
 * diagnoses 是診斷名稱陣列，如圖7
 */
1. index = 0
2. nLayers = 3
3. for iLayer = 0 to nLayers - 1
   {
3.1   index = NewIndex(index, symptoms)
3.2   if index >= 2^nLayers - 1 or symptoms[index] == Null,
        break
   }
4. diagnosis = diagnoses[index]
```

圖8. 圖1決策樹的分類演算法。函式 **NewIndex** 的演算法如圖9

函式 **NewIndex(index, symptoms)**

```
1. symptom = symptoms[index]
2. if symptom is YES
   {
     result = 2 * index + 1
   }
3. else if symptom is NO
   {
     result = 2*index + 2
   }
4. else
   {
     Should not be here, throw an exception
   }
5. return result
```

圖9. 圖8演算法中函式 **NewIndex** 的演算法

雖然此處圖8、圖9的演算法以圖1決策樹為對象，事實上它們可以應用到更廣泛的二元樹結構如圖2。

本題希望你參考以上說明，以圖1決策樹為依據，修改使用圖8演算法，寫一個C#程式，詢問使用者是否合乎測試條件，並產生診斷，顯示在螢幕，如圖10。

圖 10. 程式輸出螢幕畫面。

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%)

注意: 以下的程式碼片段可供參考利用(呼叫此處提供的函式，可不必在答案卷又宣告一次)。

```
static void InitializeConditions(out string[] conditions)
{
  conditions = new string[] {
    "檢驗條件",  // C0
    "",         // C1, Null string
    "臨床條件",  // C2
    "",
    "",
     "流行病學條件",  // C5
     "流行病學條件"   // C6
     };
     return;
}


static void InitializeDiagnoses(out string[] diagnoses)
{
  diagnoses = new string[] {
  "",
  "確定病例",  // D1
  "", "", "", "", "",
  "", "", "", "",
   "極可能病例",            // D11
   "高度懷疑之社區肺炎患者", // D12
   "自主健康管理或隔離對象", // D13
   "正常或無症狀之感染者"    // D14
   };
   return;
}
```