通識計算機程式設計期中考參考解答, 4/22/2011

1.

(a) 宣告 **int** 變數 **k**, **bool** 變數 **b**, **double** 變數 **x** (3%)

答：
```
int k;
bool b;
double x;
```

(b) 在螢幕顯示一行字，要求使用者輸入一個整數 (3%)

答：
```
Console.WriteLine("輸入一個整數");
```

(c) 自鍵盤讀入一個整數.，並將其值存入已宣告之 **int** 變數 **k** (3%)

答：
```
k = int.Parse(Console.ReadLine()); 或
k = Convert.ToInt16(Console.ReadLine());
```

(d) 令已宣告之 **bool** 變數 **b** 等於邏輯敘述： **k** 被 2 除之餘數等於 0 (3%)

答：
```
b = (k % 2 == 0);
```

(e) 若.**b** 爲真，在螢幕顯示 **”偶數“** 否則印出 **”奇數“** (3%)

答：
```
if (b)
{
    Console.WriteLine("偶數");
}
else
{
    Console.WriteLine("奇數");
}
```
或
```
string message = b ? "偶數" : "奇數";
Console.WriteLine(message);
```

2.

(a) 只用一個敘述，將已宣告設值之 **int** 變數 **n** 以遞減算子減 1 後，再設值 (assign) 給他處已宣告之 **int** 變數 **m** (3%)

答：
```
m = --n;
```

(b) 宣告 **bool** 變數 **r**，並設其值爲(非 **p**) ”或”(or) **q** 之邏輯演算的結果。

bool 變數 **P** 與 **q** 均已於他處已宣告設值 (3%)

答： `r = !p || q;`

(c) 宣告 **double** 變數 **y**，並令其值爲他處已宣告設值之 **double** 變數 **t** 與 2π之積的餘弦函數(cosine)值，注意此處π爲圓周率 (3%)

答：
```
double y = Math.Cos(2.0 * Math.PI * t);
```

(d) 宣告 **string** 變數 **state**，利用三元運算子使其在他處已宣告設值之 **double** 變數 **membranePotential** 大於等於**-55** 時等於**"激發"**，反之則等於**"靜態"** (3%)

答：
```
string state = (membranePotential >= -55) ?
    "激發" : "靜態";
```

(e) 宣告變數 **c** 爲 **char** 型別，並令其值爲倒斜線(back slash)字元 (3%)

答：
```
char c = '\\';
```

3.

(a) 宣告一個 **int** 常數 **N**，其值爲 7 (3%)

答：
```
const int N = 7;
```

(b) 宣告一個列數與行數均爲爲 **N** 的 **int** 二維陣列，命名爲 **a** (3%)

答：
```
int[,] a = new int[N, N];
```

(c) 宣告一個亂數產生器物件 **rand**，種子數爲 777 (3%)

答：
```
Random rand = new Random(777);
```

(d) 將每個 **a** 陣列元素設爲一個由 **rand** 產生，在 2(含)與 12(含)之間的亂數 (3%)

答：
```
    int i;
    int j;
    for(i=0; i<N; ++i)
    {
        for (j = 0; j < N; ++j)
        {
            a[i, j] = rand.Next() % 11 + 2;
        }
     }
```

(e) 計算 **a** 陣列各列的總和，存入一維 **int** 陣列 **rowSum** 的對應元素。令 **rowSum** 已於他處宣告 (3%)

答：
```
for (i = 0; i < N; ++i)
{
    rowSum[i] = 0;
    for (j = 0; j < N; ++j)
    {
        rowSum[i] += a[i, j];
    }
 }
```

4. .

(a) (3%)
```
int 1stPlace = 1;
```
答：變數名稱不可以數字開始

更正：
```
int FirstPlace = 1;
```

(b) (3%) 以下程式片段應產生三種輸出之一：
```
char age = Console.ReadLine().ToCharArray()[0];
switch (age)
{
    case '少': Console.WriteLine("紅燭昏羅帳");
    case '壯': Console.WriteLine("斷雁叫西風");
    default: Console.WriteLine("鬢已星星也");
 }
```
答： **case** 及 **default** 結束時須加上 **break** 敘述

更正：
```
char age = Console.ReadLine().ToCharArray()[0];
switch (age)
{
    case '少': Console.WriteLine("紅燭昏羅帳");
               break;
    case '壯': Console.WriteLine("斷雁叫西風");
               break;
    default: Console.WriteLine("鬢已星星也");
               break;
}
```

(c) (3%)

```
int i;
while( i <= 3 )
{
    Console.WriteLine( i/(i-1) );
    ++i;
}
```

答： `i = 1, i/(i-1)`會發生除以 `0` 的例外

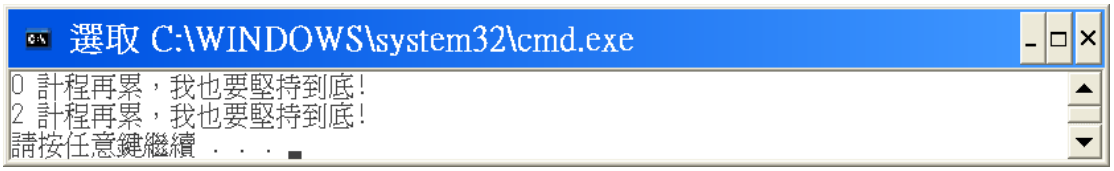
更正(合理的其他更正方式亦可)：

```
int i = 1;
while (i <= 3)
{
    ++i;
    Console.WriteLine(i / (i - 1));
}
```

(d) (3%)以下程式片段執行後，應出現如後畫面：

```
for( int i=0; i<3; ++i )
{
    Console.WriteLine(
        “{0} 計程再累，我也要堅持到底!”, i++);
}
```

答： 迴圈中控制變數 `i` 發生改變，使用到的 `i` 值只有 `0, 2` 兩個，使輸出爲



更正：

```
for (int i = 0; i < 3; ++i)
{
    Console.WriteLine(
        "{0} 計程再累，我也要堅持到底!", i+1);
}
```

(e) (3%) 下列程式片段應定義兩個函式.

```
static void f1(int x)
{
    Console.WriteLine(“f1({0})”, x);

    static void f2(int x)
    {
        Console.WriteLine(“f2({0})”, x);
    }
}
```

答： 函式 **f2** 不應定義在函式 **f1** 的一對大括弧之間.

更正：

```
static void f1(int x)
{
    Console.WriteLine("f1({0})", x);
}

static void f2(int x)
{
    Console.WriteLine("f2({0})", x);
}
```

5. 試寫出下列程式的輸出 (5%)

```
using System;

namespace MidTerm2011Problem5
{
   class Program
   {
      static void Main(string[] args)
      {

         int[] a = { 4, 5, 6 };
         int[] b = a;
         WhatDoesThisDo(a);
         Console.WriteLine("a[2] = {0}, b[2] = {1}",
            a[2], b[2]);
```

```
        }
        static void WhatDoesThisDo(int[] x)
        {
            x = new int[4] { 8, 9, 0, 1 };
            x[2] = 7;
        }
    }
}
```

答：

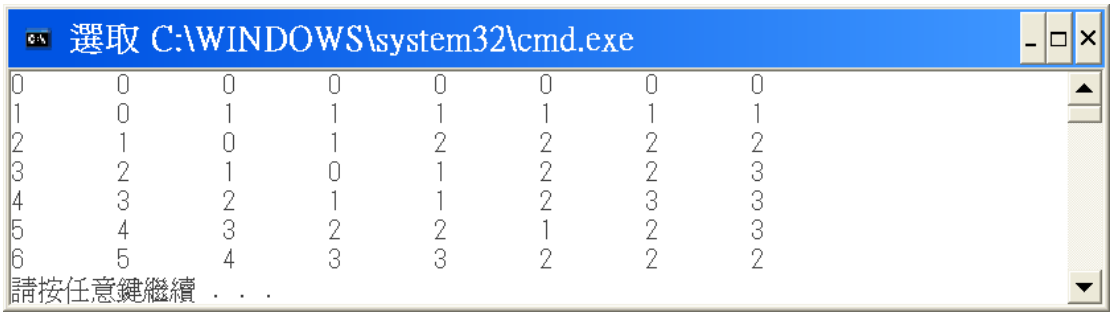
6. 試寫出下列程式的輸出 (10%)

```
using System;

namespace MidTerm2011Problem6
{
    class Program
    {
        static void Main(string[] args)
        {
            string pattern = "survey";
            char[] p = pattern.ToCharArray();
            string target = "surgery";
            char[] t = target.ToCharArray();
            int[,] c = DP(p, t);
            int n = t.Length;
            int m = p.Length;
            int i;
            int j;
            for (i = 0; i <= m; ++i)
            {
                for (j = 0; j <= n; ++j)
                {
                    Console.Write(c[i, j] + "\t");
                }
                Console.WriteLine();
```

```
            }
        }
        static int[,] DP(char[] p, char[] t)
        {
            int m = p.Length;
            int n = t.Length;
            int[,] c = new int[m+1, n+1];
            int i;
            int j;
            for (j = 0; j <= n; ++j)
            {
                c[0, j] = 0;
            }
            for (i = 0; i <= m; ++i)
            {
                c[i, 0] = i;
            }
            for (i = 1; i <= m; ++i)
            {
                for (j = 1; j <= n; ++j)
                {
                    c[i, j] = (p[i-1] == t[j-1]) ?
                        c[i - 1, j - 1] :
                        1 +
                        Minimum(c[i - 1, j], c[i, j - 1],
                                c[i - 1, j - 1]);
                }
            }
            return c;
        }
        static int Minimum(int x, int y, int z)
        {
            if (x <= y && x <= z) return x;
            if (y <= x && y <= z) return y;
            return z;
        }
    }
}
```

答：



7.

```
/*
 * MidTerm2011Problem7
 * skj, 4/21/2011
 * 尋找英文文章中的Semantic Words
 * 並計算其出現次數, 由高而低顯示
 *
 * 主程式虛擬碼
 * 1. 輸入文章中的文句, 以行爲單位存成一維陣列
 * 2. 假設儲存semantic words的陣列
 *    是semanticWord
 *    對應的出現次數是陣列freq
 * 3. 取得semanticWord, freq
 * 4. 將semanticWord及freq依freq由高而低排序
 * 5. 輸出semanticWord及freq
 *
 * 函式虛擬碼 ProcessArticle(文章, semanticWord, freq)
 * 1. 假定放入semanticWord及freq的單字數爲nSW
 *    令其初值爲 0
 * 2. for 文章每一行文句
 *    {
 * 2.1  取出單字構成陣列word
 * 2.2  for 所有word中的單字
 *      {
 * 2.2.1   if 單字是stop words, continue
 * 2.2.2   if 單字不在semanticWord中
 *         {
 * 2.2.2.1      將單字新增入semanticWord
 * 2.2.2.2      遞增nSW
```

```
 *        }
 * 2.2.3   遞增對應的freq元素
 *      }
 *    }
 *
 * 測試規劃: Robert Frost的詩The Road Not Taken之最後一段:
 *
 * I shall be telling this with a sigh
 * Somewhere ages and ages hence:
 * Two roads diverged in a wood, and I—
 * I took the one less traveled by,
 * And that has made all the difference
 *
 * 應輸出(括弧內代表出現次數, 由高而低排列,
 * 同次數單字出現序未必與此處相同):
 * ages (2)
 * telling (1)
 * sigh (1)
 * roads (1)
 * diverged (1)
 * wood (1)
 * less (1)
 * traveled (1)
 * difference (1)
 */

using System;
using System.Diagnostics;

namespace MidTerm2011Problem7
{
    class Program
    {
        static void Main(string[] args)
        {
            Debug.Assert(Test_1_OK());

            string[] semanticWord;
```

```
    int[] freq;
    int iSW;
    string[] line = InputArticle();
    ProcessArticle(line, out semanticWord, out freq);
    Array.Sort(freq, semanticWord);
    Array.Reverse(semanticWord);
    Array.Reverse(freq);
    Console.WriteLine(
        "\n上列文章的Semantic words及出現次數");
    for (iSW = 0; iSW < semanticWord.Length; ++iSW)
    {
        Console.WriteLine("{0} ({1})",
            semanticWord[iSW], freq[iSW]);
    }
}

static bool Test_1_OK()
{
    string[] semanticWord;
    int[] freq;
    string[] line = new string[5]
       {"I shall be telling this with a sigh",
        "Somewhere ages and ages hence:",
        "Two roads diverged in a wood, and I-",
        "I took the one less traveled by,",
        "And that has made all the difference" };
    ProcessArticle(line, out semanticWord, out freq);
    string[] correctSW = {
        "ages", "telling", "sigh", "roads",
        "diverged", "wood", "less", "traveled",
        "difference" };
    int[] correctFreq = { 2, 1, 1, 1,
                          1, 1, 1, 1,
                          1 };
    int iSW;
    int idx;
    bool result = (semanticWord.Length == 9 &&
                   freq.Length == 9);
```

```
    if (result)
    {
        for (iSW = 0; iSW < semanticWord.Length; ++iSW)
        {
            idx = Array.IndexOf(
                semanticWord, correctSW[iSW]);
            result = result &&
            (idx >= 0 && freq[idx] == correctFreq[iSW]);
        }
    }
    return result;
}

static void ProcessArticle(
    string[] line, out string[] semanticWord,
        out int[] freq)
{
    const int MAX_N_SW = 100;
           // 假設之semantic words字數上限

    string[] sw = new string[MAX_N_SW];
           // semantic words
           // 的local 暫時版本

    int[] fr = new int[MAX_N_SW];
           // freq的local 暫時版本

    int iSW;
    for (iSW = 0; iSW < MAX_N_SW; ++iSW)
    {
        sw[iSW] = null;
        fr[iSW] = 0;
    }
    int nLines = line.Length;
    int iL;
    string[] word;
    int nW;
    int iW;
```

```
    int nSW = 0;
    for (iL = 0; iL < nLines; ++iL)
    {
        word = ProcessLine(line[iL]);
        nW = word.Length;
        for (iW = 0; iW < nW; ++iW)
        {
            if (IsAStopWord(word[iW])) continue;
            iSW = Array.IndexOf(sw, word[iW]);
            if (iSW < 0)
            {
                iSW = nSW;
                sw[iSW] = word[iW];
                ++nSW;
            }
            fr[iSW]++;
        }
    }

    // 產生要回傳的版本
    semanticWord = new string[nSW];
    freq = new int[nSW];
    Array.Copy(sw, semanticWord, nSW);
    Array.Copy(fr, freq, nSW);
}

static string[] InputArticle()
{
    Console.Write("輸入行數: ");
    int nLines = int.Parse(Console.ReadLine());
    string[] line = new string[nLines];
    int iL;
    for (iL = 0; iL < nLines; ++iL)
    {
        Console.WriteLine("輸入第{0}行", iL+1);
        line[iL] = Console.ReadLine();
    }
    return line;
```

```
}

static string[] ProcessLine(string line)
{
    char[] separator =
      { ' ', ',', '.', ':', ';',
        '\'','\\', '"', '/', '`',
        '-', '|', '(', ')', '[',
        ']', '{', '}', '!', '@',
        '#', '$', '%', '^', '&',
        '*', '+', '_', '=', '~',
        '<', '>', '?'
      };
    string[] word = line.Split(separator);

    // 將單字改爲全小寫, 以便比對stop words
    int iW;
    for (iW = 0; iW < word.Length; ++iW)
    {
        word[iW] = word[iW].ToLower();
    }
    return word;
}

static bool IsAStopWord(string word)
{
    // 不算完全, 但可接受的stop words列表
    string[] stopword =
    {
       "","a","able","about","across",
       "after","all","almost","also","am",
       "among","an","and","any","are",
       "as","at","be","because","been",
       "but","by","can","cannot","could",
       "dear","did","do","does","either",
       "else","ever","every","for","from",
       "get","gets", "got","had","has",
       "have","he","hence","her","hers",
```

```
            "him","his","how","however","i",
            "if","in","into","is","it",
            "its","just","least","let","lets",
            "likely","made","make", "makes", "may",
            "me","might","most","must","my",
            "neither","no","nor","not","of",
            "off","often","on","one","only",
            "or","other","our","own","rather",
            "said","say","says","shall","she",
            "should","since","so","some","somewhere",
            "take","taken","takes","than","that",
            "the","their","them","then","there",
            "these","they","this","tis","to",
            "too","took", "twas","two","us",
            "want","wants","was","we","were",
            "what","when","where","which","while",
            "who","whom","why","will","with",
            "would","yet","you","your"
        };
        bool result = (
            Array.IndexOf(stopword, word) >= 0 );
        return result;
    }
  }
}
```