通識計算機程式設計期中考參考解答

4/27/2019

1.

(a) (3%)

```
bool isValid;
int r;
byte rByte;
```

(b) (3%)

```
Console.WriteLine("輸入0至255的正整數");
```

(c) (3%)

```
r = int.Parse(Console.ReadLine());
```

或

```
r = Convert.ToInt32(Console.ReadLine());
```

(d) (3%)

```
isValid = (r >= 0) && (r < 256);
```

(e) (3%)

```
if(isValid)
{
    rByte = (byte) r;
}
else
{
    Console.WriteLine("r不是合格的紅色成分數值");
}
```

2.

(a) (3%)

```
sum += term;
```

(b) (3%)

```
nWeeks = nDays / 7;
```

(c) (3%)

```
double y = x / Math.Sqrt(1.0 + x*x);
```

(d) (3%)

```
ds = (delta >= 0.0)? "實根" : "複根";
```

(e) (3%)

```
char ch = '\\';
```

3.

(a) (3%)

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

(b) (3%)

```
double[] x1 = {0.1, 0.3, 0.5};
double[] x2 = {0.2, 0.4, 0.6};
double[] w = {1.0, 0.0, -1.0};
```

(c) (3 %)

```
int n = w.Length;
```

或

```
int n = w.GetUpperBound(0) + 1;
```

(d) (3%)

```
double y = 0.2;
for(int i = 0; i < n; ++i)
{
    int rn = rand.Next();
    if(rn % 2 == 0)
    {
        y += w[i]*x1[i];
    }
    else
    {
       y += w[i]*x2[i];
    }
}
double output = ElliotSig(y);
```

(e) (3%)

```
static double ElliotSig(double y)
{
    double abs_y = Math.Abs(y);
    double result = y / (1.0 + abs_y);
```

```
    return result;
}
```

4. 找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%) (一個語法錯誤)

a == b 得到 bool 的值(true 或 false)，

而 c 仍是 int 值，所以 a == b 的結果，不能再與 c 比較
而應改為

```
if( a == b && b == c )
{
    Console.WriteLine("I is an identity matrix");
}
else
{
    Console.WriteLine(
     "Diagonal matrix I is not an identity matrix");
}
```

(b) (3%) (一個語義錯誤)

while(i <= 4)迴圈中的 x[i]會用到 x[4]，但是陣列 x 的長度為 4，所以只允許使用 x[0] 到 x[3]。

```
int[] x = {0, 1, 2, 3};
int i = 0;
while(i < 4)
{
    Console.WriteLine("x[" + i + "]=" + x[i]);
    ++i;
}
```

(c) (3%) (一個語義錯誤)

for 迴圈每執行一次，本金 **p0** 就乘以 **1 - loss_rate = 0.9** 一次。由於 **nYears** 的上限僅到 **3**，迴圈僅會被執行三次，本金最多只會被減到 **0.9 x 0.9 x 0.9 = 0.729** 倍，不到 **0.5** 倍。所以 **nYears** 上限必須提高。選擇一個大於 **7** 的上限($0.9^7 \approx 0.43$)，就沒有問題了。

```
double p0 = 10000.0;
double loss_rate = 0.1;
double p = p0;
int nYears;
for(nYears = 1; nYears < 40; ++nYears)
{
    p = p*(1.0 - loss_rate);
    if(p < 0.5*p0) break;
}
Console.WriteLine(
    "The principal is halved after {0} years",
    nYears);
```

(d) (3%) (一個語義錯誤)

二維陣列的第一個 index 對應表格的列, 第二個 index 對應表格的行。所以 **i** 的範圍應該為 **0, 1**，**j** 的範圍則為 **0, 1, 2**。

```
int[,] m = { {1, 3, 5}, {2, 4, 6}};
for(int i = 0; i < 2; ++i)
{
   for(int j = 0; j < 3; ++j)
   {
      Console.Write(
         "m[" + i + "," + j + "] = " + m[i, j] + "\t");
   }
   Console.WriteLine();
}
```

(e) (3%) (一個語義錯誤)

函式 **Transform** 內，將 **vector_a** 改為一個新的一維陣列，長度為 **4**。在 heap space 中的位置也改變了。由於是傳址呼叫，主程式中的 **vector_a** 也跟著變成長度為 **4** 的新陣列。雖然前三個元素與原來相同，但是 **Debug.Assert(vector_a.Length == 3);** 內的條件不再成立，在顯示題目所附的主控台畫面之前，就會跳離主程式。如果改用傳值呼叫，**Transform** 內對 **vector_a** 的變動，不會影響主程式中的 **vector_a**，就沒有問題了。

```
static void Main(string[] args)
{
    int[] vector_a = {1, 2, 3};
    int[] vector_c = Transform(ref vector_a);
```

```
        Debug.Assert(vector_a.Length == 3);
        Console.WriteLine(
      "Transform of [{0}, {1}, {2}] is [{3}, {4}, {5}]",
            vector_a[0], vector_a[1], vector_a[2],
            vector_c[0], vector_c[1], vector_c[2]);
}

static int[] Transform(ref int[] vector_a)
{
    vector_a = new int[] {
        vector_a[0], vector_a[1], vector_a[2], 1};
    int[,] t = { {0, 1, 0, -1},
                 {1, 0, 0, 1},
                 {0, 0, 1, 0},
                 {0, 0, 0, 1}};
    int[] vector_c = {0, 0, 0, 1};
    for(int i = 0; i < 3; ++i)
    {
        vector_c[i] = 0;
        for(int j = 0; j < 3; ++j)
        {
            vector_c[i] += t[i, j]*vector_a[j];
        }
    }
    double den = (double)(vector_c[3]);
    int[] result = new int[] {
        (int)(vector_c[0]/den),
        (int)(vector_c[1]/den),
        (int)(vector_c[2]/den)};
    return result;
}
```

5. (5 %)

```
選取 C:\Program Files\dotnet\dotn...
P( 0, 0.2 ) = 1
P( 1, 0.2 ) = 0.2
P( 2, 0.2 ) = -0.6
P( 3, 0.2 ) = -0.36
P( 4, 0.2 ) = 0.3504

按 enter/return 鍵結束
```

6. (10 %)

```
選取 C:\Program Files\dotnet\dotnet.exe

Begin Simplified LSTM cell IO demo

Creating an n=2 input, m=3 state LSTM cell
Sending input = (1.0, 2.0) to LSTM

Output is:
0.0640
0.0000
0.0080

New cell state is:
0.1600
0.0000
0.0400

End LSTM demo

按 enter/return 鍵結束
```

7.

```
/*
 * This program, Seliza, is a simplified version of ELIZA,
 * and Eliza is arguably the earliest chatbot as a psychotherapist.
 * For details, see the Wikipedia site with the keyword "Eliza"
 * Seliza is a modification of an implementation of Eliza
 * for Chinese using Java by 陳鍾誠 教授 (金門大學) in
 * http://ccckmit.wikidot.com/code:eliza
 *
```

```
* The basic idea of ELIZA is to apply some rules for manipulating
* words in user's input to generate a "believable" answer.
* Eliza does not actually understand the user's input,
* it simply searches for some key words in the user's input,
* and find a response from a database according some simple rules.
*
* In this program, we assume that the rule base consists of one
* 1-d array "key_Words" and one Nk X Nr table "responses,"
* where Nk and Nr are number of keywords in array keyWords and
* the maximum number of choices for responses in the whole rule base.
*
* The initial pseudo code of this program is given below,
* with arrays "key_words" and "responses" ready:
*
* Pseudo code for the main program of Seliza v0.1
* 1. 取得一行使用者鍵盤輸入, 令為 input
* 2. while(input 不是 "再見")
*    {
* 2.1   檢查並處理一般key Word 和responses
* 2.2    在螢幕上顯示 answer
* 2.3    取得一行使用者鍵盤輸入, 令為 input
*      }
* 3. 結束程式
*
* * Pseudo code for 檢查並處理一般key word 和 responses
* 1.  檢查keyWords 中的元素是否在input 中出現?
* 2.  if(出現在第i 個位置)
*     {
* 2.1   產生回應 answer
*     }
*     else
*     {
* 2.2   產生比對失敗時的回應 answer
*     }
* 3.  return answer
*
*
*Pseudo code for 檢查keyWords 中的元素是否在input 中出現?
```

```
* 1. for each keyWord in keyWords,
*    {
* 1.1  i = String.IndexOf(input, keyword)
* 1.2  if(i >= 0) break
*    }
*    return i
*
*Pseudo code for 產生回應 answer, v0.1
*1. 產生 0 到 Nr 之間的亂數 j
*2. return response[i, j]
*
*The next step is to deal with responses with template.
*For example, if the user says "我覺得計程的考試很難,"
*A possible answer is "為什麼計程的考試很難?"
*Assume that key words with templated response are different
*from the original key words, we can check the key words
*related to templated responses first.
*Again, these keywords and templeted responses are
*stored as a 1-d array keywordsRelatedToTemplatedResponses and
*a 2-d Nkt X Nrt array templatedResponses. This is set to be version
0.2:
*
*Pseudo code for the main program of Seliza v0.2
* 1. 取得一行使用者鍵盤輸入, 令為 input
* 2. while(input 不是 "再見")
*    {
* 2.1   檢查並處理 templated responses 的 key word, 取得 answer
* 2.2   if answer 是 "",
*       {
* 2.2.1    檢查並處理一般 key word 和 responses, 取得 answer
*        }
* 2.3   在螢幕上顯示 answer
* 2.4   取得一行使用者鍵盤輸入, 令為 input
*      }
* 3. 結束程式
*
*
* Pseudocode for 檢查並處理 templated responses 的 key word
```

```
 * 1. answer = null
 * 2. 檢查templated responses的key word是否在input中出現?
 * 3. if(出現在第i個位置)
 *    {
 * 3.1   tail = input 中 keyword 之後的子字串
 * 3.2   answer = keyword + tail
 *    }
 * 4. return answer
 *
 * Another case is that Seliza should exchange the positions of "你" and
"我".
 * For example, if the input is "我覺得你很聰明," and the response may be
 * "你為什麼覺得我很聰明?"
 * Thus, include this rule to the program, and the pseudo code become
v0.3:
 *
 * Pseudo code for the main program of Seliza v0.3
 * 1. 取得一行使用者鍵盤輸入, 令為 input
 * 2. while(input 不是 "再見")
 *    {
 * 2.1    檢查並處理templated responses的key word, 取得answer
 * 2.2   if answer是 ""
 *       {
 * 2.2.1    檢查並處理一般key word 和 responses, 取得 answer
 *        }
 * 2.3   在螢幕上顯示 answer
 * 2.4   取得一行使用者鍵盤輸入, 令為 input
 *      }
 * 3. 結束程式
 *
 * Pseudocode for 檢查並處理templated responses的key word
 * 1. answer = null
 * 2. 檢查templated responses的key word是否在input中出現?
 * 3. if(出現在第i個位置)
 *    {
 * 3.1   tail = input 中 keyword 之後的子字串
 * 3.2   複製 tail 到字元陣列 buffer
 * 3.3   找出 tail 中的'你', 把buffer的對應位置換成'我'
```

```
* 3.4   找出 tail 中的'我', 把buffer 的對應位置換成'你'
* 3.5   將buffer 轉為字串 tail
* 3.6   answer = keyword + tail
*    }
* 4. return answer
*
* Pseudocode for 找出 tail 中的'你', 把buffer 的對應位置換成'我'
* 1. 找出第一個'你'在tail 的位置 idx
* 2. while(idx 為有效位置)
*    {
* 2.1   buffer[idx] = '我';
* 2.2   找下一個'你'在tail 的位置 idx
*    }
*
* Pseudocode for 找出 tail 中的'我', 把buffer 的對應位置換成'你'
* 1. 找出第一個'我'在tail 的位置 idx
* 2. while(idx 為有效位置)
*    {
* 2.1   buffer[idx] = '你';
* 2.2   找下一個'我'在tail 的位置 idx
*    }
*/
```

```
using System;
using System.Threading;

namespace Problem7
{
    class Program
    {
        static int IdxOfKeyWordsInInput(string input,
            string[] key_words)
        {
            int idx = 0;
            int foundAt = -1;
            for(idx = 0; idx < key_words.Length; ++idx)
            {
                foundAt = input.IndexOf(key_words[idx]);
                if(foundAt >= 0) break;
            }
            if(foundAt < 0) idx = -1;
            return idx;
        }
```

```
static string ResponseForInput(int idx, string[,] responses,
    Random rand)
{
    int idx_j = rand.Next() % (responses.GetUpperBound(1) + 1);
    string answer = responses[idx, idx_j];
    return answer;
}
static string ResponseDueToGeneralKeyWords(
    string input,
    string[] key_words,
    string[,] responses,
    Random rand)
{
    string answer = "";
    int matchedAtIdx = IdxOfKeyWordsInInput(input, key_words);
    if(matchedAtIdx >= 0)
    {
        answer = ResponseForInput(
                      matchedAtIdx, responses, rand);
    }
    return answer;
}

static string ResponseDueToKeyWordsRelatedToTemplates(
    string input,
    string[] key_words_for_templated_responses,
    string[,] responses_with_template,
    Random rand
)
{
    string answer = "";
    string key_word = "";
    int idx = -1;
    string tail = "";
    int matchedAtIdx = IdxOfKeyWordsInInput(
        input, key_words_for_templated_responses);
    if(matchedAtIdx >= 0)
    {
        answer = ResponseForInput(
            matchedAtIdx,
            responses_with_template, rand);
        key_word =
             key_words_for_templated_responses[matchedAtIdx];
        idx = input.IndexOf(key_word);
        tail = input.Substring(idx + key_word.Length);

        // V0.3
        if(tail.Length != 0)
        {
            char[] buffer = tail.ToCharArray();
            SwitchYouToI(tail, buffer);
            SwitchIToYou(tail, buffer);
            tail = new string(buffer);
        }
    }

    return answer + tail;
}

static void SwitchYouToI(string tail, char[] buffer)
{
```

```
    int idx = tail.IndexOf('你');;
    while(idx >= 0 && idx < tail.Length)
    {
        buffer[idx] = '我';

        // check if has arrived at the end of input
        if(idx >= tail.Length - 1) break;

        idx = tail.IndexOf('你', idx+1);
    }
}

static void SwitchIToYou(string tail, char[] buffer)
{
    int idx = tail.IndexOf('我');
    while(idx >= 0 && idx < tail.Length)
    {
        buffer[idx] = '你';

        // check if has arrived at the end of input
        if(idx >= tail.Length - 1) break;

        idx = tail.IndexOf('我', idx+1);
    }
}

static void Main(string[] args)
{
    string[] key_words = {
        "謝謝",
        "對不起",
        "抱歉",
        "不好意思",
        "我是",
        "甚麼",
        "什麼",
        "何時",
        "誰",
        "哪裡",
        "如何",
        "為何",
        "原因",
        "理由",
        "你好",
        "嗨",
        "或許",
        "不曉得",
        "不知道",
        "總是",
        "常常",
        "像",
        "對",
        "朋友",
        "電腦",
        "難過",
        "高興"
```

```
};

string[,] responses =
{
    {"不客氣!", "不客氣!", "不客氣!"},
    {"沒問題", "不會", "沒問題"},
    {"沒問題", "不會", "沒問題"},
    {"沒問題", "不會", "沒問題"},
    {"你好, 久仰, 久仰", "你好, 我是Seliza", "你好"},
    {"你的答案是甚麼?", "為甚麼你對這問題有興趣?",
     "何不問問別人?"},
    {"你的答案是甚麼?", "為甚麼你對這問題有興趣?",
     "何不問問別人?"},
    {"你認為是甚麼時間?", "為甚麼你對這問題有興趣?",
     "何不問問別人?"},
    {"你認為是誰?", "為甚麼你對這問題有興趣?", "何不問問別人?"},
    {"你認為是哪裡?", "為甚麼你對這問題有興趣?", "何不問問別人?"},
    {"你的答案是甚麼?", "為甚麼你對這問題有興趣?",
     "何不問問別人?"},
    {"你認為是甚麼原因?", "為甚麼你對這問題有興趣?",
     "何不問問別人?"},
    {"真的?", "有其他原因嗎?", "真的?"},
    {"這說明了甚麼?", "有其他理由嗎?", "真的?"},
    {"你好, 近來可好?", "你好, 高興看到你", "你好."},
    {"嗨, 近來可好?", "嗨, 高興看到你", "你好."},
    {"你好像不太確定?", "或許吧.", "或許吧."},
    {"訊息不夠嗎?", "可能要問專家吧", "資訊不夠嗎?"},
    {"資訊不夠嗎?", "可能可以問問內行的人吧", "訊息不夠嗎?"},
    {"可以說得具體一點嗎?", "甚麼時候?", "甚麼時候?"},
    {"舉個例子.", "說說看, 甚麼時候?", "甚麼時候?"},
    {"有多像?", "哪裡最像?", "是嗎?"},
    {"確定?", "了解", "噢!"},
    {"多告訴我一些他的事", "你們認識多久?", "交情好嗎?"},
    {"你說的電腦是我嗎?", "你說的電腦是我嗎?",
     "你說的電腦是我嗎?"},
    {"別想那麼多", "別難過, 事情總會解決",
     "有甚麼我可以幫忙的嗎?"},
    {"不錯", "好極了", "真棒"}
};

string[] key_words_for_templated_responses =
{
    "可不可以",
    "我想",
    "你是",
    "覺得",
    "認為",
    "以為",
    "不能",
    "不想",
    "不希望",
    "不",
    "請",
    "你"
```

```
};

string[,] responses_with_template =
{
    {"你確定要", "你確定要", "你確定要"},
    {"你確定想", "你確定想", "你確定想"},
    {"你認為我是", "你認為我是", "你認為我是"},
    {"為什麼你覺得", "是嗎? 為什麼", "為什麼" },
    {"為什麼你認為", "是嗎? 為什麼", "為什麼" },
    {"為什麼你以為", "是嗎? 為什麼", "為什麼" },
    {"為什麼不能", "試一下, 或許就能", "再試一下"},
    {"為什麼不想", "試一下, 或許就能", "再試一下"},
    {"為什麼不希望", "試一下, 或許就能", "再試一下"},
    {"為什麼不", "為什麼不", "為什麼不"},
    {"我該怎樣", "是否想要我", "為什麼要我"},
    {"為什麼你這麼關心我", "你自己", "為什麼你這麼關心我"}
};

string[] responses_without_matched_keywords =
{
    "我了解",
    "還有問題嗎 ?",
    "請繼續說",
    "可以說的更詳細一點嗎?",
    "這樣喔! 我知道!",
    "然後呢? 發生甚麼事?",
    "再來呢? 可以多說一些嗎?",
    "接下來呢? ",
    "然後呢?",
    "多談談你自己，好嗎?",
    "祝福你"
};

Console.WriteLine("Seliza: Hi, 請問如何稱呼您?");
string user_name = Console.ReadLine();
Console.WriteLine(
     "Seliza: 您好, 請逐行按鍵輸入對話, 要結束就說\"再見\"");

Console.Write(user_name+": ");
string input = Console.ReadLine();

string answer = "";
Random rand = new Random();

/*
// V0.1
while(input != "再見")
{
    answer = ResponseDueToGeneralKeyWords(
        input,
        key_words, responses,
        rand
        );

    if(answer == "")
    {
        int idx = rand.Next() %
```

```
                    responses_without_matched_keywords.Length;
                answer = responses_without_matched_keywords[idx];
            }

            Thread.Sleep(2000); // to simulate time for human
                                // typing
            Console.WriteLine("Seliza: " + answer);
            Console.Write(user_name + ": ");
            input = Console.ReadLine();
        }
        */


        // V0.2 and V0.3
        while(input != "再見")
        {
            answer = ResponseDueToKeyWordsRelatedToTemplates(
                input,
                key_words_for_templated_responses,
                responses_with_template,
                rand
            );
            if(answer == "")
            {
                answer = ResponseDueToGeneralKeyWords(
                input,
                key_words, responses,
                rand
                );
            }
            if(answer == "")
            {
                int idx = rand.Next() %
                    responses_without_matched_keywords.Length;
                answer = responses_without_matched_keywords[idx];
            }

            Thread.Sleep(2000); // to simulate time for human
                                // typing
            Console.WriteLine("Seliza: " + answer);
            Console.Write(user_name + ": ");
            input = Console.ReadLine();
        }

        Thread.Sleep(2000); // to simulate time for human typing
        Console.WriteLine("Seliza: 再見!");


        // ending the program
         Console.WriteLine();
        Console.WriteLine("按 enter/return 鍵結束");
        Console.ReadLine();
    }
  }
}
```