

通識計算機程式設計期末考

6/21/2019

試題共 7 題，兩面印製 18 頁，滿分 103

1. 本題假定要模仿機器人演唱台大校歌最後兩小節的歌聲，曲譜如圖1所示。程式概念的 UML 類別圖參見圖2。程式的主控台螢幕輸出如圖3。
 - (a) 撰寫結構(struct) **MusicNote**，宣告的成員變數為整數的 **pitch** 和 **length**。其中 **pitch** 代表音高，以樂器自動演奏控制程式檔案格式 midi 中，音高的整數表示為準。表 1 為音樂音高(固定唱名)與 midi 音高的對應。由表 1 可知台大校歌最後兩小節的各音符音高 **pitch** 為 74、77、70、72、70。成員變數 **length** 則代表音符長度，以 1/32 音符音長的倍數表示。所以我們的五個音符長度分為 32、48、16、32、96。此外還要加上一個建構式，設定 **pitch** 及 **length** 之值 (6%)
 - (b) 宣告類別 **Voice** 的成員變數：**note** (音符) 及 **lyric** (歌詞單字)。(3%)
 - (c) 實作 **Voice** 的建構式。(3%)
 - (d) 參考圖 3，實作 **Voice** 中的函式 **Sing**，在螢幕印出單一 **Voice** 物件的音高、音長、及對應歌詞單字。這模擬機器人唱出的單一聲音。(6%)
 - (e) 寫類別 **Program** 中的主程式 **Main**，建立 **MusicNote** 及字元陣列，分別命名為 **notes** 及 **lyrics**。依照前述說明，設定這兩個陣列的內容。(6%)
 - (f) 完成類別 **Program** 之主程式 **Main**。用一個 **for** 迴圈，建立五個 **Voice** 物件，呼叫其 **Sing** 函式，印出每一字歌詞及對應音符。最後的螢幕輸出如圖 3。(6%)

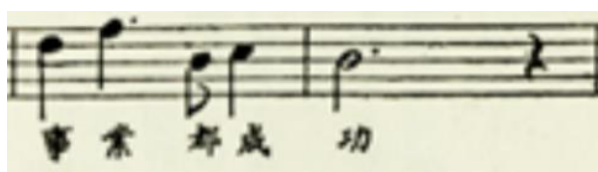


圖 1. 台大校歌最後兩小節的曲譜，注意第三、五個音符要降半音。

截自台大圖書館特藏組 <https://www.ntu.edu.tw/about/song.html>

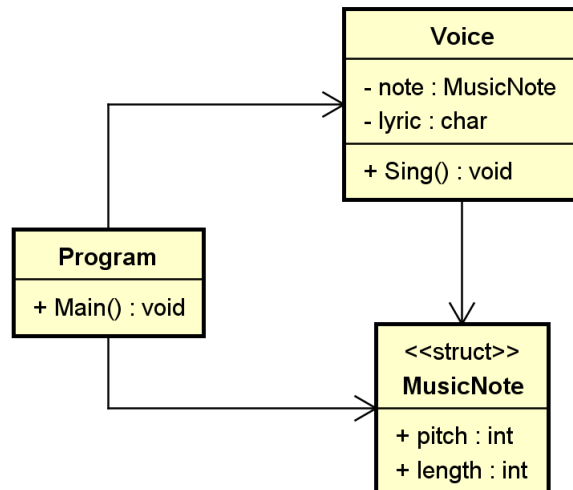


圖 2. 第 1 題概念對應的 UML 類別圖

```

選取 C:\Program Files\dotn...
pitch = 74, length = 32, lyric = 事
pitch = 77, length = 48, lyric = 業
pitch = 70, length = 16, lyric = 都
pitch = 72, length = 32, lyric = 成
pitch = 70, length = 96, lyric = 功
按 enter/return 鍵結束
  
```

圖 3. 第 1 題程式碼，執行時的主控台螢幕畫面

表 1/圖 4. 由 C4(中央 C)起兩個八度音高的對應 midi 音高編號

音樂 音高	C4	C#4/ Db4	D4	D#4/ Eb4	E4	F4	F#4/ Gb4	G4	G#4/ Ab4	A4	A#4/ Bb4	B4
Midi 編號	60	61	62	63	64	65	66	67	68	69	70	71
音樂 音高	C5	C#5/ Db5	D5	D#5/ Eb5	E5	F5	F#5/ Gb5	G5	G#5/ Ab5	A5	A#5/ Bb5	B5
Midi 編號	72	73	74	75	76	77	78	79	80	81	82	83

2. 找出以下程式片段之錯誤，並予更正.

(a) (3%) 一個錯誤 (以 **Debug.Assert** 敘述的要求為準)

```
class Circle {
    private double radius = 0;
    public Circle() {}
    public Circle(double radius) {
        this.radius = radius;
    }
    public double Area(double radius) {
        return Math.PI*radius*radius;
    }
    public double Perimeter() {
        return 2.0*Math.PI*radius;
    }
}

class Program {
    static void Main(string[] args) {
        Circle c = new Circle();
        Console.WriteLine("area of c = " + c.Area(1.0));
        double p = c.Perimeter();
        Debug.Assert(Math.Abs(p-2.0*Math.PI) < 1.0e-8);
    }
}
```

(b) (3%) 一個錯誤 (以 **Debug.Assert** 敘述的要求為準)

```
struct Student
{
    string name;
    string registerNumber;
    public Student(string na, string rn)
    {
        name = na;
        registerNumber = rn;
    }
}

class Program {
```

```

static void Main(string[] args) {
    Student b = new Student("sk", "B645331");
    Debug.Assert(b.name == "sk");
}

```

(c) (3%) 一個錯誤。

```

class Test {
    double c;
    public Test(double c) {
        this.c = c;
    }
    public double Get_c() {
        return c;
    }
    public int Get_c() {
        return (int) c;
    }
}

```

(d) (3%) 一種錯誤 (以 **Debug.Assert** 敘述的要求為準)

```

class STest {
    int d;

    // number of STest objects constructed
    static int nGenerated = 0;
    public STest() {
        d = 0;
        ++nGenerated;
    }
    public STest(int i) {
        d = i;
        ++nGenerated;
    }
    public int NGenerated() {
        return nGenerated;
    }
}

```

```

class Program {
    static void Main(string[] args) {
        STest ds1 = new STest();
        STest ds2 = new STest(3);
        Assert(STest.NGenerated() == 2);
    }
}

```

(e) (3%) 一種錯誤 (以 **Debug.Assert** 敘述的要求為準)

```

class TestE {
    private int e;
    public TestE() {
        e = 0;
    }
    public TestE(int a) {
        e = a;
    }
    public int E {
        set { e = value; }
        get { return e; }
    }
}

```

```

class Program {
    static void Main(string[] args) {
        TestE te1 = new TestE(2);
        TestE te2 = te1;
        te2.E = 4;
        Debug.Assert(te1.E == 2);
    }
}

```

3. 試寫出下列程式的輸出 (12%)

```

// Program for a cellular automata
// Reference:

```

```

// The Coding Train,
// 7.2: Wolfram Elementary Cellular Automata - The Nature of Code
// https://www.youtube.com/watch?v=W1zKu3fDQR8

using System;

namespace Problem3 {
    class Program {
        static void Main(string[] args) {
            const int N_CELLS = 10;
            const int SPIKE_IDX = 5;
            CA_90 ca_90 = new CA_90(N_CELLS, SPIKE_IDX);
            int[] cells = new int[N_CELLS];
            for(int n = 0; n < N_CELLS; ++n) {
                cells = ca_90.Cells();
                Print(cells);
                ca_90.GenerateNext();
            }

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

        static void Print(int[] cells) {
            char ch;
            for(int i = 0; i < cells.Length; ++i) {
                ch = (cells[i] == 1) ? '\u25A0' // black square ■
                    : '\u25A1'; // white square □

                Console.Write(ch);
            }
            Console.WriteLine();
        }
    }
}

// CA.cs
using System;

```

```

namespace Problem3 {
    class CA {
    private int[] cells;
        protected int[] ruleset;
    public CA() {}
    public CA(int nCells, int spike_idx) {
        cells = new int[nCells];
        for(int i = 0; i < cells.Length; ++i) {
            cells[i] = 0;
        }
        cells[spike_idx] = 1;
        ruleset = new int[8];
        for(int i = 0; i < 8; ++i) {
            ruleset[i] = 0;
        }
    }

    virtual public void SetRuleSet() {}
    public int Rules(int a, int b, int c) {
        if(a == 1 && b == 1 && c == 1) return ruleset[0];
        if(a == 1 && b == 1 && c == 0) return ruleset[1];
        if(a == 1 && b == 0 && c == 1) return ruleset[2];
        if(a == 1 && b == 0 && c == 0) return ruleset[3];
        if(a == 0 && b == 1 && c == 1) return ruleset[4];
        if(a == 0 && b == 1 && c == 0) return ruleset[5];
        if(a == 0 && b == 0 && c == 1) return ruleset[6];
        if(a == 0 && b == 0 && c == 0) return ruleset[7];
        return 0;
    }
    public int[] Cells() {
        return cells;
    }
    public void GenerateNext() {
        int[] next = new int[cells.Length];
        for(int i = 1; i < cells.Length-1; ++i) {
            int left = cells[i-1];
            int me = cells[i];

```

```

        int right = cells[i+1];
        next[i] = Rules(left, me, right);
    }
    for(int i = 0; i < cells.Length; ++i) {
        cells[i] = next[i];
    }
    }
}

// CA_90.cs
// Cellular Automata with Wolfram's rule 90
using System;

namespace Problem3 {
    class CA_90 : CA {
        public CA_90(int nCells, int init_spike) :
            base(nCells, init_spike) {
            SetRuleSet();
        }
        override public void SetRuleSet() {
            ruleset[0] = 0;
            ruleset[1] = 1;
            ruleset[2] = 0;
            ruleset[3] = 1;
            ruleset[4] = 1;
            ruleset[5] = 0;
            ruleset[6] = 1;
            ruleset[7] = 0;
        }
    }
}

```

4. 試寫出以下程式在下列狀況時的主控台螢幕輸出。

(a) (3%) 檔案 **test.surface** 尚未建立。

(b) (3%) 檔案 **test.surface** 已在正確位置，且內容為


```
3 2
v 0 0
v 0.25 0
v 0 0.3
v 0.25 0.3
f 1 2 3
f 2 4 3
```

(c) (3%) 檔案 **test.surface** 已在正確位置，且內容為

```
4 2
v 0 0
v 0.25 0
v 0 0.3
v 0.25 0.3
f 0 1 2
f 1 3 2
```

(d) (3%) 檔案 **test.surface** 已在正確位置，且內容為

```
4 2
v 0 0
v 0.25 0
v 0 0.3
v 0.25 0.3
f 1 2 3
f 2 4 3
```

程式碼

```
// Problem4
using System;
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;
namespace Problem4 {
    class Program {
        static void Main(string[] args) {
```

```

try {
    string fileName = "test.surface";
    Surface surf = new Surface(fileName);
    Console.WriteLine();

    BinaryFormatter formatter = new BinaryFormatter();
    FileStream output = new FileStream("surface.buffer",
    FileMode.Create, FileAccess.Write);
    formatter.Serialize(output, surf);
    output.Close();
    Console.WriteLine();

    FileStream input = new FileStream("surface.buffer",
    FileMode.Open, FileAccess.Read);
    Object obj = formatter.Deserialize(input);
    if (obj.GetType() == surf.GetType()) {
        Surface surf1 = (Surface)obj;
        Console.WriteLine("nVertices: " + surf1.NVertices);
        Console.WriteLine("nFacets: " + surf1.NFacets);
        for(int i = 0; i < surf1.NFacets; ++i) {
            Console.WriteLine("Vertices of triangle " + i);
            Triangle tri = surf1.Triangles[i];
            Console.WriteLine("( " + tri.r1.x + ", " + tri.r1.y + ")");
            Console.WriteLine("( " + tri.r2.x + ", " + tri.r2.y + ")");
            Console.WriteLine("( " + tri.r3.x + ", " + tri.r3.y + ")");
        }
    }
    else {
        throw new SerializationException();
    }
} catch (AbnormalParsingException e) {
    Console.WriteLine(e);
} catch (FileNotFoundException) {
    Console.WriteLine("File not found");
} catch (SerializationException) {
    Console.WriteLine(
        "Error in serializing/deserializing objects");
} catch (IOException) {

```

```

        Console.WriteLine("Can not open or close file");
    } catch (Exception e) {
        Console.WriteLine(e.Message);
    }
    // ending the program
    Console.WriteLine();
    Console.WriteLine("按 enter/return 鍵結束");
    Console.ReadLine();
}
}

// Surface.cs
using System;
using System.IO;
namespace Problem4 {
    [Serializable]
    struct Vector2D {
        public double x;
        public double y;
        public Vector2D(double x, double y) {
            this.x = x;
            this.y = y;
        }
    }

    [Serializable]
    struct Triangle {
        public Vector2D r1;
        public Vector2D r2;
        public Vector2D r3;
        public Triangle(Vector2D r1, Vector2D r2, Vector2D r3) {
            this.r1 = r1;
            this.r2 = r2;
            this.r3 = r3;
        }
    }
}

```

```

[Serializable]
struct Facet {
    public int idx_r1;
    public int idx_r2;
    public int idx_r3;
    public Facet(int idx_r1, int idx_r2, int idx_r3) {
        this.idx_r1 = idx_r1;
        this.idx_r2 = idx_r2;
        this.idx_r3 = idx_r3;
    }
}

[Serializable]
class Surface {
    private int nVertices;
    private int nFacets;
    private Vector2D[] vertices;
    private Facet[] facets;
    private Triangle[] triangles;
    public int NVertices { get { return nVertices; } }
    public int NFacets { get { return nFacets; } }
    public Vector2D[] Vertices { get { return vertices; } }
    public Facet[] Facets { get { return facets; } }
    public Triangle[] Triangles { get {return triangles; } }

    public Surface(string fileName) {
        try {
            StreamReader input = new StreamReader(fileName);
            try {
                string indicator;
                double vx, vy;
                int iv1, iv2, iv3;
                string line = input.ReadLine();
                string[] terms = line.Split(" ");
                nVertices = int.Parse(terms[0]);
                nFacets = int.Parse(terms[1]);
                vertices = new Vector2D[nVertices];
                facets = new Facet[nFacets];
            }
        }
    }
}

```

```

triangles = new Triangle[nFacets];
for(int i = 0; i < nVertices; ++i) {
    line = input.ReadLine();
    terms = line.Split(" ");
    indicator = terms[0];
    if(indicator != "v") {
        throw new AbnormalParsingException("vertices");
    }
    vx = double.Parse(terms[1]);
    vy = double.Parse(terms[2]);
    vertices[i] = new Vector2D(vx, vy);
}
for(int i = 0; i < nFacets; ++i) {
    line = input.ReadLine();
    terms = line.Split(" ");
    indicator = terms[0];
    if(indicator != "f") {
        throw new AbnormalParsingException("facets");
    }
    iv1 = int.Parse(terms[1]);
    iv2 = int.Parse(terms[2]);
    iv3 = int.Parse(terms[3]);
    facets[i] = new Facet(iv1, iv2, iv3);
    Vector2D v1 = vertices[iv1-1];
    Vector2D v2 = vertices[iv2-1];
    Vector2D v3 = vertices[iv3-1];
    triangles[i] = new Triangle(v1, v2, v3);
}
} catch(AbnormalParsingException e) {
    Console.WriteLine(e.Message);
    throw e;
} catch(IndexOutOfRangeException e) {
    Console.WriteLine("index out of range");
    throw e;
} finally {
    Console.WriteLine(
        "Enter finally in constructor of Surface");
    input.Close();
}

```

```

        Console.WriteLine("Close file");
    }
} catch (Exception e) {
    Console.WriteLine(
        "Throw an exception from constructor of Surface");
    throw e;
}
}
}

// AbnormalParsingException.cs
using System;

namespace Problem4 {
    class AbnormalParsingException : ApplicationException {
        private string indicator;
        public AbnormalParsingException(string indicator): base() {
            this.indicator = indicator;
        }
        public override string ToString() {
            return "Exception in parsing " + indicator;
        }
    }
}
}

```

5. 依據以下描述及 Unity C# 腳本程式，回答問題。(6%)

程式描述：顯示一個名為 Quit 的按鈕(圖 5)，按了之後就結束程式的執行。

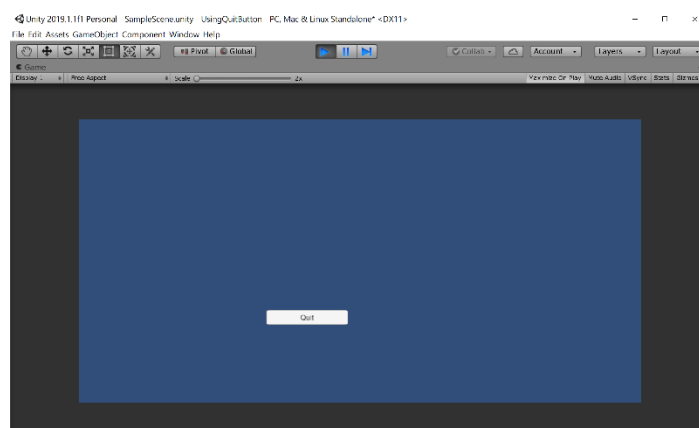


圖 5. 有一個 Quit 按鈕的圖形介面。


問題: 對應的 Unity C# 腳本如下，請回答第 18 行

```
EditorApplication.isPlaying = false;
```

和第 20 行

```
Application.Quit();
```

的程式敘述，功能分別為何。



```
C# QuitButton.cs x
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  #if UNITY_EDITOR
5  using UnityEditor;
6  #endif
7
8  0 references
9  public class QuitButton : MonoBehaviour
10 {
11     0 references
12     public void OnPressed()
13     {
14         Quit();
15     }
16
17     1 reference
18     private void Quit()
19     {
20         #if UNITY_EDITOR
21             EditorApplication.isPlaying = false;
22         #else
23             Application.Quit();
24         #endif
25     }
26 }
```

圖 6. Quit 按鈕的 C# 腳本程式

6. 大富翁(Monopoly)遊戲是很風行的桌遊，很多人從小到大都玩過。本題希望能夠撰寫一個相似的遊戲。由於時間的限制，我們不可能鉅細靡遺地模擬整個遊戲。仔細想一想，大富翁遊戲的精隨在於購買地產，讓別的玩家走到我

們地產，付出高額過路費，以致破產，讓我們堅持到最後，成為大富翁。

因此，依據這個主要的概念，可以把大富翁的遊戲盤(board)簡化為圖 7：

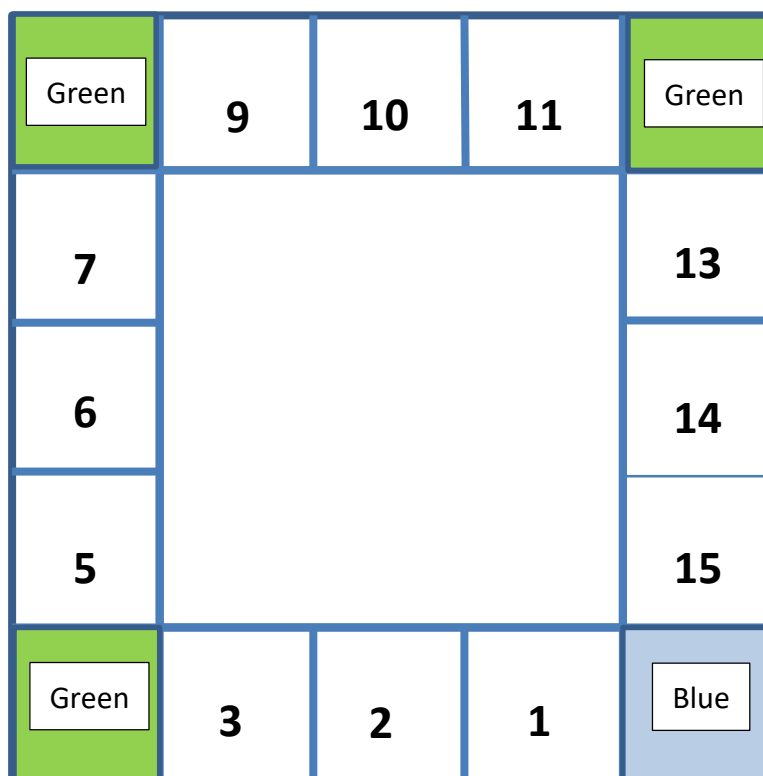


圖 7. 簡化大富翁遊戲的遊戲盤

兩個玩家(人類及電腦)由藍色方塊出發，各發 2000 元現金，先後擲一個骰子，決定往順時針方向的前進步數。有數目字的格子是地產，一開始的時候都是銀行所有，最先走到的玩家可以依照數字乘以 100 的價格(price，例如，數字 14 的地產，定價為 1400 元)購買。如果走到另一個玩家所有的地產，需要照價格的一半，付給擁有該地產的玩家過路費(toll)。當某一玩家無法支付過路費(手上現金少於過路費)，便宣告破產，另一玩家獲勝。藍色(相當於數字 0 的方格)及綠色方塊(相當於數字 4、8、12 的方格)，是可以免費過境的地區。為早點產生大富翁，玩家通過或再次停到藍色方格時，不能支領薪津，只能依賴另一玩家支付過路費，增加手中現金。一個典型的遊戲過程摘要，顯示於圖 8 的主控台螢幕。當人類玩家走到某個可購買之地產時，系統會詢問玩家是否願意購買，而電腦玩家則依照產生的亂數是否為偶數，決定是否購買地產。


```

選取 C:\Progra...  -  □  ×
第1回合
skjeng資產:
現金 2000
地產: 無

Computer資產:
現金 2000
地產: 無

按 enter 鍵以擲出骰子

擲出了 1
抵達 cell 1

地產 1 出售
價格: 100
過路費: 50
要購買嗎?
Y
skjeng購買了地產 1

電腦擲出 5
抵達 cell 5
Computer購買了地產 5

```

```

選取 C:\Progra...  -  □  ×
第3回合
skjeng資產:
現金 1700
地產: 1, 2

Computer資產:
現金 800
地產: 5, 7

按 enter 鍵以擲出骰子

擲出了 1
抵達 cell 3

地產 3 出售
價格: 300
過路費: 150
要購買嗎?
Y
skjeng購買了地產 3

電腦擲出 2
抵達 cell 9
Computer沒有買地產 9

```

```

選取 C:\Progra...  -  □  ×
第2回合
skjeng資產:
現金 1900
地產: 1

Computer資產:
現金 1500
地產: 5

按 enter 鍵以擲出骰子

擲出了 1
抵達 cell 2

地產 2 出售
價格: 200
過路費: 100
要購買嗎?
Y
skjeng購買了地產 2

電腦擲出 2
抵達 cell 7
Computer購買了地產 7

```

```

選取 C:\Progra...  -  □  ×
第6回合
skjeng資產:
現金 800
地產: 1, 2, 3

Computer資產:
現金 1400
地產: 5, 7

按 enter 鍵以擲出骰子

擲出了 4
抵達 cell 11

地產 11 出售
價格: 1100
過路費: 550
現金不足, 無法購買
skjeng沒有買地產 11

電腦擲出 2
抵達 cell 0
免費過境

```

```
第14回合
skjeng資產:
現金 500
地產: 1, 2, 3, 6

Computer資產:
現金 0
地產: 5, 7, 11

按 enter 鍵以擲出骰子

擲出了 2
抵達 cell 4
免費過境

電腦擲出 6
抵達 cell 4
免費過境
```

```
第15回合
skjeng資產:
現金 500
地產: 1, 2, 3, 6

Computer資產:
現金 0
地產: 5, 7, 11

按 enter 鍵以擲出骰子

擲出了 1
抵達 cell 5
此為Computer地產, 付過路費250

電腦擲出 2
抵達 cell 6
此為skjeng地產, 付過路費300

Computer破產

按 enter/return 鍵結束
```

圖 8. 簡化大富翁遊戲，第 1、2、3、6、14、15 回合，主控台畫面摘要

本題希望你充分應用前述內容，撰寫達成以上功能的 C# 程式。

本題滿分 25 分，全部程式集中寫成一個大 **Main** 函式，不區分其他函式者，最高得 18 分；善用函式者，最高得 20 分；能利用虛擬碼或 UML 類別圖思考，適當劃分類別(class)者，最高得 22 分；善用類別繼承與多型(polymorphism)者，最高得 25 分。(25%)

7. 請寫下本課程教學「待改進」之處及改進方法建議。(3%)