

# 計算機程式設計

## 主題 繪圖程式

機械一 朱本毅

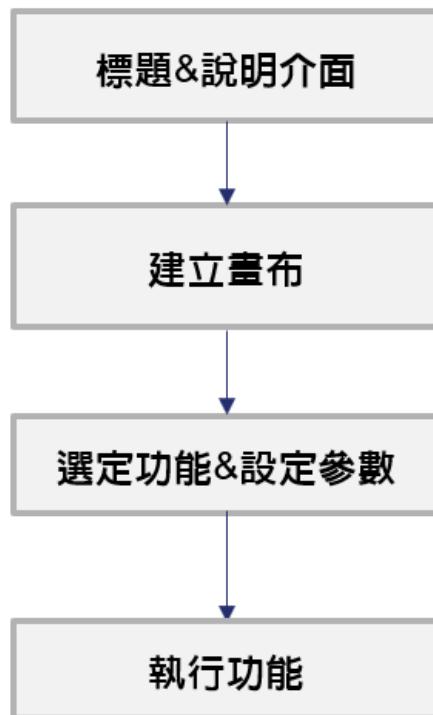
### 一、動機:

系上必修有工程圖學，那就做個繪圖程式好了

網路上看到的 繪圖程式都有搭配 視窗，想試試看 有多難用

練習用物件以及

### 二、程式架構與規劃:



## 我會需要的物件

一張畫布，用來執行繪圖功能

```
class Paper
{
    Memory[,] memory; //引入類別 Memory的二維陣列 儲存畫布上的資訊
    int xMax = 0;
    int yMax = 0;
    char currentChar = ' ';
    ConsoleColor currentForeColor = ConsoleColor.Black;
    ConsoleColor currentBackColor = ConsoleColor.White;
    public Paper() //畫布建構式 預設為10x10
    {
        memory = new Memory[10,10];
        for (int i = 0; i < 10; ++i) //y
        {
            xMax = 10;
            yMax = 10;
            for (int j = 0; j < 10; ++j) //x
            {
                memory[i, j] = new Memory();
                memory[i, j].BackGroundColor = currentBackColor;
                memory[i, j].ForeColor = currentForeColor;
                memory[i, j].Graph = ' ';
            }
            currentBackColor = ConsoleColor.Black;
        }
    }
    public Paper(int xMax, int yMax)
    {
        this.xMax = xMax;
        this.yMax = yMax;
        memory = new Memory[yMax, xMax];
        for (int i = 0; i < yMax; ++i)
        {
            for (int j = 0; j < xMax; ++j)
            {
                memory[i, j] = new Memory();
                memory[i, j].BackGroundColor = currentBackColor;
                memory[i, j].ForeColor = currentForeColor;
                memory[i, j].Graph = ' ';
            }
        }
        currentBackColor = ConsoleColor.Black;
    }
}
```

儲存格，為一個二維陣列，大小由一開始建立的畫布決定，不浪費多於記憶體，每個儲存格紀錄畫布上對應格子的資訊，包括字元、前景色、背景色

```
class Memory //每格位置的記憶體 儲存顏色、字元資料
{
    public ConsoleColor ForegroundColor;
    public ConsoleColor BackGroundColor;
    public char Graph;
```

位置，紀錄當前作圖的起始點、終點等資訊，建立函數以精簡程式碼

```
class Position //紀錄座標資訊
{
    /* -----> X 軸
     * |
     * |
     * |
     * V
     * Y 軸
    */
    private int x;
    private int y;
    public int X { get => x; set => x = value; }
    public int Y { get => y; set => y = value; }
    public void SetX()
    {
        Console.WriteLine("請輸入x座標(1 to xMax)");
        x = int.Parse(Console.ReadLine());
    }
    public void SetY()
    {
        Console.WriteLine("請輸入y座標(1 to yMax)");
        y = int.Parse(Console.ReadLine());
    }
}
```

### 三、功能：

畫點、線、面和方塊，可搭配設定符號，以符號繪圖。

受限於 視窗的視窗、其他形狀太複雜了畫不出來

大致上就是先設定繪圖起始點，再設定參數，之後把結果紀錄進去儲存格。

最基本的就是畫出一點，有設定 判斷畫圖位置，避免因為輸入座標超出畫布中

用來記憶繪圖資訊的陣列，導致程式直接結束

```
public void DrawDot() //填入一格顏色
{
    Console.Clear();
    Position dot = new Position();
    dot.SetX();
    dot.SetY();

    bool outOfRange = (dot.X > xMax || dot.Y > yMax) ? true : false;
    if (outOfRange == false)
    {
        memory[dot.Y - 1, dot.X - 1].BackGroundColor = currentBackGroundColor;
        memory[dot.Y - 1, dot.X - 1].Graph = currentChar;
    }
    else
    {
        Console.WriteLine("超出畫布範圍");
        Console.ReadLine();
    }
}
```

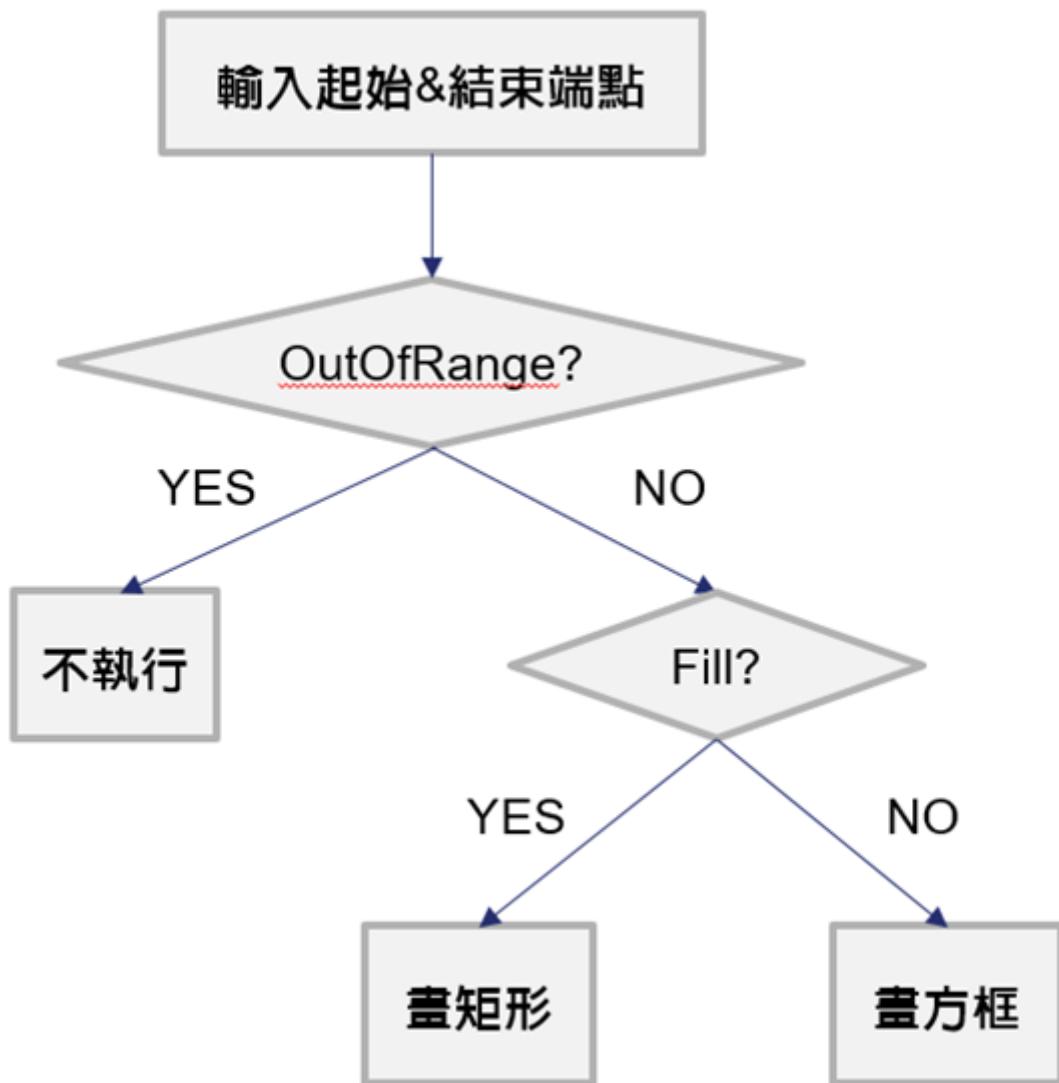
## 再來就是畫線，增加了線條畫出方向的功能

```
設定起始位置(左上)
請輸入x座標(1 to xMax)
3
請輸入y座標(1 to yMax)
5
請輸入線段長度
10
橫畫(→)or直畫(↓)
```

```
private void DrawLine()
{
    Console.WriteLine("請輸入線段長度");
    int length = int.Parse(Console.ReadLine());
    Console.WriteLine("橫畫(→)or直畫(↓)");
    ConsoleKey input = Console.ReadKey().Key;
    if (input == ConsoleKey.RightArrow)
    {
        bool outOfRange = (p.X + length) > xMax ? true : false;
        if (outOfRange == false)
        {
            for (int i = p.X; i < p.X + length; ++i)
            {
                memory[p.Y - 1, i - 1].BackGroundColor = currentBackGroundColor;
                memory[p.Y - 1, i - 1].Graph = currentChar;
            }
        }
        else
        {
            Console.WriteLine("超出畫布範圍");
            Console.ReadLine();
        }
    }
    else if (input == ConsoleKey.DownArrow)
    {
        bool outOfRange = (p.Y + length) > yMax ? true : false;
        if (outOfRange == false)
        {
            for (int i = p.Y; i < p.Y + length; ++i)
            {
                memory[i - 1, p.X - 1].BackGroundColor = currentBackGroundColor;
                memory[i - 1, p.X - 1].Graph = currentChar;
            }
        }
        else
        {
            Console.WriteLine("超出畫布範圍");
            Console.ReadLine();
        }
    }
}
```

而畫方塊的部分，會判斷是否要將方塊填滿

```
設定起始位置(左上)
請輸入x座標(1 to xMax)
2
請輸入y座標(1 to yMax)
2
設定結束位置(右下)
請輸入x座標(1 to xMax)
40
請輸入y座標(1 to yMax)
20
要填滿嗎(Y/N)
```

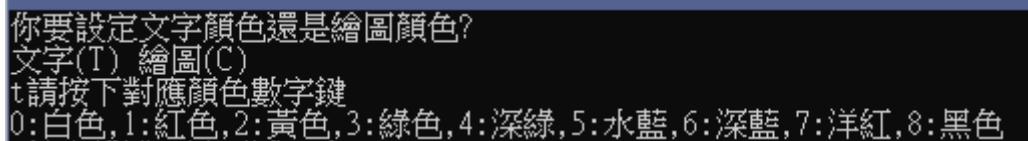


## 橡皮擦

選取範圍，清除儲存格紀錄的字元，將前景和背景色調整為預設

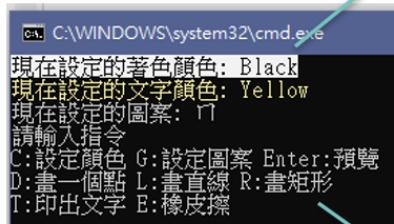
調整色彩，前景色代表文字，背景色則為主要畫出的色彩，在主要的 介面中，

顯示當前資訊的部分也會隨著當前色彩而改變



```
public void SetColor() //設定顏色
{
    Console.Clear();
    Console.WriteLine("你要設定文字顏色還是繪圖顏色?\n" +
                      "文字(T) 繪圖(C)");
    ConsoleKey choose = Console.ReadKey().Key;
    Console.WriteLine("請按下對應顏色數字鍵");
    Console.WriteLine("0:白色,1:紅色,2:黃色,3:綠色,4:深綠,5:水藍,6:深藍,7:洋紅,8:黑色");
    bool selected = false;
    do
    {
        ConsoleKey input = Console.ReadKey().Key;
        switch (input){...}
    } while (selected != true);
}
```

顯示當前繪圖資訊  
顏色分為前景色以及背景  
(會隨當前顏色改變)



指令區  
以及輸入參數的位置

## 在指定位置、方向印出文字

將輸入的字串  
，依照指定的方向逐一輸出

```
public void Text() //畫出一行字串
{
    Console.Clear();
    Console.WriteLine("設定起始位置(左上)");
    Position position = new Position();
    position.SetX();
    position.SetY();
    Console.WriteLine("請輸入字串(English Only 不然中文字的寬度是兩格)");
    char[] text = Console.ReadLine().ToCharArray();
    Console.WriteLine("橫畫(→)or直畫(↓)");
    ConsoleKey input = Console.ReadKey().Key;
    int charCount;
    if (input == ConsoleKey.RightArrow)
    {
        bool outOfRange = (position.X + text.GetLength(0)) > xMax ? true : false;
        if (outOfRange == false)
        {
            charCount = 0;
            for (int i = position.X; i < position.X + text.GetLength(0); ++i)
            {
                memory[position.Y - 1, i - 1].ForeColor = currentForeColor;
                memory[position.Y - 1, i - 1].BackColor = currentBackColor;
                memory[position.Y - 1, i - 1].Graph = text[charCount];
                ++charCount;
            }
        }
        else
        {
            Console.WriteLine("超出畫布範圍");
            Console.ReadLine();
        }
    }
}
```

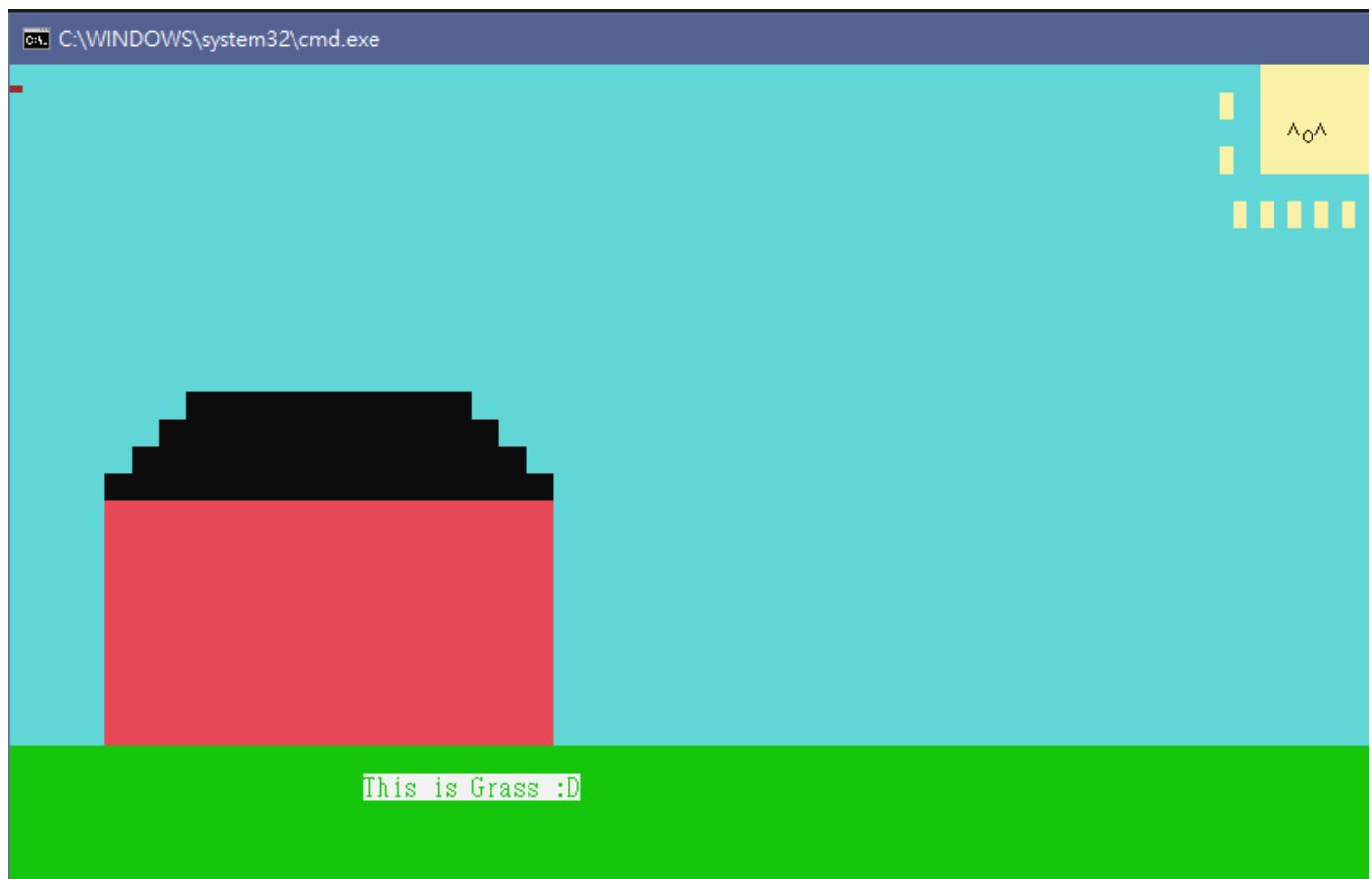
## 預覽當前繪圖結果

```
public void Preview()
{
    Console.Clear();
    for (int i = 0; i < yMax; ++i) //y
    {
        for (int j = 0; j < xMax; ++j) //x
        {
            Console.BackgroundColor = memory[i, j].BackColor;
            Console.ForegroundColor = memory[i, j].ForeColor;
            Console.Write(memory[i, j].Graph);
        }
        Console.WriteLine();
    }
    ConsoleCursorPosition(0, 0);
    Console.ReadLine();
    Console.BackgroundColor = ConsoleColor.Black;
}
```

#### 四、遇到的困難 反思

首先，我預計這個程式即便功能不多，但仍舊能在視覺上有一定程度的表現，想是繪圖用的字元可以調整，也可以直接輸入字串，其實要做出指定大小的表格或報表等等也是可行的，但主要功能還是在繪圖。

然而我這次最大的敗筆就是在繪圖方面，以底下這個圖為例，照座標布圖、調整就花了十五分鐘，看起來比我幼稚園畫的還爛。



以下是我的心得及反思

如果使用方向鍵操控著色位置，這樣就可以直接看著畫布做畫，同時也知道自己當前使用的顏色及字元，這樣效率會更高。

有些功能，硬要用物件去寫，反而使得程式碼變得比較繁雜，但我的主要目的是練習使用物件，這也無可奈何。不過在撰寫過程中也可明顯見其優點。

本身就不是很好的繪圖環境，每格有一定的長寬比例，而且英文、符號和中文

的字寬又不一樣，中文會使得畫布的位置跑掉，除非我要以兩格作為單位來記錄繪圖資訊，但這樣會浪費兩倍的記憶體。