第 8 章簡答題 2

方法	輸出
ToUpper()	VISUAL C# 程式設計範例教本
Substring(2,4)	sual
IndexOf("程式")	10

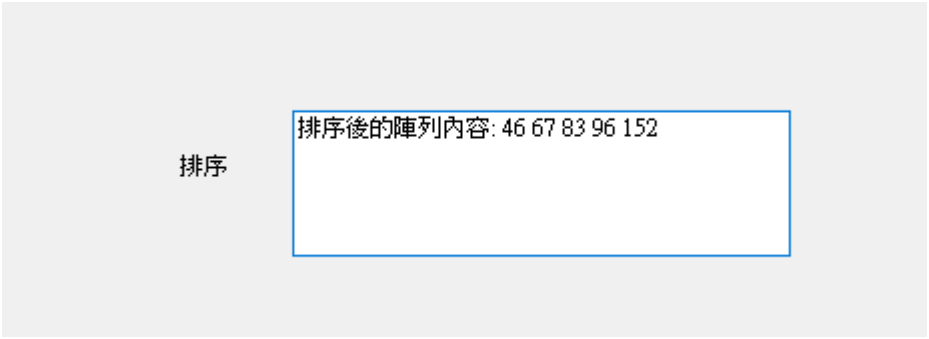
第 8 章簡答題 6

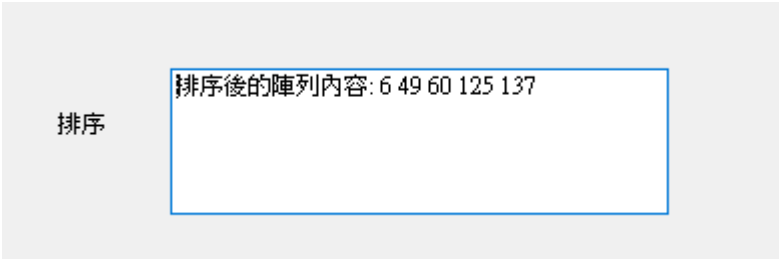
搜尋（Search）：
指的是在一堆資料中找出特定資料的過程，比如從一個名單中找出某個人的名字。

排序（Sort）：
指的是依照特定規則（例如大小、字母順序）把資料重新排列，例如把一堆數字從小排到大。

無序搜尋（Unordered Search）、有序搜尋（Ordered Search）

第 8 章實作題 2



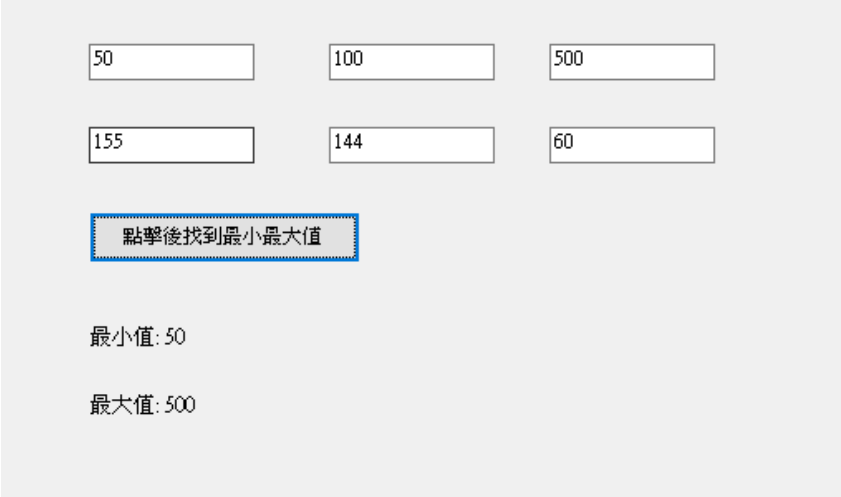


label1 的副程式:

```
private void label1_Click(object sender, EventArgs e)
 {
     int[] arr = new int[5]; // 宣告5個元素的一維陣列
     Random rnd = new Random(); // 建立亂數物件

     // 產生 1~200 的亂數並填入陣列
     for (int i = 0; i < arr.Length; i++)
     {
         arr[i] = rnd.Next(1, 201); // 產生1~200之間的整數
     }

     // 排序陣列
     Array.Sort(arr);

     // 將排序後的陣列內容組成字串
     string result = "排序後的陣列內容:\n";
     foreach (int num in arr)
     {
         result += num.ToString() + "\n";
     }
```

```
    // 顯示在 Label 控制項上
    textBox1.Text = result;
}
```

第 8 章實作題 4




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace 實作8_4
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        // 找最小值的函數
        private int arrMin(int[] arr)
```

```
{
    int min = arr[0];
    foreach (int num in arr)
    {
        if (num < min)
            min = num;
    }
    return min;
}

// 找最大值的函數
private int arrMax(int[] arr)
{
    int max = arr[0];
    foreach (int num in arr)
    {
        if (num > max)
            max = num;
    }
    return max;
}
private void bt1_Click(object sender, EventArgs e)
{
    try
    {
        // 將6個TextBox的數值取出並轉成整數陣列
        int[] numbers = new int[6]
        {
            int.Parse(textBox1.Text),
            int.Parse(textBox2.Text),
            int.Parse(textBox3.Text),
            int.Parse(textBox4.Text),
            int.Parse(textBox5.Text),
            int.Parse(textBox6.Text)
        };

        // 呼叫函數找最小值和最大值
        int minValue = arrMin(numbers);
```

```
                int maxValue = arrMax(numbers);

                // 顯示結果到Label
                label1.Text = "最小值: " + minValue.ToString();
                label2.Text = "最大值: " + maxValue.ToString();
            }
            catch (Exception ex)
            {
                MessageBox.Show("請正確輸入6個整數！\n" + ex.Message);
            }
        }
    }

}
```

第 9 章簡答題 6

傳統應用程式開發（程序導向開發）

- 重點在「流程」：一步一步照著流程處理資料。
- 程式以功能為中心，像是「輸入資料 → 處理資料 → 輸出資料」。
- 資料和操作資料的程式通常是分開的。
- 開發時，比較偏向寫一堆**函式（Function）**來完成各種事情。
- 缺點是：程式變大時，修改或擴充很容易出錯，維護困難。

物件導向應用程式開發（OOP）

- 重點在「物件」：把資料和對資料的操作打包在一起，變成「物件」。
- 程式以物件（Object）為中心，每個物件代表一個真實世界中的東西或概念。
- 強調封裝（Encapsulation）、繼承（Inheritance）、**多型（Polymorphism）**這三大特性。
- 好處是：程式模組化程度高，容易維護、擴充、重複使用。

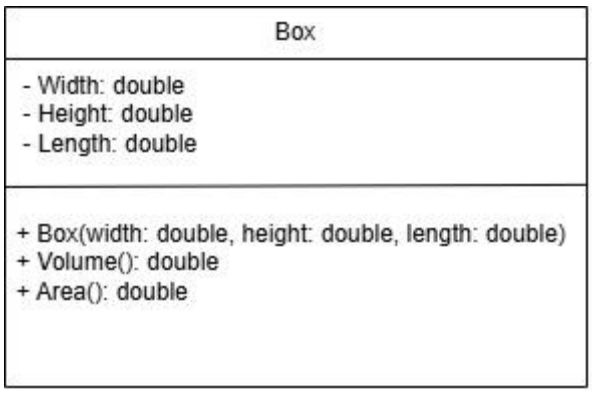
第 9 章簡答題 1

修飾子	說明	可以被誰使用
private	私有，只能在自己這個類別內使用。	自己類別內部
protected	受保護，自己用得到，子類別也用得到。	自己類別 + 繼承它的子類別
public	公開，任何地方都能用。	任何地方

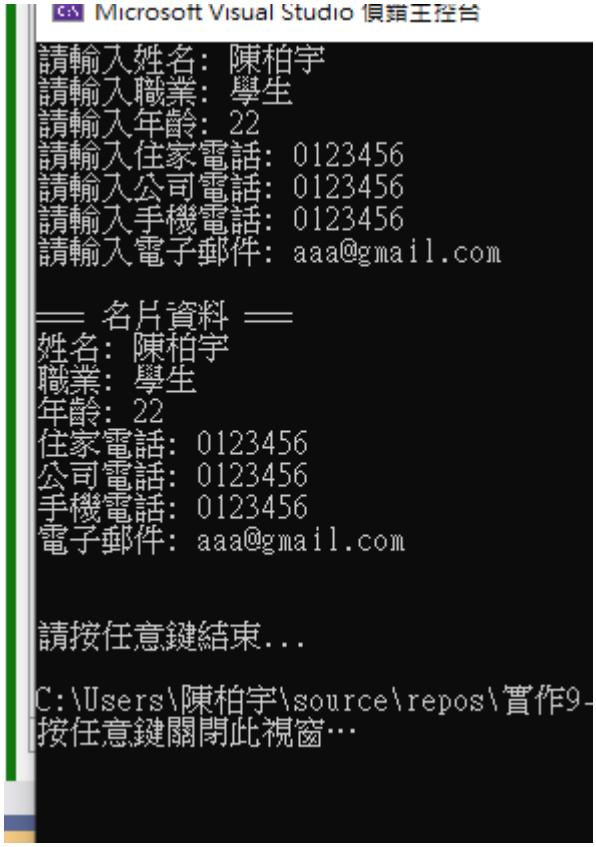
第 9 章實作題 2


```
class Box
{
    private double Length;
    private double Width;
    private double Height;


    public Box(double width, double height, double length)
    {
        Length = length;
        Width = width;
        Height = height;
```

```
    }

    public double Volume()
    {
        return Width * Height * Length;
    }

    public double Area()
    {
        return   Width * Length;
    }
}

private void button1_Click(object sender, EventArgs e)
{
    try
    {
        // 讀取使用者輸入
        double length = double.Parse(textBox1.Text);
        double width = double.Parse(textBox2.Text);
        double height = double.Parse(textBox3.Text);

        // 建立 Box 物件
        Box myBox = new Box(width, height, length);

        // 計算體積和面積
        double volume = myBox.Volume();
        double area = myBox.Area();

        // 顯示結果到 Label
        label1.Text = "體積: " + volume.ToString();
        label2.Text = "面積: " + area.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show("請輸入正確的數字！\n" + ex.Message);
    }
}
```



第 9 章實作題 4




```
using System;

class PhoneList
{
    public string HomePhone;
    public string BusinessPhone;
```

```
    public string CellPhone;

    public PhoneList(string homePhone, string businessPhone, string cellPhone)
    {
        HomePhone = homePhone;
        BusinessPhone = businessPhone;
        CellPhone = cellPhone;
    }
}

class Cards
{
    private string Name;
    private string Occupation;
    private int Age;
    private PhoneList Phone;
    private string Email;

    public Cards(string name, string occupation, int age, PhoneList phone, string
email)
    {
        Name = name;
        Occupation = occupation;
        Age = age;
        Phone = phone;
        Email = email;
    }

    public string GetCard()
    {
        string cardInfo = "";
        cardInfo += $"姓名: {Name}\n";
        cardInfo += $"職業: {Occupation}\n";
        cardInfo += $"年齡: {Age}\n";
        cardInfo += $"住家電話: {Phone.HomePhone}\n";
        cardInfo += $"公司電話: {Phone.BusinessPhone}\n";
        cardInfo += $"手機電話: {Phone.CellPhone}\n";
        cardInfo += $"電子郵件: {Email}\n";
```

```
        return cardInfo;
    }
}

class Program
{
    static void Main(string[] args)
    {
        // 提示使用者輸入資料
        Console.Write("請輸入姓名: ");
        string name = Console.ReadLine();

        Console.Write("請輸入職業: ");
        string occupation = Console.ReadLine();

        Console.Write("請輸入年齡: ");
        int age = int.Parse(Console.ReadLine());

        Console.Write("請輸入住家電話: ");
        string homePhone = Console.ReadLine();

        Console.Write("請輸入公司電話: ");
        string businessPhone = Console.ReadLine();

        Console.Write("請輸入手機電話: ");
        string cellPhone = Console.ReadLine();

        Console.Write("請輸入電子郵件: ");
        string email = Console.ReadLine();

        // 建立 PhoneList 和 Cards 物件
        PhoneList phone = new PhoneList(homePhone, businessPhone, cellPhone);
        Cards myCard = new Cards(name, occupation, age, phone, email);

        // 顯示名片資料
        Console.WriteLine("\n=== 名片資料 ===");
        Console.WriteLine(myCard.GetCard());
```

```
            Console.WriteLine("\n請按任意鍵結束...");
            Console.ReadKey();
        }
    }
```