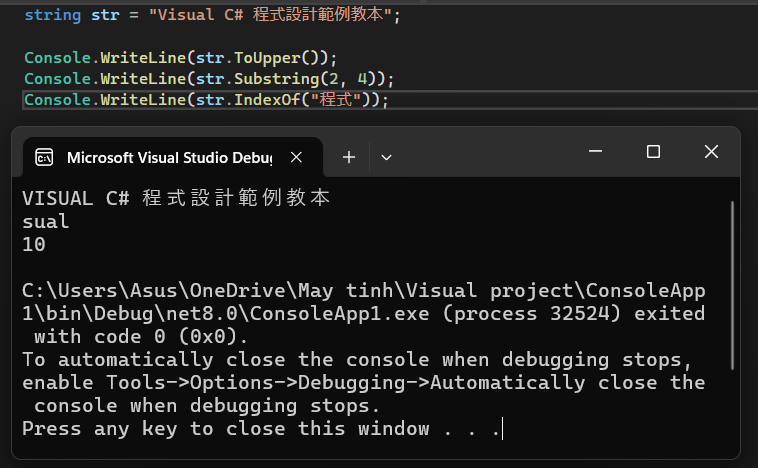
**章8**

簡答題

2.



6

**排序**工作是將一些資料依照特定原則排列成遞增或遞減的順序。

**搜尋**是在資料中找尋特定值,這個值稱為「鍵值」(Key)。

搜尋方法依照搜尋的資料分為兩種,如下所示:

**沒有排序的資料**:針對沒有排序的資料執行搜尋,我們需要從資料的第1個元素開始比較,從頭到尾以確認資料是否存在。

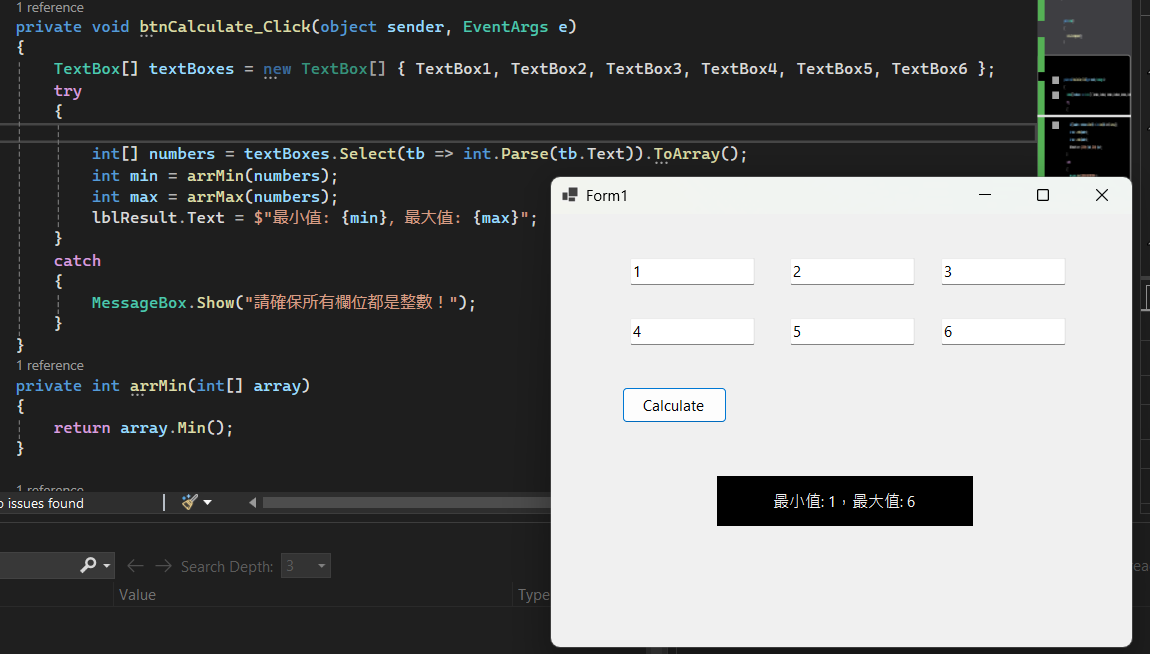
**已經排序的資料**:資料已經排序,所以搜尋就不需從頭開始——比較。例如:在電話簿找電話,相信沒有人是從電話薄的第1頁開始找,而是直接從姓名出現的頁數開始找,因為電話簿已經依照姓名進行排序。

實作題

2.



4.



**章9**

簡答題

1.

 **物件導向程式開發**

* **概念**：以物件為核心，強調封裝、繼承、多型等特性。
* **設計方式**：程式是由許多物件互相合作來完成目標，每個物件具有自己的屬性（數據）與方法（行為）。
* **好處**：代碼可重用性高、維護容易、擴展方便，適合複雜的大型系統開發。
* **實例**：C#、Java、Python 都是支持 OOP 的程式語言。

 **傳統應用程式開發**

* **概念**：以程序（步驟）為核心，注重邏輯流程和功能模塊。
* **設計方式**：程式是由一組步驟或函數來解決問題，數據與邏輯分離。
* **缺點**：程式重用性低，維護和擴展困難，對於大型專案往往不夠靈活。
* **實例**：C、Pascal 屬於程序導向語言。

6.

**public**:

* 表示成員可以被任何地方存取，無限制。

**private**:

* 表示成員僅能在宣告它的類別內部存取。

**protected**:

* 表示成員只能被宣告它的類別或其子類別存取。

**工具方法 (Utility Methods)**：

* **概念**：工具方法是一些不依賴物件實例即可使用的功能方法，通常設為靜態方法（static）。
* **特徵**：
  + 為特定任務設計，重用性高。
  + 通常存放於「工具類別」（Utility Classes）中。

實作題

2. using System;

public class Box

{

public double Width { get; set; }

public double Height { get; set; }

public double Length { get; set; }

public Box(double width, double height, double length)

{

Width = width;

Height = height;

Length = length;

}

public double Volume()

{

return Width \* Height \* Length;

}

public double Area()

{

return 2 \* (Width \* Height + Width \* Length + Height \* Length);

}

}

class Program

{

static void Main(string[] args)

{

Box myBox = new Box(3.0, 4.0, 5.0);

double volume = myBox.Volume();

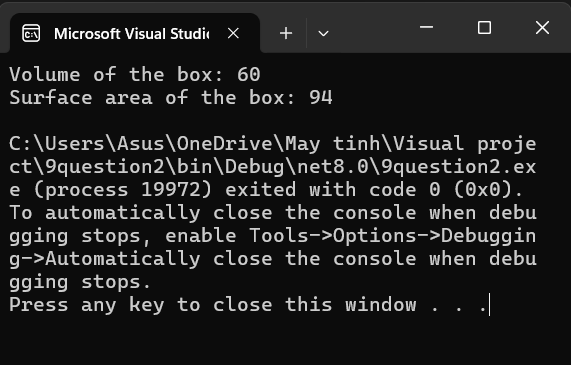
Console.WriteLine("Volume of the box: " + volume);

double area = myBox.Area();

Console.WriteLine("Surface area of the box: " + area);

}

}



|  |
| --- |
| Box |
| - Width: double  - Height: double  - Length: double |
| + Box(double, double, double)  + Volume(): double  + Area(): double |

4.

using System;

public class PhoneList

{

public string HomePhone { get; set; }

public string BusinessPhone { get; set; }

public string CellPhone { get; set; }

public PhoneList(string homePhone, string businessPhone, string cellPhone)

{

HomePhone = homePhone;

BusinessPhone = businessPhone;

CellPhone = cellPhone;

}

}

public class Cards

{

public string Name { get; set; }

public string Occupation { get; set; }

public int Age { get; set; }

public PhoneList Phone { get; set; }

public string Email { get; set; }

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()

{

return $"Name: {Name}\nOccupation: {Occupation}\nAge: {Age}\n" +

$"Home Phone: {Phone.HomePhone}\nBusiness Phone: {Phone.BusinessPhone}\n" +

$"Cell Phone: {Phone.CellPhone}\nEmail: {Email}";

}

}

class Program

{

static void Main(string[] args)

{

PhoneList myPhones = new PhoneList("123-456-6543", "0979788149", "555-123-3333");

Cards myCard = new Cards("DUONG THI LAM THUY ", "Student", 20, myPhones, "dichduongthientuy22082004@gmail.com");

Console.WriteLine("Card Information:");

Console.WriteLine(myCard.GetCard());

}

}

