## 第8章 - 簡答題

2.

str.ToUpper(); // "VISUAL C# 程式設計範例教本"   
str.Substring(2, 4); // "sual"   
str.IndexOf("程式"); // 10

6.  
搜尋是指在資料集合中，找出符合條件的資料位置或內容。  
排序是指將一組資料依照某種規則（如數字大小、字母順序）重新排列。  
  
搜尋方法可分為：  
- 無序搜尋  
- 有序搜尋

## 第8章 - 實作題

2.

private void button1\_Click(object sender, EventArgs e)  
{  
 // 宣告亂數物件  
 Random rnd = new Random();  
  
 // 建立並產生5個1~200的亂數整數  
 int[] numbers = new int[5];  
 for (int i = 0; i < numbers.Length; i++)  
 {  
 numbers[i] = rnd.Next(1, 201); // 上限為201，實際只到200  
 }  
  
 // 排序陣列  
 Array.Sort(numbers);  
  
 // 顯示結果到 label  
 label1.Text = "排序後的陣列： " + string.Join(", ", numbers);  
}

4.

private int arrMin(int[] arr) => arr.Min();  
private int arrMax(int[] arr) => arr.Max();  
  
private void button1\_Click(object sender, EventArgs e)  
{  
 try  
 {  
 // 將輸入字串分割並轉成整數陣列  
 int[] numbers = textBox1.Text  
 .Split(',')  
 .Select(n => int.Parse(n.Trim()))  
 .ToArray();  
  
 if (numbers.Length != 6)  
 {  
 MessageBox.Show("請輸入 6 個整數，用逗號分隔", "格式錯誤");  
 return;  
 }  
  
 int min = arrMin(numbers);  
 int max = arrMax(numbers);  
  
 label1.Text = $"最小值: {min}，最大值: {max}";  
 }  
 catch  
 {  
 MessageBox.Show("輸入格式錯誤，請輸入整數", "錯誤");  
 }  
}

## 第9章 - 簡答題

6.

public 任何地方都能存取 最開放，用於希望其他類別也能使用的成員  
protected 只有自己類別 + 子類別（繼承類別）能存取，常用於類別設計中允許子類別自訂行為  
private 只有自己類別內部能存取，最封閉，用來保護內部資料或方法不被濫用  
  
工具方法（utility methods）是指那些提供通用功能、可重複使用的靜態方法，通常不依賴特定物件狀態。

1.  
程序式：像是在工廠裡，一條生產線上所有步驟緊密相連，任何改動都可能影響整條線。  
  
物件導向：像是把工廠分成模組化「工作站」，每個工作站（物件）自己負責一段流程，彼此透過明確介面協作，方便增減或更換

## 第9章 - 實作題

2.

class Program  
{  
 static void Main(string[] args)  
 {  
 // 創建 Box 類別物件  
 Box myBox = new Box();  
  
 // 設定屬性值  
 myBox.Width = 5.0;  
 myBox.Height = 3.0;  
 myBox.Length = 4.0;  
  
 // 計算並顯示體積  
 double volume = myBox.Volume();  
 Console.WriteLine($"這個盒子的體積是: {volume} 立方單位");  
 }  
}  
  
public class Box  
{  
 public double Width { get; set; }  
 public double Height { get; set; }  
 public double Length { get; set; }  
  
 public double Volume()  
 {  
 return Width \* Height \* Length;  
 }  
}

UML類別圖

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

4.

internal class Program  
{  
 static void Main(string[] args)  
 {  
 Cards myCard = new Cards  
 {  
 Name = "John",  
 Occupation = "Software Engineer",  
 Age = 25,  
 Phone = "123-456-7890",  
 Email = "zzzz@example.com"  
 };  
 Console.WriteLine(myCard.GetCard());  
 Console.ReadKey();  
 }  
}  
  
public class Cards  
{  
 public string Name { get; set; }  
 public string Occupation { get; set; }  
 public int Age { get; set; }  
 public string Phone { get; set; }  
 public string Email { get; set; }  
  
 public string GetCard()  
 {  
 return $"[名片資訊]\n" +  
 $"姓名：{Name}\n" +  
 $"職業：{Occupation}\n" +  
 $"年齡：{Age}\n" +  
 $"電話：{Phone}\n" +  
 $"Email：{Email}";  
 }  
}