1. 請從技術面及使用面解釋interface及abstract的差異。(25%)

使用面:interface用在當一個物件須要和其他物件共同使用時，為了要確保

其他物件有我想要的方法method，所以定下interface要該物件遵守，在設計模式下常見，如bridge。

則abstract用在繼承體系的最上層，用來定義出整個繼承體系該有哪些

方法，子類別可以對這些方法加以override，或維持和abstract相同

的功能。在設計模式中如template method。

技術面:interface使用組合技術，必須建立一個interface，強迫要被你組合的

物件，需實做這個interface，這樣當你要使用這個物件時，才能確保

有方法method可以呼叫。

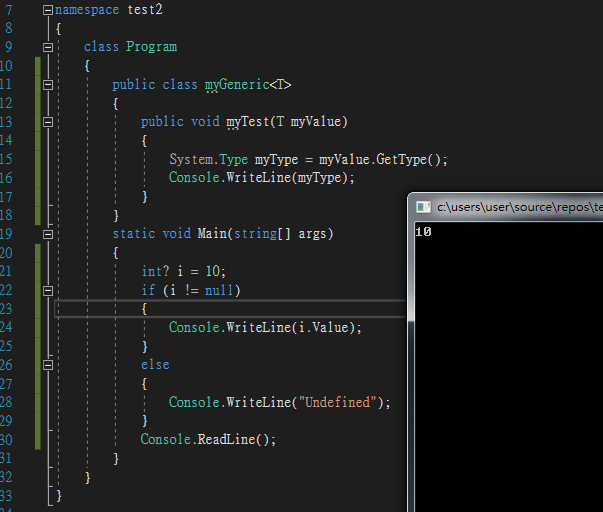
abstract使用繼承技術，將所有method由abstract去宣告，然後由個

子類別去override，若不得已某些class有自己的特殊方法，則由class自己宣告。

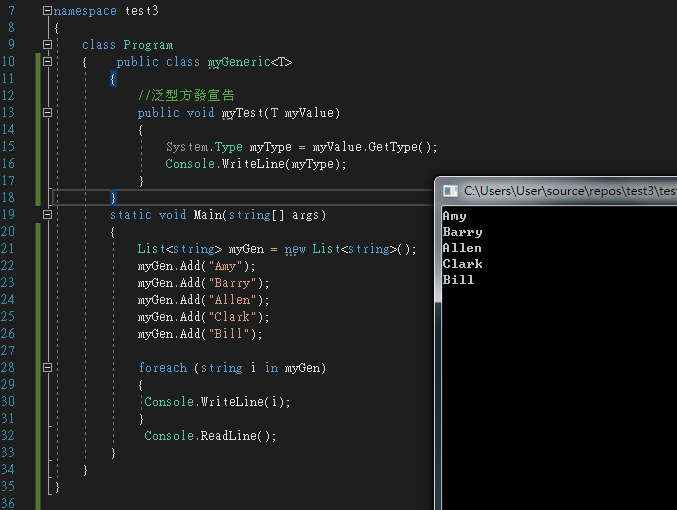
1. 什麼是泛型 (Generic Types)，特性有哪些？請示著宣告一個可為Null的實值型別變數。(10%)

 泛型將型別參數的概念引進 .NET Framework 中，使得類別和方法在設計時，可以先行擱置一或多個類型的規格，直到用戶端程式碼對類別或方法進行宣告或具現化時再行處理。

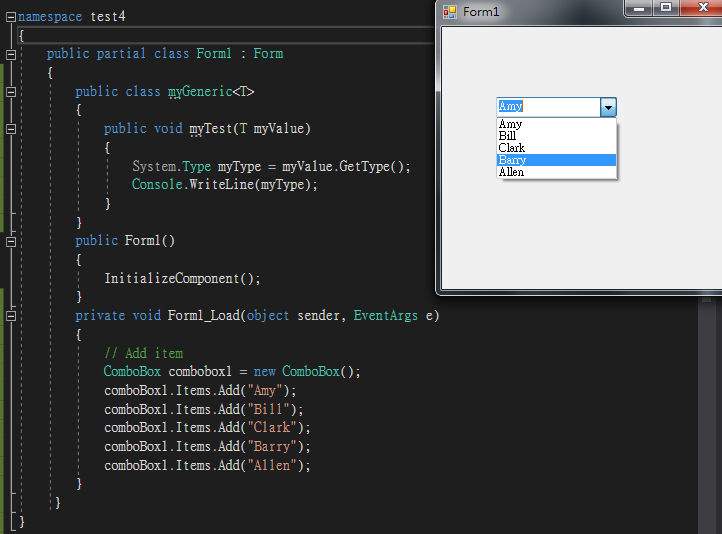
特性可以參考開放式泛型型別(未獲得任何型別參數泛型型別)以及封閉式建構泛型型別(為所有型別參數提供引數)



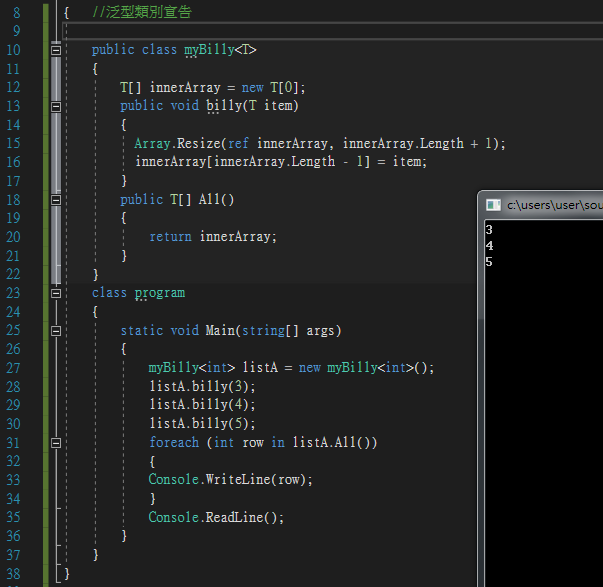
1. 承1，請試著使用.NET Framework既有泛型，使用Console.WriteLine(string s)，列出任意自建字串清單內容(例如英文人名清單)。(10%)



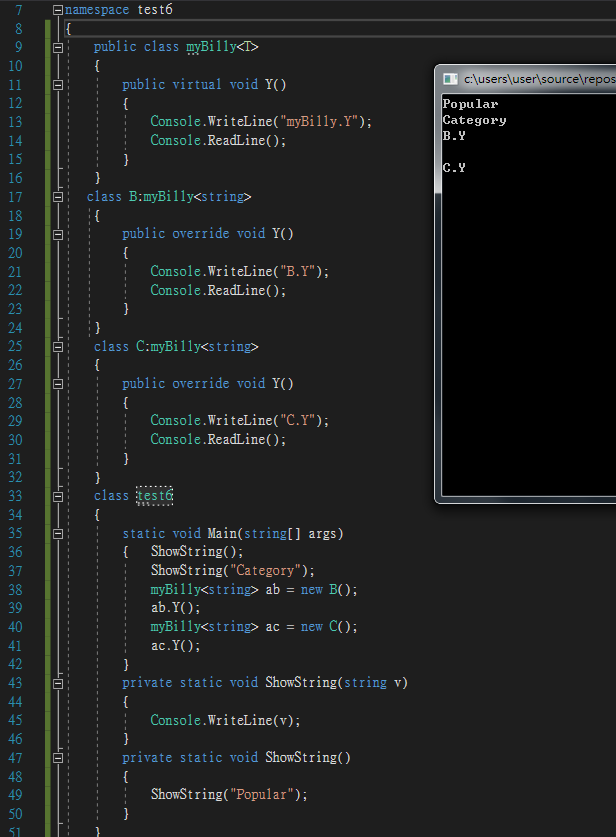
1. 承2，請改寫為WPF、Windows Forms或是ASP.NET/C#(擇一)，使用清單類型的控制項(Control)，顯示該字串清單。(20%)



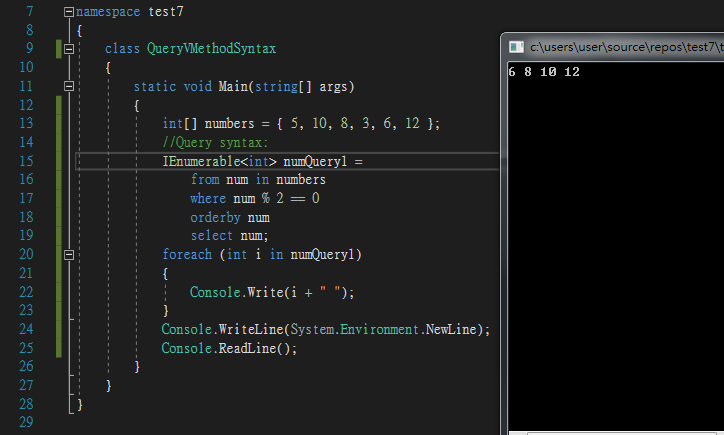
1. 請示著自行定義一個泛型類別(Class)及包含至少一個泛型方法(Method)，並使用該自定義類別宣告並初始化一個物件。(15%)



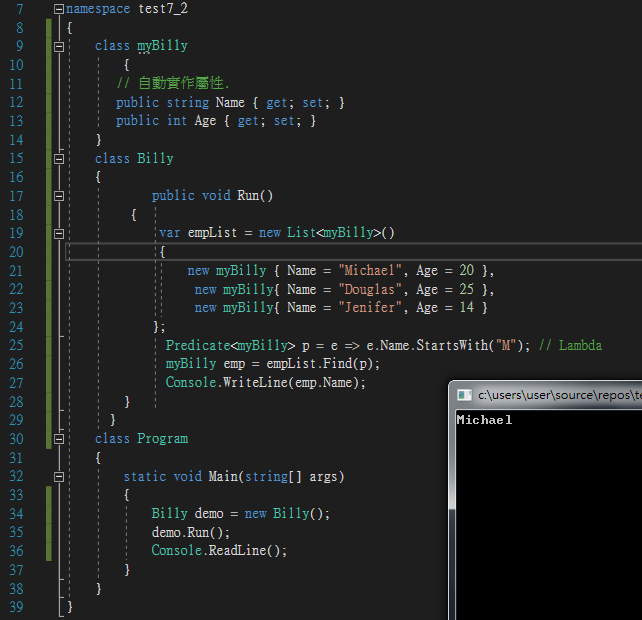
1. 承5，請繼承該自定義類別，覆寫(overriding)至少一個方法(提示：virtual、override)，加入一個重載(overloading)方法；並加上Property及Field混搭應用。(20%)



1. 加分題，試著寫一個Linq的語法來查詢泛型(Generic Types)：(25%)
2. 使用Query Expressions寫一個範例



1. 使用Lambda寫一個範例



1. 加分題，3至7題可以有意義的方式混合應用，即物件導向概念應用。(25%)