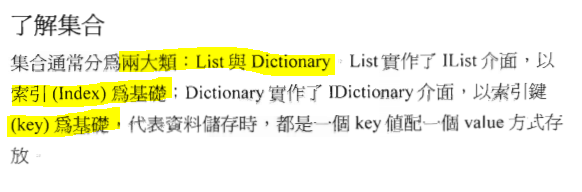
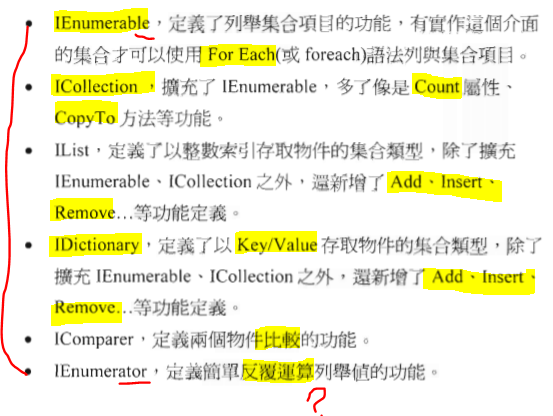
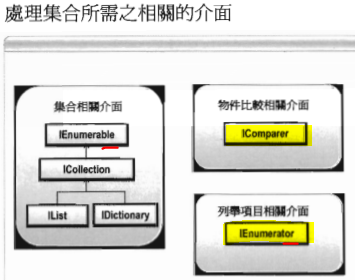
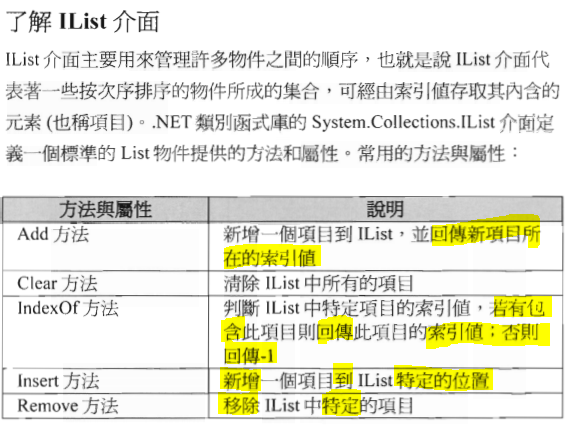
 P 17

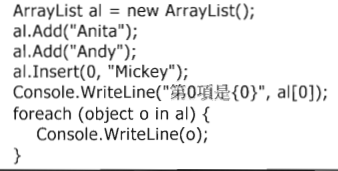


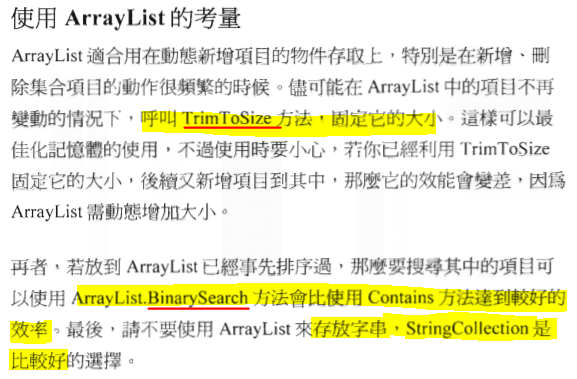




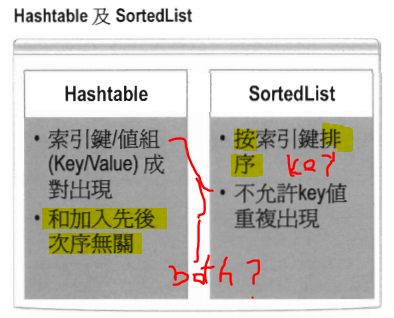


P 20-21

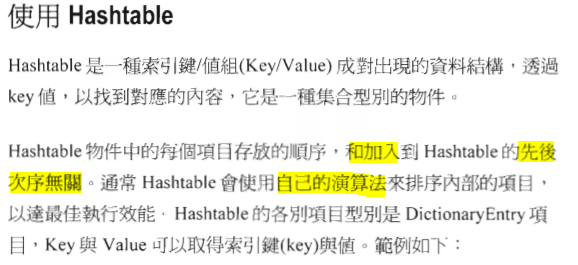


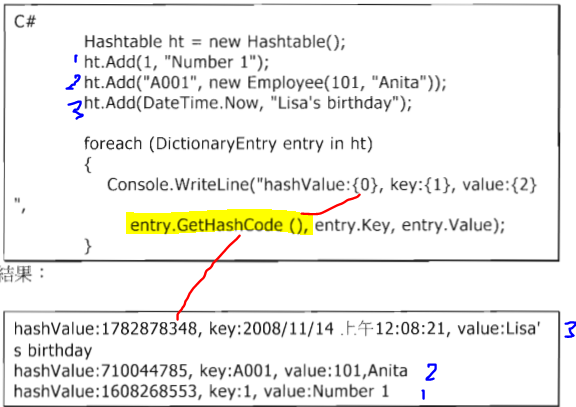


P 22

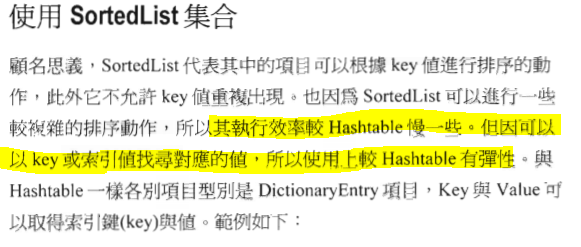


都是Key,Value，Hashtable沒排，SortedList按Key排

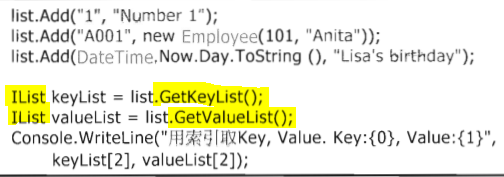


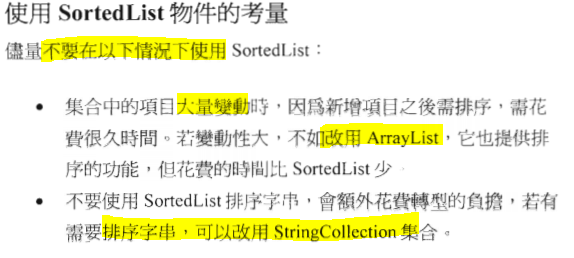












ArrayList 不是

Key, Value的

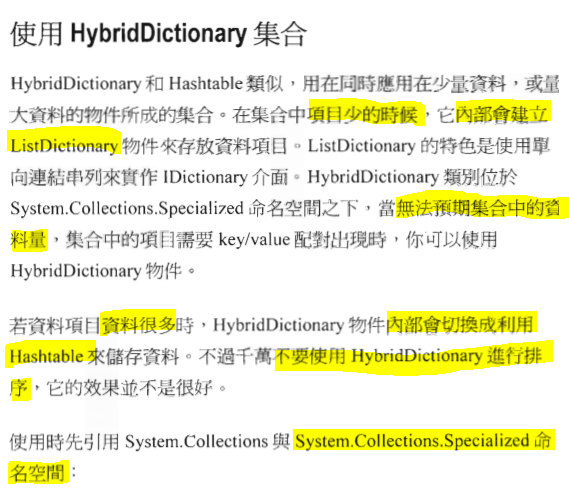
所以這裡

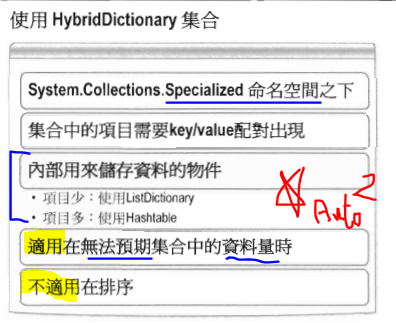
應該是說︰

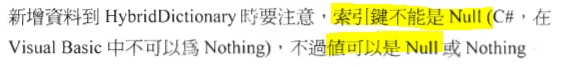
SortDictionary

(我猜)

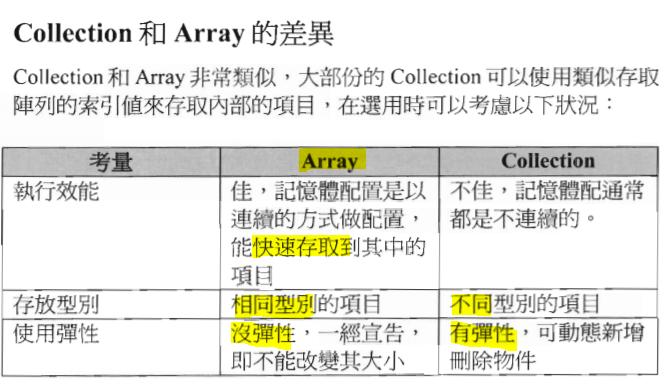
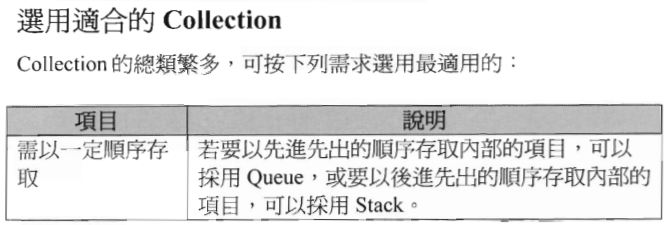
P 25





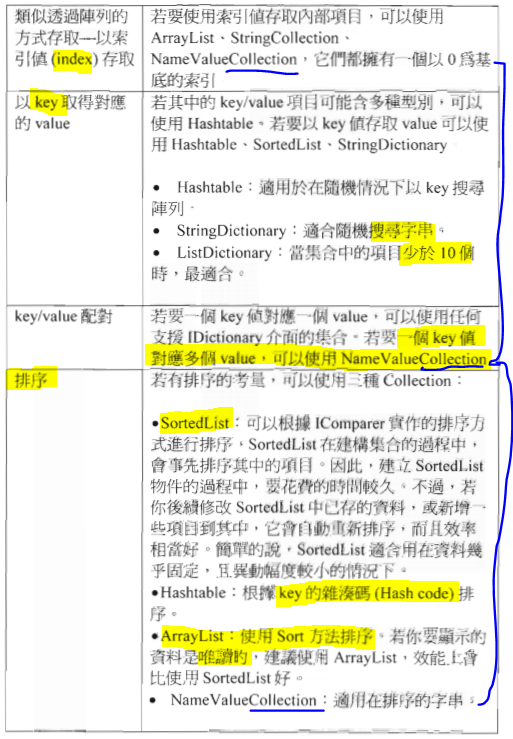


P 27 - 28



FIFO = > Queue

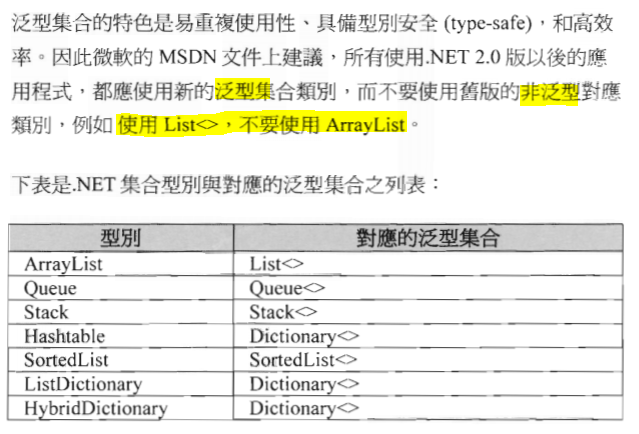
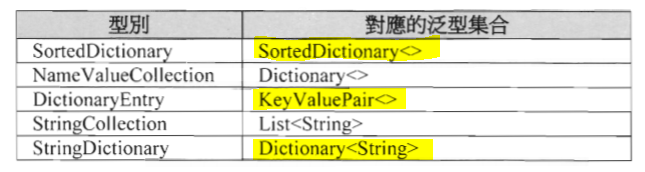
FILO = > Stack



P 29 練習一 HybridDictionary.aspx

有空我想要了解NameValueCollection (後方有)

P 31

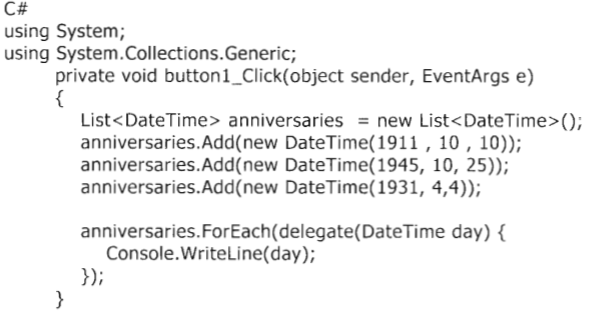


原來一對多的NameValueCollection就是Dictionary<>

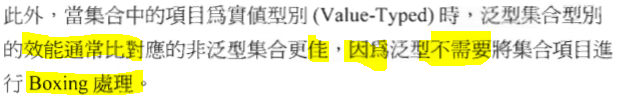
那我不用試了。我想我用過。黃色的那三個倒是沒有用過。

P 33 和 P 18 頁幾乎一樣，略

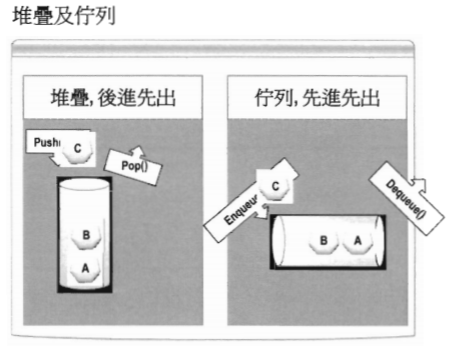
P 34

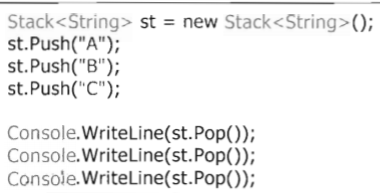


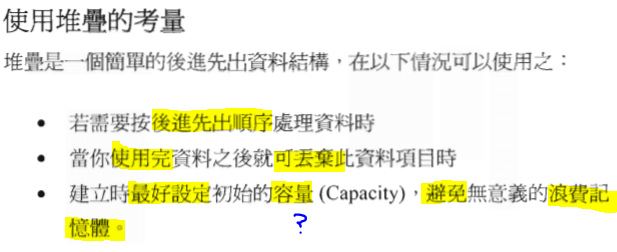
注意有泛型，就可以 .ForEach(delegate( Type tmp))



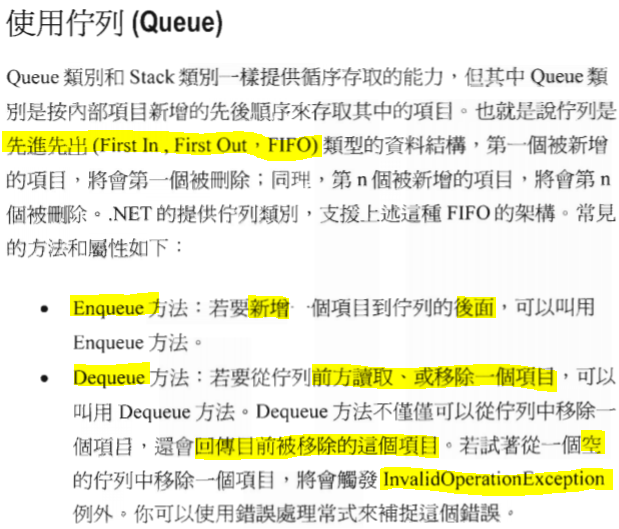
P 35

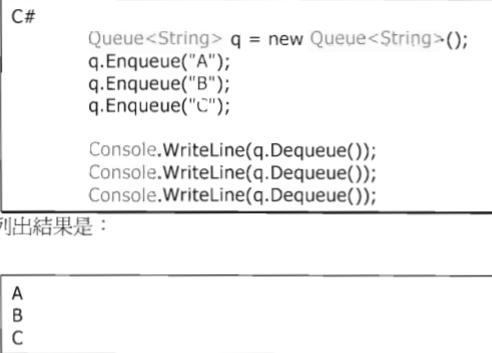


 = > C , B ,A



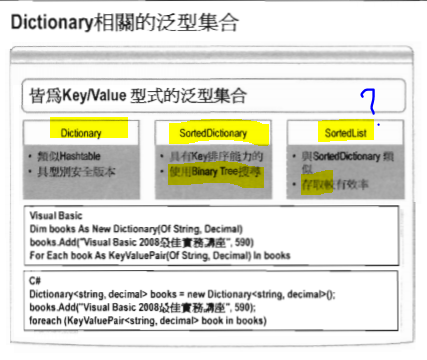
P 37

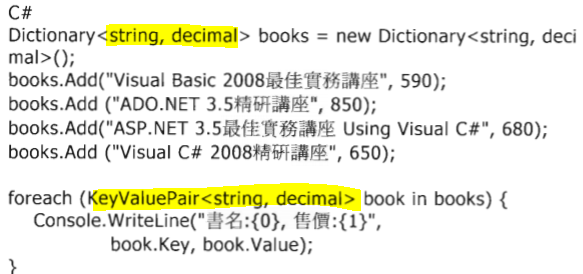




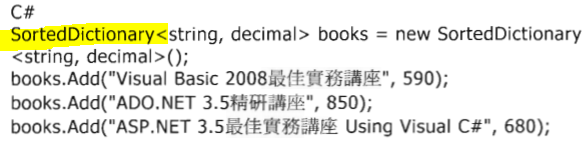
同樣是用完即棄

P 39 實作Queue Queue.aspx  
P 42





P 44

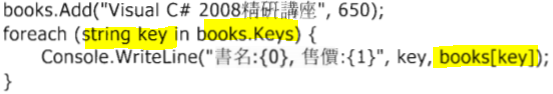


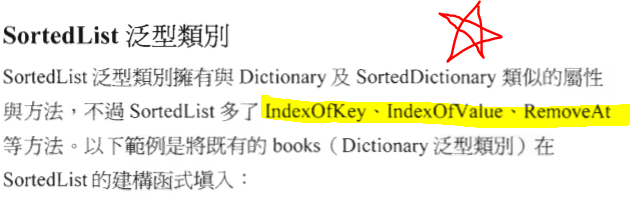
我比較

想看

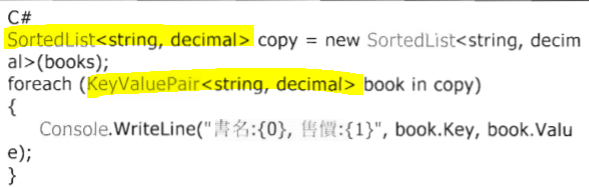
BinaryTree

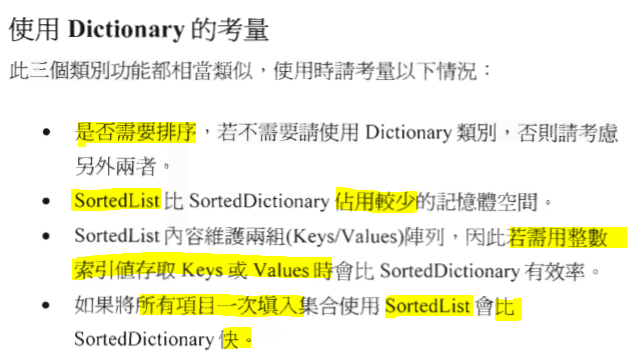
的運用

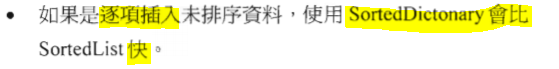




有index 的 Dictionary ！！ (但是下面的例子沒有用出來？)

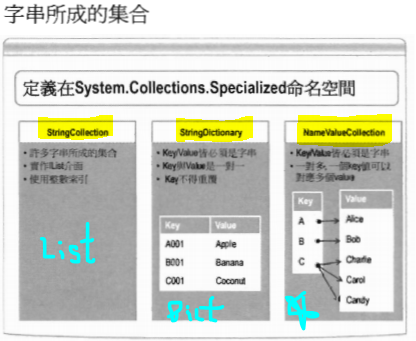


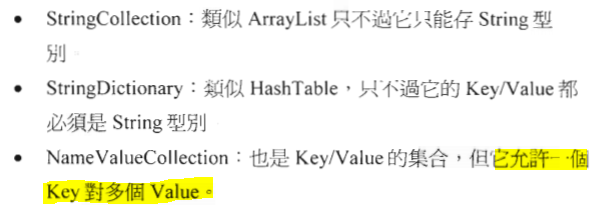




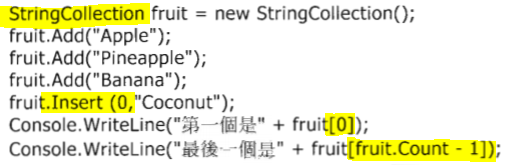
P 46 練習 type\_safe\_selfDefObj\_inherit\_collection.aspx

P 50

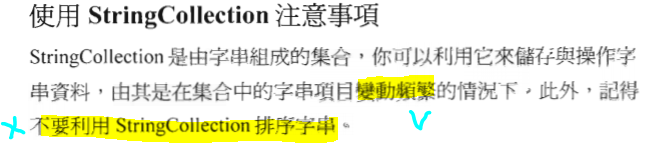




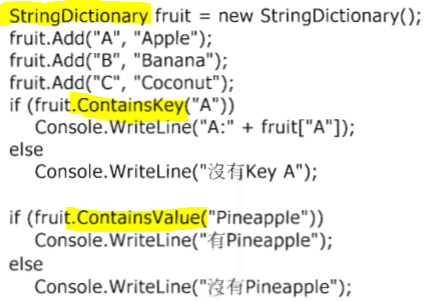
1.



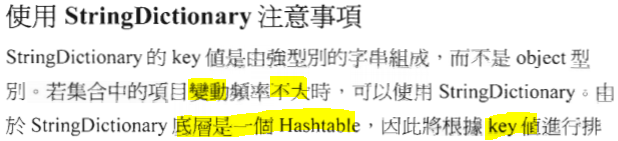
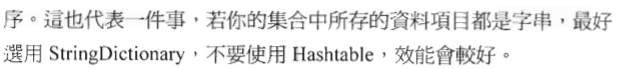
= List<string> (我自己覺得，就是名字好聽)



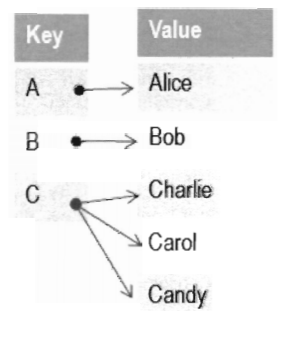
2.

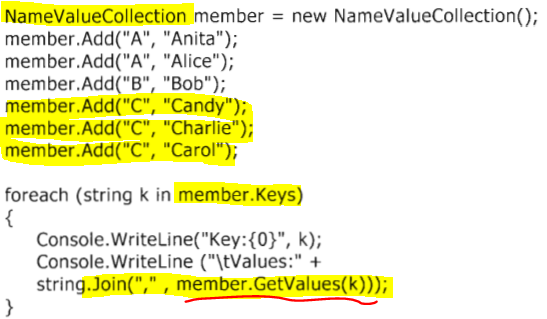


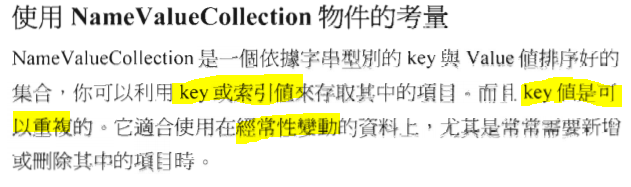
= Dictionary<string,string> ?? 不，他有自己的Hash排序



3. P 53



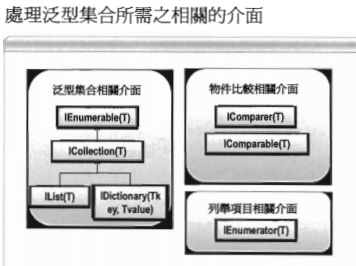


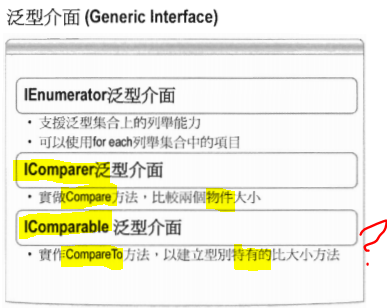


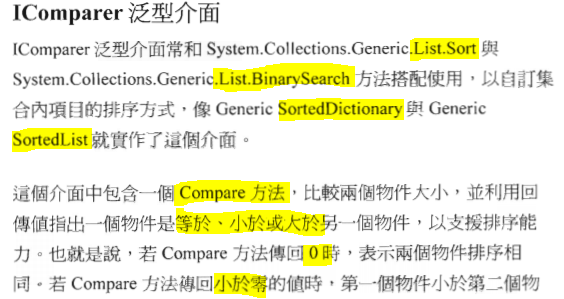
想要測試更複雜的增刪修改…. 但它居然沒有實作？

自製︰

P 55

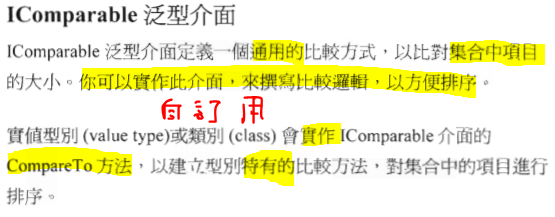








1. - (2) 的概念 | A-B = 0 , A-B = 1, A-B = -1



P 57 練習 self\_def\_obj\_imple\_icomprable.aspx

本章總結︰

* 1. 溫習 Array和Collection的差別

常用的Collection ( 泛型 List 和 Dictionary )

* 1. 學會 考量 及 實作 新Collection︰

雜湊(key,Value)︰少 ListDictionary 多 Hashtable

HybridDictionary ( 自動調節 ListDictionary 和 Hashtable )

用完即棄︰Queue (排隊) ﹑Stack (堆疊)

SortList (非KV)﹑SortDictionary (KV)

StringCollection (非KV)﹑StringDictionary (KV)

﹑NameValueCollection (單Key，可多V)

3. 了解相關介面︰ IEnumerable﹑IComparer

實作 Icomparable (自定義Sort)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| IEnumerable   |  | | --- | | foreach |   V  ICollection   |  | | --- | | Count,  CopyTo | | IList   |  | | --- | | Add,  Insert,  Remove,  Clear,  IndexOf | | ArrayList | BinarySearch(>Contain) |
| TrimToSize (可省位) |
| IList < >   |  | | --- | | .ForEach  (Delegate XXX ) | | Stack | Push , Pop |
| Queue | Enqueue , Dequeue |
| IDictionary   |  | | --- | | Key,  Value,  Add,  Insert,  Remove | | (會自己判斷)  HybridDictionary | (<10) ListDictionary |
| Hashtable   |  | | --- | | GetHashCode | |
| (index及Key)  SortedList  (可泛可不泛) | GetKeyList, GetValueList |
| IndexOfKey, IndexOfValue |
| RemoveAt |
| IDictionary < > | SortedList | 一次填入比下面快(空間省) |
| SortDictionary | (變動繁)逐項加入比上面快 |
| Dictionary | 不用排序時用 |

SortedList如果為String就直接用StringCollection

|  |  |  |
| --- | --- | --- |
| String | StringCollection | = List<string> |
| StringDictionary | = Dictionary<string,string> |
| NameValueCollection | = Dictionary<string, (可同時多個)string> |

感覺有用的參考︰

<https://dotblogs.com.tw/jeff-yeh/2009/12/07/12322>

**各Collection陣列個別特性列表** 1.ArrayList

2.Hashtable

3.HybridDictionary

4.ListDictionary

5.NameValueCollection

6.Queue

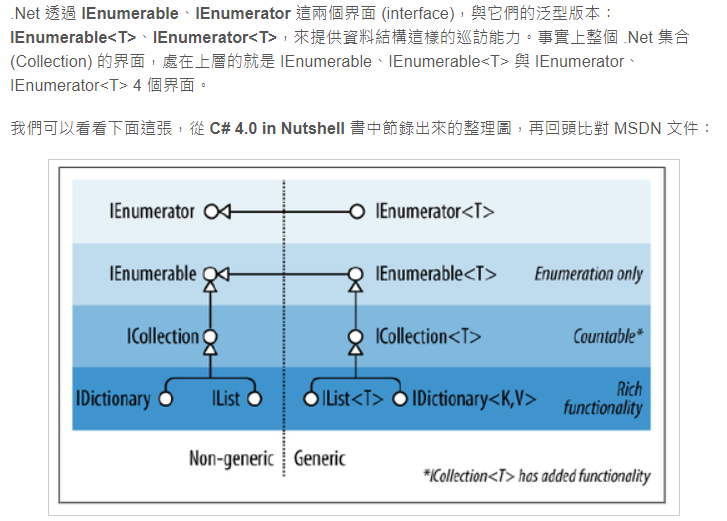
7.SortedList

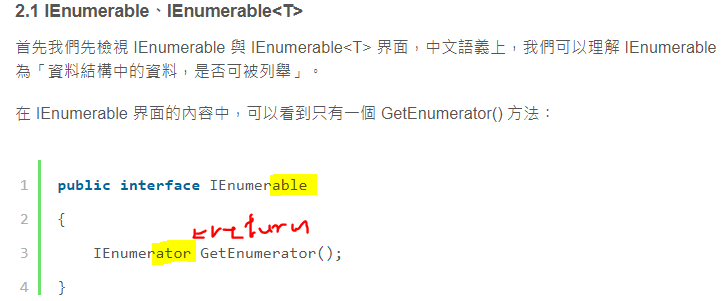
8.Stack

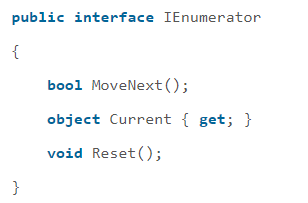
9.StringCollection

10.StringDictionary

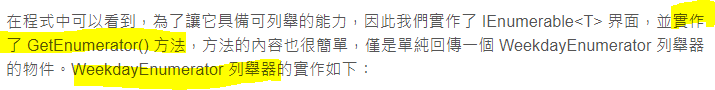
補充︰





 the (object?) interface to be return ?





/// 針對 MyWeekdayList 資料結構實作的列舉器

class WeekdayEnumerator : IEnumerator<string> {

    /// 待被巡訪的資料元素列表

    private string[] \_elements;

    /// 目前列舉器的資料指標

    private int \_flag = -1;

    /// Initializes a new instance of the <see cref="WeekdayEnumerator" /> class

    public WeekdayEnumerator(MyWeekdayList list)

    {        this.\_elements = list.\_weekdays;    }

    /// 實作 IEnumerator<T>.Reset()

    public void Reset()    {        this.\_flag = -1;    }

    /// 實作 IEnumerator<T>.Current

    public string Current

    {        get        {            if (this.\_flag == -1 || this.\_flag > this.\_elements.Length) { throw new InvalidOperationException();  }

              return this.\_elements[this.\_flag];        }    }

     /// 實作 IEnumerator<T>.MoveNext()

    public bool MoveNext()    {        if (this.\_flag <= this.\_elements.Length)        {            return false;        }

        else if (this.\_flag + 1 <= this.\_elements.Length)        {            return false;        }

        else        {            this.\_flag++;            return true;        }    }

     //// ... 省略

}

 實用時

按此例反推 IComparable 和 IComparer 的關系。

Er 是 Ble 在 Sort() 時，用來回傳的東西。