Conjunto

Se llama conjunto a una estructura de datos que se comporta como la entidad matemática del mismo nombre. O sea, en un conjunto los elementos se almacenan sin que exista alguna relación de orden entre ellos y garantizando que no pueda tener elementos repetidos.

Programe la clase siguiente

```
namespace Weboo.Programming.ThirdRound
{
	public class Set : Weboo.Collections.Set.ISet
	{
		public Set()
		{
			… … … … …
		}
		… … … … …
	}
}
```

La interfaz `Weboo.Collections.Set.ISet` (implementada por la clase `Weboo.Programming.ThirdRound.Set`) se brinda en la biblioteca de clases `Weboo.Collections.Set.dll` y se define tal y como se muestra a continuación.

```
namespace Weboo.Collections.Set
{
	public interface ISet : System.Collections.IEnumerable
	{
		//Permite conocer la cantidad de elementos
		//almacenados en el conjunto
		int Count {get;}

		//Añade un  elemento al conjunto.
		//Si el elemento ya se encuentra en el conjunto no
		//se hace nada
		//Si se intenta añadir el elemento null al conjunto
		//el método debe lanzar ArgumentNullException
		void Add(object o);

		//Quita el elemento del conjunto
		//Si no está no se hace nada
		void Remove(object o);

		//Devuelve true si el elemento o
		//se encuentra en el conjunto y
		//false en caso contrario
		bool Contains(object o);

		//Devuelve un nuevo conjunto que resulta
		//de realizar la unión de este conjunto (this)
		//con el conjunto other. Ni este conjunto (this)
```

```
            //ni el conjunto other son afectados por esta operación.
            //Si el valor de other es null el método debe
            //lanzar la excepción ArgumentNullException
            ISet Union(ISet other);

            //Devuelve un nuevo conjunto que resulta
            //de realizar la intersección de este conjunto (this)
            //con el conjunto other. Ni este conjunto (this)
            //ni el conjunto other son afectados por esta operación.
            //Si el valor de other es null el método debe
            //lanzar la excepción ArgumentNullException
            ISet Intersection(ISet other);
      }
}
```

Note que la interfaz `Weboo.Collections.Set.ISet` implementa la interfaz `System.Collections.IEnumerable`. Para la clase que usted debe programar, o sea la clase `Weboo.Programming.ThirdRound.Set`, el enumerador asociado itera sobre todos los elementos del conjunto en cualquier orden. Recuerde que si ocurren cambios en el conjunto (se adicionan o se quitan elementos) todos los enumeradores obtenidos antes de la ocurrencia de dichos cambios deben quedar inválidos, o sea, al ejecutar `MoveNext()` o `Reset()` se debe lanzar `InvalidOperationException` y al ejecutar `Current` se debe obtener el mismo resultado que se hubiera obtenido si no se hubiera modificado la colección. Para más información puede consultar en la ayuda la especificación de los métodos de la interfaz `System.Collections.IEnumerator`.

Nota: En su implementación usted NO PUEDE utilizar las siguientes clases e interfaces ni herederos de las mismas:

- `System.Collections.ArrayList.`
- `System.Collections.CollectionBase`
- `System.Collections.IDictionary`

Ejemplos.

```
Set s = new Set();

s.Add("a");
s.Add("z");
s.Add(5);
s.Add("h");

Console.WriteLine(s.Contains(5));          Aparece en la consola: true
Console.WriteLine(s.Contains("k"));        Aparece en la consola: false
DateTime dt = new DateTime(4, 5, 2005);
Console.WriteLine(s.Contains(dt));         Aparece en la consola: false;

foreach(object o in s)                     Aparece en la consola (no
      Console.Write(o + " ");              precisamente en éste orden):
                                                 a z 5 h

Console.WriteLine(s.Count);                Aparece en la consola: 4
```

```
s.Add("a");
Console.WriteLine(s.Count);              Aparece en la consola: 4
s.Remove("i");
Console.WriteLine(s.Count);              Aparece en la consola: 4

Set other = new Set();

other.Add("k");
other.Add(5);
other.Add("z");
other.Add("m");
other.Add("l");

ISet union = s.Union(other);

foreach(object o in union)               Aparece en la consola (no
      Console.Write(o + " ");            precisamente en este orden):
                                               a z 5 h k m l
foreach(object o in s)                   Aparece en la consola (no
      Console.Write(o + " ");            precisamente en éste orden):
                                               a z 5 h
foreach(object o in other)               Aparece en la consola (no
      Console.Write(o + " ");            precisamente en éste orden):
                                               k 5 z m l

ISet intersection = s.Intersection(other);

foreach(object o in intersection)        Aparece en la consola (no
      Console.Write(o + " ");            precisamente en este orden):
                                               5 z
foreach(object o in s)                   Aparece en la consola (no
      Console.Write(o + " ");            precisamente en éste orden):
                                               a z 5 h
foreach(object o in other)               Aparece en la consola (no
      Console.Write(o + " ");            precisamente en éste orden):
                                               k 5 z m l

IEnumerator e = s.GetEnumerator();

e.MoveNext();
Console.WriteLine(e.Current);            Aparece en la pantalla algún
                                         elemento de s, por ejemplo:
                                               a
e.MoveNext();
Console.WriteLine(e.Current);            Aparece en la pantalla otro
                                         elemento de s, por ejemplo:
                                               z
s.Remove("h");

Console.WriteLine(e.Current);            Aparece en la pantalla el
                                         mismo elemento de s que se
                                         obtuvo al hacer e.Current la
                                         vez anterior, o sea:
                                               z
e.MoveNext();                            Lanza
                                         InvalidOperationException
```

```
                                                porque la colección ha sido
                                                modificada
e.Reset();                                      Lanza
                                                InvalidOperationException
                                                porque la colección ha sido
                                                modificada
```