COLECCIONES

```
public interface IList<T>
{
- FUNCIONES
int GetIndexOf(T element)
int GetListCount()
T GetElementAt(int index)

void AddElement(T element)
void RemoveElement(T element)
void RemoveElementAt(int index)

bool Contains(T element)

bool IsEmpty()
bool IsSort()
bool IsValid()

void Sort()
void Filter()
void Visit()

void Clear()
List<T> Clone()
}
```

```
public interface IDictionary<K,V>
{
- ATRIBUTOS
private Item[] _item;

- PROPERTIES
public int Count
public bool IsEmpty

- FUNCIONES
int GetIndexOf(V value)
V GetElementAt(K key)

void AddElement(K key, V value)
void RemoveElementAt(int index)

bool Contains(V value)

bool Equals(object obj)
bool AreIdentical(object obj)
int GetHashCode()

bool IsValid()

void Sort()
void Filter()
void Visit()

void Clear()
string ToString()
}
```

```
public class Stack<T>
{
- ATRIBUTOS
private T[] _stack;

- PROPERTIES
public bool IsEmpty
public int Count

- CONSTRUCTORES
public Stack()

- FUNCIONES
public void Push(T element)
public T Pop()
public T Top()

public T[] Clone()
public void Clear()
public override string ToString()
}
```

```
public class Queue<T>
{
- ATRIBUTOS
private T[] _queue;

- PROPERTIES
public bool IsEmpty
public int Count
public T First
public T Last

- CONSTRUCTORES
public Queue()

- FUNCIONES
public void Enqueue(T element)
public T Dequeue()

public void QueueMultipleElements(T[] elements)
public T[] Clone()
public void Clear()
public override string ToString()
}
```

```
public class Set<T>
{
- ATRIBUTOS
private T[] _stack;

- PROPERTIES
public bool IsEmpty
public int Count

- CONSTRUCTORES
public Stack()

- FUNCIONES
public void Push(T element)
public T Pop()
public T Top()

public void Clear()
public T[] Clone()
public override string ToString()

}
```

```
public class HashSet<T>
{
- ATRIBUTOS
private T[] _stack;

- PROPERTIES
public bool IsEmpty
public int Count

- CONSTRUCTORES
public Stack()

- FUNCIONES
public void Push(T element)
public T Pop()
public T Top()
public void Clear()
public override string ToString()
}
```

public class **ItemSet<T>**
{
- ATRIBUTOS
private T[] _queue;

- PROPERTIES
public bool **IsEmpty**
public int **Count**
public T **First**
public T **Last**

- CONSTRUCTORES
public Queue()

- FUNCIONES
public void **Enqueue**(T element)
public T **Dequeue**()
public T[] **Clone**(T[] queue)
public void **QueueMultipleElements**(T[] elements)
public void **Clear**()
public override string **ToString**()
}

public class **SortSet<T>**
{
- ATRIBUTOS
private T[] _queue;

- PROPERTIES
public bool **IsEmpty**
public int **Count**
public T **First**
public T **Last**

- CONSTRUCTORES
public Queue()

- FUNCIONES
public void **Enqueue**(T element)
public T **Dequeue**()
public T[] **Clone**(T[] queue)
public void **QueueMultipleElements**(T[] elements)
public void **Clear**()
public override string **ToString**()
}

```
public class Tree<T>
{
}
```

```
public class TreeWeak<T>
{
}
```