HashSet<T> e SortedSet<T>

- Representa um conjunto de elementos (similar ao da Álgebra)
  - Não admite repetições
  - Elementos não possuem posição
  - Acesso, inserção e remoção de elementos são rápidos
  - Oferece operações eficientes de conjunto: interseção, união, diferença.
- HashSet
  - https://msdn.microsoft.com/en-us/library/bb359438(v=vs.110).aspx
- SortedSet
  - https://msdn.microsoft.com/en-us/library/dd412070(v=vs.110).aspx

Diferenças

- HashSet
  - Armazenamento em tabela hash
  - Extremamente rápido: inserção, remoção e busca O(1)
  - A ordem dos elementos não é garantida
- SortedSet
  - Armazenamento em árvore
  - Rápido: inserção, remoção e busca O(log(n))
  - Os elementos são armazenados ordenadamente conforme implementação IComparer<T>

Alguns métodos importantes

- Add
- Clear
- Contains
- UnionWith(other) - operação união: adiciona no conjunto os elementos do outro conjunto, sem repetição
- IntersectWith(other) - operação interseção: remove do conjunto os elementos não contidos em other
- ExceptWith(other) - operação diferença: remove do conjunto os elementos contidos em other
- Remove(T)
- RemoveWhere(predicate)

Demo 1

```
using System;
using System.Collections.Generic;

namespace Course {
    class Program {

        static void Main(string[] args) {
            HashSet<string> set = new HashSet<string>();

            set.Add("TV");
            set.Add("Notebook");
            set.Add("Tablet");

            Console.WriteLine(set.Contains("Notebook"));

            foreach (String p in set) {
                Console.WriteLine(p);
            }
        }
    }
}
```

Demo 2

```
using System;
using System.Collections.Generic;

namespace Course {
    class Program {

        static void Main(string[] args) {
            SortedSet<int> a = new SortedSet<int>() { 0, 2, 4, 5, 6, 8, 10 };
            SortedSet<int> b = new SortedSet<int>() { 5, 6, 7, 8, 9, 10 };

            //union
            SortedSet<int> c = new SortedSet<int>(a);
            c.UnionWith(b);
            printCollection(c);

            //intersection
            SortedSet<int> d = new SortedSet<int>(a);
            d.IntersectWith(b);
            printCollection(d);

            //difference
            SortedSet<int> e = new SortedSet<int>(a);
            e.ExceptWith(b);
            printCollection(e);
        }

        static void printCollection<T>(IEnumerable<T> collection) {
            foreach(T obj in collection) {
                Console.Write(obj + " ");
            }
            Console.WriteLine();
        }
    }
}
```

Como as coleções Hash testam igualdade?

http://educandoweb.com.br

Prof. Dr. Nelio Alves