Set

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Set<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.
  - Principais implementações:
    - **HashSet** - mais rápido (operações O(1) em tabela hash) e não ordenado
    - **TreeSet** - mais lento (operações O(log(n)) em árvore rubro-negra) e ordenado pelo compareTo do objeto (ou Comparator)
    - **LinkedHashSet** - velocidade intermediária e elementos na ordem em que são adicionados

- https://docs.oracle.com/javase/10/docs/api/java/util/Set.html

Alguns métodos importantes

- add(obj), remove(obj), contains(obj)
  - Baseado em equals e hashCode
  - Se equals e hashCode não existir, é usada comparação de ponteiros

- clear()
- size()
- removeIf(predicate)

- addAll(other) - **união**: adiciona no conjunto os elementos do outro conjunto, sem repetição
- retainAll(other) - **interseção**: remove do conjunto os elementos não contitos em other
- removeAll(other) - **diferença**: remove do conjunto os elementos contidos em other

Demo 1

```
package application;

import java.util.HashSet;
import java.util.Set;

import Entities.Product;

public class Program {

	public static void main(String[] args) {

		Set<String> set = new HashSet<>();

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

		System.out.println(set.contains("Notebook"));

		for (String p : set) {
			System.out.println(p);
		}
	}
}
```

Demo 2

```
package application;

import java.util.Arrays;
import java.util.Set;
import java.util.TreeSet;

public class Program {

	public static void main(String[] args) {

		Set<Integer> a = new TreeSet<>(Arrays.asList(0,2,4,5,6,8,10));
		Set<Integer> b = new TreeSet<>(Arrays.asList(5,6,7,8,9,10));

		//union
		Set<Integer> c = new TreeSet<>(a);
		c.addAll(b);
		System.out.println(c);

		//intersection
		Set<Integer> d = new TreeSet<>(a);
		d.retainAll(b);
		System.out.println(d);

		//difference
		Set<Integer> e = new TreeSet<>(a);
		e.removeAll(b);
		System.out.println(e);
	}
}
```