JAC444 - BTP400 Course Object-Oriented Software Development II - Java

Collections

Segment 2

The Set Interface

The Set Interface

Set is a collection that **cannot contain duplicate elements**

Set interface inherits from **Collection** adds the restrictions to eliminate the duplicate elopements

Implementations:

HashSet - stores its elements in a hash table and is the best-performing implementation

TreeSet - The elements are ordered using their natural ordering

LinkedHashSet — Hash table with linked list running trough it

The Set<E> Interface

```java
public interface Set<E> {
        // Group 1
        int size();
        boolean isEmpty();
        boolean contains(Object element);
        boolean add(E element);
        boolean remove(E element);
        Iterator<E> iterator();

        // Group 2
        boolean containsAll(Collection<?> c);
        boolean addAll(Collection<?>  c);
        boolean removeAll(Collection<?>  c);
        boolean retainAll(Collection<?>  c);
        void clear();

        // Group 3
        Object[] toArray();
        <T> T[] toArray(T[] a)

    }
```

1 Basic Operations

2 Bulk Operations

3 Array Operations

Basic Operations

```java
import java.util.*;

public class FindDups {

  public static void main(String args[]) {
    Set<String> s = new HashSet<>();
    for (int i=0; i < args.length; i++)
      if (!s.add(args[i]))
        System.out.println("Duplicate: "+args[i]);

    System.out.println(s.size()+ " distinct : "+s);
  }
}
```
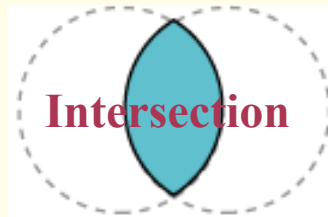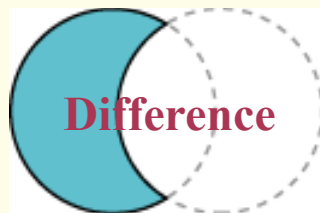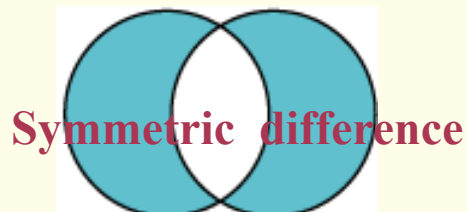
Bulk Operations

Union

```
Set union = new HashSet(s1);
union.addAll(s2);
```


Intersection

```
Set intersection = new HashSet(s1);
intersection.retainAll(s2);
```


Difference

```
Set difference = new HashSet(s1);
difference.removeAll(s2);
```

```
Set symmetricDiff = new HashSet(s1);
symmetricDiff.addAll(s2);
```


Symmetric difference

```
Set tmp = new HashSet(s1);
tmp.retainAll(s2);
symmetricDiff.removeAll(tmp);
```

Duplicate Words Sample

```java
import java.util.*;

public class FindDuplicateWords {
  public static void main(String args[]) {
    Set<String> uniques = new HashSet<>();
    Set<String>  dups = new HashSet<>();

    for (int i=0; i < args.length; i++)
      if (!uniques.add(args[i]))
        dups.add(args[i]);

    uniques.removeAll(dups); //Destructive set-difference

    System.out.println("Unique: " + uniques);
    System.out.println("Duplicate: " + dups);
  }
}
```