JAVA

Class 32

Agenda

Set Interface and HashSet Class

Task

1. Create a class Insurance that will have an attribute as insuranceName and unimplemented behaviour as getQuote and cancelInsurance. Create 3 subclasses Car, Pet, Health. Car class has it's own attribute as carModel and Class Pet has petType attribute.
- Create 3 objects of the sub classes and store them in ArrayList. Using for loop/advanced for loop/ iterator access methods from different classes.

Set

Set is a Collection that can’t contain duplicate elements.

There are three main implementations of Set interface:

- HashSet
- LinkedHashSet
- TreeSet

Set

HashSet, which stores its elements in a hash table, is the best-performing implementation.

TreeSet, which stores its elements in a red-black tree, orders its elements based on their values; it is substantially slower than HashSet.

LinkedHashSet, which is implemented as a hash table with a linked list running through it, orders its elements based on the order in which they were inserted into the set.

Hashset in Java

- **HashSet** is the class, Which implements the Set interface in Java.
- The HashSet does not add any additional methods beyond those found in the Set interface.
- **HashSet does not guarantee any insertion orders of the set** but it allows null elements.
- HashSet can be used in place of ArrayList to store the object if you require no duplicate and don’t care about insertion order.
- HashSet doesn’t allow duplicates. If you try to add a duplicate element in HashSet, the old value would be overwritten.

Methods in Hashset

- **boolean add(Element e)**: It adds the element e to the list.
- **void clear():** It removes all the elements from the list.
- **boolean contains(Object o):** It checks whether the specified Object o is present in the list or not. If the object has been found it returns true else false.
- **boolean isEmpty():** Returns true if there is no element present in the Set.
- **Iterator iterator():** Used to return an iterator over the element in the set.
- **int size():** It gives the number of elements of a Set.
- **Boolean remove (Object o):** It removes the specified Object from the Set.
- **Object clone():** This method returns a shallow copy of the HashSet.

NOTE: A Set doesn't provide any method for data retrieval.

```
import java.util.HashSet;
public class HashSetExample {
   public static void main(String args[]) {
HashSet<String> hset = new
HashSet<String>();

     // Adding elements to the HashSet
     hset.add("Apple");
     hset.add("Mango");
     hset.add("Grapes");
     hset.add("Orange");
     hset.add("Fig");
     //Addition of duplicate elements
     hset.add("Apple");
     hset.add("Mango");
     //Addition of null values
     hset.add(null);
     hset.add(null);

     //Displaying HashSet elements
     System.out.println(hset);
   }
}
```

```
import java.util.HashSet;
import java.util.Iterator;

class IterateHashSet{
  public static void main(String[] args) {
     // Create a HashSet
     HashSet<String> hset = new
HashSet<String>();

     //add elements to HashSet
     hset.add("Chaitanya");
     hset.add("Rahul");
     hset.add("Tim");
     hset.add("Rick");
     hset.add("Harry");

     Iterator<String> it = hset.iterator();
     while(it.hasNext()){
        System.out.println(it.next());
     }
  }
}
```

Task

1. Create an HashSet of cities and add duplicates into it.
- Retrieve all values from hashset in 2 different ways.
- Retrieve all values in alphabetical order.