Comparator Interface

Comparator interface is used to compare objects of user defined classes.

Suppose we have an Array / ArrayList of our own class type containing fields like rollno, name, address, DOB etc and we need to sort the Array/arrayList based on rollno or name?

METHOD 1 :

One obvious approach is to write our own sort() function using one oof the standard algorithms. This solution requires rewriting the whole sorting code for different criteria like rollno and name

METHOD 2: Using Comparator Interface

Comparator interface is used to order the objects of the user defined class. This interface is present in java.util package and contains 2 methods

```
1. compare(Object obj1, Object obj2)
2. equals(Object element)
```

Using a comparator we can sort the elements based on data members. For instance it may be on rollno, name , age or anything else.

Working

```
Collections.sort(List list, ComparatorClass c);
Arrays.sort(Array arr, ComparatorClass c);
```

- Given that List and ComparatorClass implement a comparator interface

Internally the sort method class Compare method of the classes it is sorting.

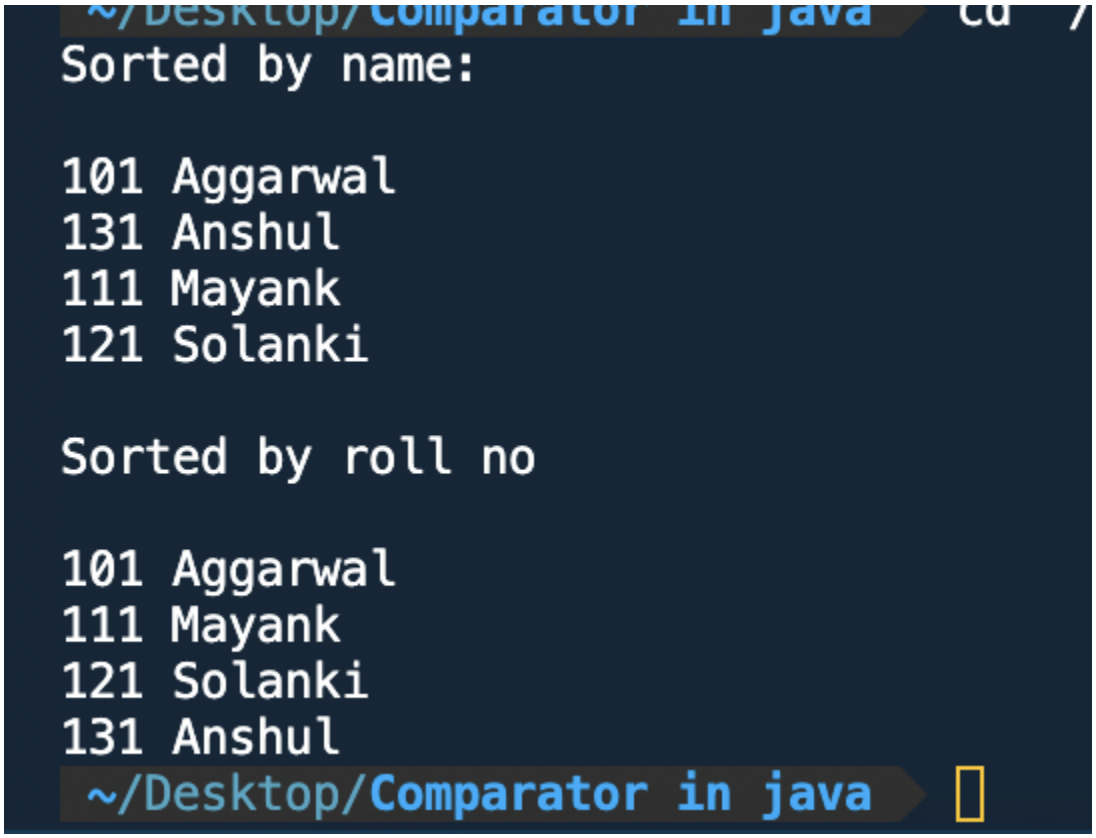
Example:

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;

class Student {

   int rollno;
   String name;

   Student(int rollno, String name){
       this.rollno = rollno;
       this.name = name;
   }

   // To print student details in main()
   public String toString()
   {

       return this.rollno + " " + this.name;
   }
```

```
}

//creating class to sort list acording rollno

class Sortbyroll implements Comparator<Student>{
   @Override
   public int compare(Student a, Student b) {
       return a.rollno - b.rollno;
   }
}

//creating class to sort list by name

class Sortbyname implements Comparator<Student>{
   @Override
   public int compare(Student a, Student b) {
       return a.name.compareTo(b.name);
   }
}


class comp {

   public static void main(String[] args) {


       ArrayList<Student> ar = new ArrayList<>();

       ar.add(new Student(111, "Mayank"));
       ar.add(new Student(131, "Anshul"));
       ar.add(new Student(121, "Solanki"));
       ar.add(new Student(101, "Aggarwal"));

       System.out.println("Sorted by name:");
       System.out.println();


       // calling the sort method with the object of the class that we
created
       Collections.sort(ar, new Sortbyname());

       for (int i = 0; i < ar.size(); i++) {
           System.out.println(ar.get(i));
```

```
        }

        // calling the sort method with the object of the class that we
 created
        Collections.sort(ar, new Sortbyroll());

        System.out.println("Sorted by roll no");
        System.out.println();

        for (int i = 0; i < ar.size(); i++) {
            System.out.println(ar.get(i));
        }

    }
}
```

Output:

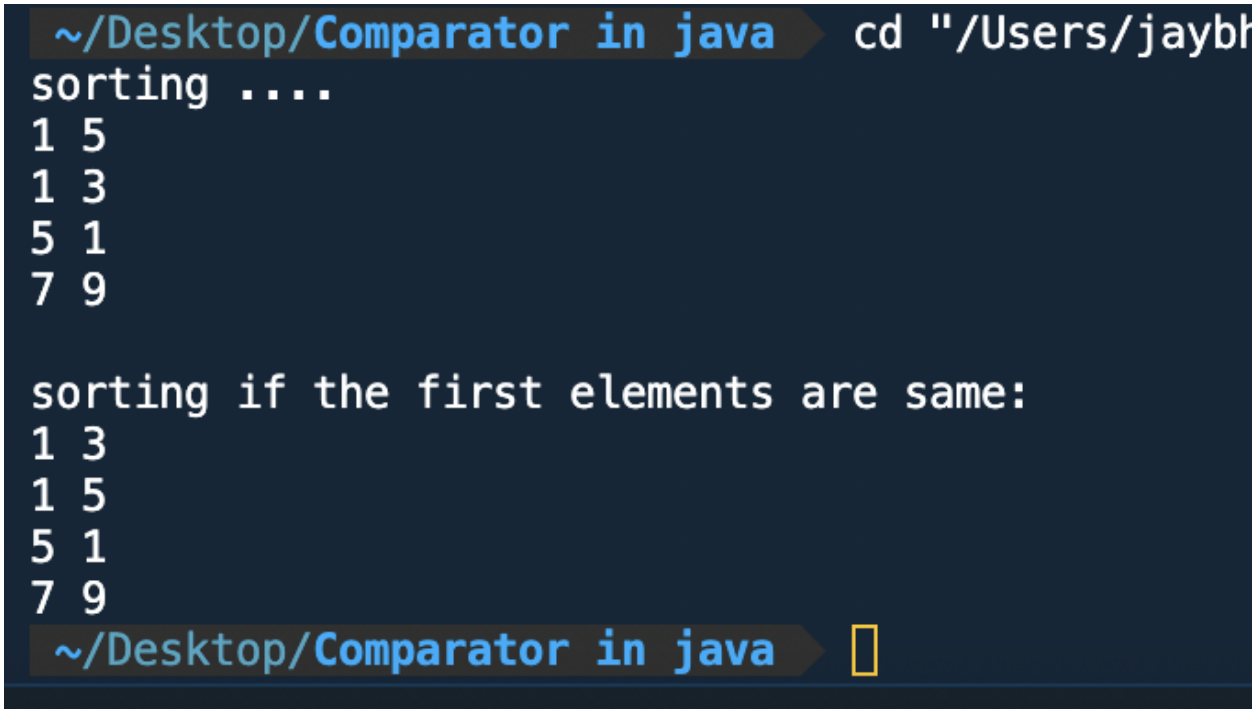


Example:

```
import java.lang.reflect.Array;
import java.util.Arrays;
import java.util.Collections;
import java.util.Comparator;

class sortarray implements Comparator<int[]>{
   @Override
   public int compare(int[] a, int[] b) {

       return a[0] - b[0];
   }
}

class Sortarray2 implements Comparator<int[]>{
   @Override
   public int compare(int[] o1, int[] o2) {
       if(o1[0]==o2[0]){
           return o1[1] - o2[1];
       }
       else{
           return o1[0] - o2[0];
       }
   }
}



public class dhoom {
   public static void main(String[] args) {

       int[][] a = {{1,5},{7,9},{5,1},{1,3}};



       Arrays.sort(a, new sortarray());

       System.out.println("sorting ....");
       for (int i = 0; i < a.length; i++) {
           for (int j = 0; j < 2; j++) {
               System.out.print(a[i][j]+" ");
```

```
            }
            System.out.println();
        }

        System.out.println();
        Arrays.sort(a, new Sortarray2());
        System.out.println("sorting if the first elements are same:");

        for (int i = 0; i < a.length; i++) {
            for (int j = 0; j < 2; j++) {
                System.out.print(a[i][j]+" ");
            }
            System.out.println();
        }
    }
}
```

Output:



📝 Note:

Sort method can also be called as follows (shorthand method)

```
int[][] a = {{1,5},{7,9},{5,1},{1,3}};

  Arrays.sort(a,new Comparator<int[]>() {
           @Override
           public int compare(int[] o1, int[] o2) {
              return o1[0] - o2[0];
           }
       });
```