

# Sorting

We can sort the elements of:

1. String objects
2. Wrapper class objects
3. User-defined class objects

**Collections** class provides static methods for sorting the elements of collection. If collection elements are of Set type, we can use TreeSet. But we cannot sort the elements of List. Collections class provides methods for sorting the elements of List type elements.

## Method of Collections class for sorting List elements

**public void sort(List list):** is used to sort the elements of List. List elements must be of Comparable type.

**Note: String class and Wrapper classes implement the Comparable interface. So if you store the objects of string or wrapper classes, it will be Comparable.**

Example of Sorting the elements of List that contain String objects

```
import java.util.*;
class TestSort1{
    public static void main(String args[]){

        ArrayList<String> al=new ArrayList<String>();
        al.add("Virus");
        al.add("Saurav");
        al.add("Mukesh");
        al.add("Tahir");

        Collections.sort(al);
        Iterator itr=al.iterator();
        while(itr.hasNext()){
            System.out.println(itr.next());
        }
    }
}
```



**RELIGARE** Health Insurance  
Religare Health Insurance

**Why Spend your saving in hospital?**

**Insure your family with us**

- ✓ Auto Recharge of sum insured.
- ✓ Get Health Check-ups, every year.
- ✓ Cashless treatment @ 4100+ hospitals

**BUY EASILY IN 2 mins**

Insurance is a subject matter of solicitation. Religare Health Insurance Co. Ltd. IRDA reg. no -148. For more details on risk factors, terms and conditions please read sales brochure carefully before concluding a sale. UAN 12121080

**Test it Now**

Output: Mukesh  
Saurav  
Tahir  
Virus

## Example of Sorting the elements of List that contains Wrapper class objects

```
import java.util.*;
class TestSort2{
public static void main(String args[]){

ArrayList al=new ArrayList();
al.add(Integer.valueOf(201));
al.add(Integer.valueOf(101));
al.add(230);//internally will be converted into objects as Integer.valueOf(230)

Collections.sort(al);

Iterator itr=al.iterator();
while(itr.hasNext()){
System.out.println(itr.next());
}
}
}
```

**Test it Now**

Output: 101  
201  
230

&lt;&lt;prev

next&gt;&gt;

Share  0