

```

1  -----TestVector.java-----
2  package com.ameya.test;
3
4  import java.util.Vector;
5
6  public class TestVector {
7
8      public static void main(String[] args) {
9          Vector<Integer> v=new Vector<Integer>(5,2);
10         // size-->current no. elements in vector
11         //capacity--> capacity of the vector
12         System.out.println("SIZE :: "+v.size());
13         System.out.println("CAPACITY :: "+v.capacity());
14         v.addElement(10);
15         v.addElement(20);
16         v.addElement(30);
17         v.addElement(40);
18         v.addElement(50);
19         System.out.println("SIZE :: "+v.size());
20         System.out.println("CAPACITY :: "+v.capacity());
21         v.addElement(10);
22         v.addElement(20);
23         v.addElement(30);
24         // v.addElement(40);
25         // v.addElement(50);
26         System.out.println("SIZE :: "+v.size());
27         System.out.println("CAPACITY :: "+v.capacity());
28         // v.addElement(50);
29         // System.out.println("SIZE :: "+v.size());
30         // System.out.println("CAPACITY :: "+v.capacity());
31     }
32 }
33 -----TestHashSet.java-----
34 package com.ameya.test;
35
36 import java.util.HashSet;
37 import java.util.Iterator;
38
39 import com.ameya.domain.Person;
40
41 public class TestHashSet {

```

```

42
43     public static void main(String[] args) {
44         HashSet<Person> hs=new HashSet<Person>();
45         hs.add(new Person(5, "aaaa", "aaaa", 25));
46         hs.add(new Person(3, "bbbb", "bbbb", 26));
47         hs.add(new Person(4, "cccc", "cccc", 24));
48         hs.add(new Person(1, "dddd", "dddd", 28));
49         hs.add(new Person(2, "eeee", "eeee", 27));
50         hs.add(new Person(1, "ffff", "ffff", 28)); //Silently ignored
51         Iterator<Person> itr=hs.iterator();
52         while(itr.hasNext()) {
53             Person person=itr.next();
54             System.out.print(person);
55         }
56     }
57
58 }
59 -----TestTreeSet.java-----
60 package com.ameya.test;
61
62 import java.util.Iterator;
63 import java.util.TreeSet;
64
65 public class TestTreeSet {
66
67     public static void main(String[] args) {
68         TreeSet<Integer> ts=new TreeSet<Integer>();
69         ts.add(5);
70         ts.add(3);
71         ts.add(4);
72         ts.add(1);
73         ts.add(2);
74         System.out.println(ts);
75         System.out.println("=====");
76         Iterator<Integer> itr=ts.iterator();
77         while(itr.hasNext()) {
78             int n=itr.next();
79             System.out.println(n);
80         }
81         System.out.println("=====");
82         TreeSet<Integer> revTs=(TreeSet<Integer>) ts.descendingSet();

```

```

83      System.out.println(revTs);
84      System.out.println("=====");
85      Iterator<Integer> itr1=revTs.iterator();
86      while(itr1.hasNext()) {
87          int n=itr1.next();
88          System.out.println(n);
89      }
90  }
91
92  }
93  -----Person.java-----
94  package com.ameya.domain;
95
96  public class Person implements Comparable<Person>{
97      private long id;
98      private String firstName;
99      private String lastName;
100     private int age;
101     public Person() {
102         super();
103         // TODO Auto-generated constructor stub
104     }
105     public Person(long id, String firstName, String lastName, int age) {
106         super();
107         this.id = id;
108         this.firstName = firstName;
109         this.lastName = lastName;
110         this.age = age;
111     }
112     public long getId() {
113         return id;
114     }
115     public void setId(long id) {
116         this.id = id;
117     }
118     public String getFirstName() {
119         return firstName;
120     }
121     public void setFirstName(String firstName) {
122         this.firstName = firstName;
123     }

```

```

124     public String getLastName() {
125         return lastName;
126     }
127     public void setLastName(String lastName) {
128         this.lastName = lastName;
129     }
130     public int getAge() {
131         return age;
132     }
133     public void setAge(int age) {
134         this.age = age;
135     }
136     @Override
137     public String toString() {
138         return "Person [id=" + id + ", firstName=" + firstName + ",
139             lastName=" +
140             lastName + ", age=" + age + "]\n";
141     }
142     @Override
143     public boolean equals(Object obj) {
144         return this.id==((Person)obj).getId() ? true : false;
145     }
146     @Override
147     public int hashCode() {
148         final long prime=31;
149         long result=1;
150         result=prime*result+id;
151         return (int)result;
152     }
153     @Override
154     public int compareTo(Person o) {
155         return ((int)(this.id-o.getId()));
156     }
157 -----TestPersonTreeSet.java-----
158 package com.ameya.test;
159
160 import java.util.Iterator;
161 import java.util.TreeSet;
162
163 import com.ameya.domain.Person;

```

```

164
165 public class TestPersonTreeSet {
166
167     public static void main(String[] args) {
168         TreeSet<Person> ts=new TreeSet<Person>();
169         ts.add(new Person(5, "aaaa", "aaaa", 25));
170         ts.add(new Person(3, "bbbb", "bbbb", 26));
171         ts.add(new Person(4, "cccc", "cccc", 24));
172         ts.add(new Person(1, "dddd", "dddd", 28));
173         ts.add(new Person(2, "eeee", "eeee", 27));
174         System.out.println(ts);
175         System.out.println("=====");
176         Iterator<Person> itr=ts.iterator();
177         while(itr.hasNext()) {
178             Person person=itr.next();
179             System.out.print(person);
180         }
181         System.out.println("\n=====");
182         TreeSet<Person> rev=(TreeSet<Person>) ts.descendingSet();
183         Iterator<Person> itr1=rev.iterator();
184             System.out.println(rev);
185         System.out.println("=====");
186         while(itr1.hasNext()) {
187             Person person=itr1.next();
188             System.out.print(person);
189         }
190     }
191
192 }
193
194

```