

```
1 -----Person.java-----
2 package com.ameya.domain;
3
4 public class Person implements Comparable<Person>{
5     private long id;
6     private String firstName;
7     private String lastName;
8     private int age;
9     public Person() {
10         super();
11         // TODO Auto-generated constructor stub
12     }
13     public Person(long id, String firstName, String lastName, int age) {
14         super();
15         this.id = id;
16         this.firstName = firstName;
17         this.lastName = lastName;
18         this.age = age;
19     }
20     public long getId() {
21         return id;
22     }
23     public void setId(long id) {
24         this.id = id;
25     }
26     public String getFirstName() {
27         return firstName;
28     }
29     public void setFirstName(String firstName) {
30         this.firstName = firstName;
31     }
32     public String getLastName() {
33         return lastName;
34     }
35     public void setLastName(String lastName) {
36         this.lastName = lastName;
37     }
38     public int getAge() {
39         return age;
40     }
41     public void setAge(int age) {
```

```

42     this.age = age;
43 }
44 @Override
45 public String toString() {
46     return "Person [id=" + id + ", firstName=" + firstName + ",
47         lastName=" +
48     lastName + ", age=" + age + "]\n";
49 }
50 @Override
51 public boolean equals(Object obj) {
52     return this.id==((Person)obj).getId() ? true : false;
53 }
54 @Override
55 public int hashCode() {
56     final long prime=31;
57     long result=1;
58     result=prime*result+id;
59     return (int)result;
60 }
61 @Override
62 public int compareTo(Person o) {
63     return ((int)(this.id-o.getId()));
64 }

```

65 -----FirstNameComparator.java-----

```

66 package com.ameya.utils;
67
68 import java.util.Comparator;
69
70 import com.ameya.domain.Person;
71
72 public class FirstNameComparator implements Comparator<Person>{
73
74     @Override
75     public int compare(Person o1, Person o2) {
76         return o1.getFirstName().compareTo(o2.getFirstName());
77     }
78
79 }

```

80 -----AgeComparator.java-----

```

81 package com.ameya.utils;

```

```

82
83 import java.util.Comparator;
84
85 import com.ameya.domain.Person;
86
87 public class AgeComparator implements Comparator<Person>{
88
89     @Override
90     public int compare(Person o1, Person o2) {
91         return o1.getAge()-o2.getAge();
92     }
93
94 }
95 -----TestComparators.java-----
96 package com.ameya.test;
97
98 import java.util.ArrayList;
99 import java.util.Collections;
100
101 import com.ameya.domain.Person;
102 import com.ameya.utils.AgeComparator;
103 import com.ameya.utils.FirstNameComparator;
104
105 public class TestComparators {
106
107     public static void main(String[] args) {
108         ArrayList<Person> list=new ArrayList<Person>();
109         list.add(new Person(5,"cccc","cccc",22));
110         list.add(new Person(3,"eeee","eeee",27));
111         list.add(new Person(4,"aaaa","aaaa",23));
112         list.add(new Person(1,"bbbb","bbbb",25));
113         list.add(new Person(2,"dddd","dddd",24));
114
115         System.out.println("-----
116         -----");
117         System.out.println("List - No Sorting Criteria");
118
119         System.out.println("-----
120         -----");
121         System.out.println(list);
122

```

```

119      System.out.println("-----
120      -----");
121      System.out.println("List - Default Sorting Criteria - on ID");
122
123      System.out.println("-----
124      -----");
125      Collections.sort(list);
126      System.out.println(list);
127
128      System.out.println("-----
129      -----");
130      System.out.println("List - Sorting Criteria - on FIRSTNAME");
131
132      System.out.println("-----
133      -----");
134      Collections.sort(list,new FirstNameComparator());
135      System.out.println(list);
136
137      System.out.println("-----
138      -----");
139      System.out.println("List - Sorting Criteria - on AGE");
140
141      System.out.println("-----
142      -----");
143      Collections.sort(list,new AgeComparator());
144      System.out.println(list);
145
146      System.out.println("-----
147      -----");
148      System.out.println("List - Sorting Criteria - on ID - Descending");
149
150      System.out.println("-----
151      -----");
152      Collections.sort(list,Collections.reverseOrder());
153      System.out.println(list);
154      System.out.println("List - Sorting Criteria - on FIRSTNAME -
155      Descending");
156
157      System.out.println("-----
158      -----");
159      Collections.sort(list,Collections.reverseOrder(new

```

```

        FirstNameComparator()));
141     System.out.println(list);
142     System.out.println("List - Sorting Criteria - on AGE -
        Descending");
143
        System.out.println("-----
        -----");
144     Collections.sort(list,Collections.reverseOrder(new
        AgeComparator()));
145     System.out.println(list);
146 }
147
148 }
149 -----USING ANNONYMOUS INNER
        CLASSES-----
150 -----PersonSortingUtil.java-----
151 package com.ameya.utils;
152
153 import java.util.Collections;
154 import java.util.Comparator;
155 import java.util.List;
156
157 import com.ameya.domain.Person;
158
159 public class PersonSortingUtil {
160     private List<Person> persons;
161     public PersonSortingUtil(List<Person> persons) {
162         this.persons=persons;
163     }
164     public void sortOnIdDesc() {
165         Collections.sort(persons,Collections.reverseOrder());
166     }
167     public void sortOnIdAsc() {
168         Collections.sort(persons);
169     }
170     public void sortOnFirstNameAsc() {
171         Collections.sort(persons,
172             new Comparator<Person>() {
173                 @Override
174                 public int compare(Person p1,Person p2) {
175                     return

```

```

        p1.getFirstName().compareTo(p2.getFirstName()));
176     }
177     });
178 }
179 public void sortOnAgeAsc() {
180     Collections.sort(persons,
181         new Comparator<Person>() {
182             @Override
183             public int compare(Person p1, Person p2) {
184                 return p1.getAge()-p2.getAge();
185             }
186         });
187 }
188 public void sortOnFirstNameDesc() {
189     Collections.sort(persons, Collections.reverseOrder(
190         new Comparator<Person>() {
191             @Override
192             public int compare(Person p1, Person p2) {
193                 return
194                     p1.getFirstName().compareTo(p2.getFirstName()));
195             }
196         }));
197 }
198 public void sortOnAgeDesc() {
199     Collections.sort(persons, Collections.reverseOrder(
200         new Comparator<Person>() {
201             @Override
202             public int compare(Person p1, Person p2) {
203                 return p1.getAge()-p2.getAge();
204             }
205         }));
206 }
207 public void printPersonsList() {
208     System.out.println("=====");
209     System.out.println(persons);
210     System.out.println("=====");
211 }
212 }
213 -----TestComparatorsAnonymous.java-----
214 package com.ameya.test;

```

```
215 import java.util.ArrayList;
216
217 import com.ameya.domain.Person;
218 import com.ameya.utils.PersonSortingUtil;
219
220 public class TestComparatorsAnonymous {
221
222     public static void main(String[] args) {
223         ArrayList<Person> list=new ArrayList<Person>();
224         list.add(new Person(5,"cccc","cccc",22));
225         list.add(new Person(3,"eeee","eeee",27));
226         list.add(new Person(4,"aaaa","aaaa",23));
227         list.add(new Person(1,"bbbb","bbbb",25));
228         list.add(new Person(2,"dddd","dddd",24));
229         PersonSortingUtil util=new PersonSortingUtil(list);
230         util.sortOnIdAsc();
231         util.printPersonsList();
232         util.sortOnIdDesc();
233         util.printPersonsList();
234         util.sortOnFirstNameAsc();
235         util.printPersonsList();
236         util.sortOnFirstNameDesc();
237         util.printPersonsList();
238         util.sortOnAgeAsc();
239         util.printPersonsList();
240         util.sortOnAgeDesc();
241         util.printPersonsList();
242
243     }
244
245 }
246
```