

```
1 19-may-2022          STREAM API-JAVA8
2 -----
3
4 package Java8Features;
5 import java.util.ArrayList;
6 class Product
7 {
8     int id;
9     String name;
10    double price;
11    public Product(int id, String name, double price) {
12        super();
13        this.id = id;
14        this.name = name;
15        this.price = price;
16    }
17
18 }
19 public class StreamTestProduct {
20
21     public static void main(String[] args) {
22         ArrayList<Product> al=new ArrayList<Product>();
23         al.add(new Product(1,"Dell Laptop",45000.2));
24         al.add(new Product(2,"HP Laptop",55000.2));
25         al.add(new Product(3,"LENOVA Laptop",25000.2));
26         al.add(new Product(4,"APPLE Laptop",90000.2));
27         al.stream()
28             .filter(p->p.id>=2)
29             .forEach(p->System.out.println(p.name));
30
31
32     }
33
34 }
```

```
35 OUTPUT:
36 -----
```

```
37 HP Laptop
38 LENOVA Laptop
39 APPLE Laptop
40 -----
```

```
41
42 2.THE TASK IS TO COLLECT THE NEGATIVE EVEN NUMBERS FROM THE
43 GIVEN ARRAYLIST AND STORE THE VALUES INTO NEW ARRAY;LIST
44 BY USING FILTER AND COLLECTOR METHOD.
```

```
45
46 package Java8Features;
47 import java.util.Arrays;
48 import java.util.List;
49 import java.util.stream.Collectors;
50
51 public class StreamTest1 {
52
53     public static void main(String[] args) {
54         List<Integer> al=Arrays.asList(1,-2,3,4,-6,-5,-4);
55         List<Integer>nl=al.stream()
56             .filter(p->((p<0)&&(p%2==0)))
57             .collect(Collectors.toList());
58         System.out.println(nl);
59     }
60
61 }
```

```
62 OUTPUT:
```

```
63 -----
64 [-2, -6, -4]
65 -----
```

```
66 3.GET AN EMPLOYEE DEATAILS AND SET THE LOCATION AS PUNE
67 AND PRINT THE RESPECTIVE EMPLOYEE DETAILS
```

```
68
69
70 package Java8Features;
71
72 import java.util.ArrayList;
```

```
73 import java.util.List;
74 import java.util.stream.Collectors;
75 import java.util.stream.Collectors;
76
77 class EmployeeTest
78 {
79     int empNo;
80     String name;
81     int age;
82     String location;
83     public EmployeeTest(int empNo, String name, int age, String location) {
84         super();
85         this.empNo = empNo;
86         this.name = name;
87         this.age = age;
88         this.location = location;
89     }
90     @Override
91     public String toString() {
92         return "EmployeeTest [empNo=" + empNo + ", name=" + name + ", age=" + age +
93             ", location=" + location
94             + "]";
95     }
96 }
97 public class StreamEmployeeTest {
98     public static void main(String[] args) {
99         ArrayList<EmployeeTest> al=new ArrayList<EmployeeTest>();
100         al.add(new EmployeeTest(1,"Abimanu",21,"mumbai"));
101         al.add(new EmployeeTest(2,"Beema",23,"mangalore"));
102         al.add(new EmployeeTest(3,"mukesh",22,"chennai"));
103         al.add(new EmployeeTest(4,"sakthi",24,"pune"));
104         al.add(new EmployeeTest(5,"buvi",25,"pune"));
105         ArrayList ls=(ArrayList)al.stream()
106             .filter(e->e.location=="pune")
107             .collect(Collectors.toList());
```

```
108         ls.forEach(System.out::println);
109
110     }
111
112 }
113 -----
114 OUTPUT:
115 -----
116 EmployeeTest [empNo=4, name=sakthi, age=24, location=pune]
117 EmployeeTest [empNo=5, name=buvi, age=25, location=pune]
118 -----
```

```
119
120 3.FILTER THE PASS MARK STUDENT WHOSE MARKS IS 50 AND ABOVE
121 SOLVE THIS PROBLEM BY USING COUNT AND FILTER METHOD.
```

```
122
123 package Java8Features;
124
125 import java.util.ArrayList;
126 import java.util.stream.Collectors;
127
128 class StudentTest
129 {
130     int roll;
131     String name;
132     int mark;
133     public StudentTest(int roll, String name, int mark) {
134         super();
135         this.roll = roll;
136         this.name = name;
137         this.mark = mark;
138     }
139
140 }
141 public class StreamStudentTest {
142
143     public static void main(String[] args)
```

```

144     {
145         ArrayList<StudentTest> al=new ArrayList<StudentTest>();
146         al.add(new StudentTest(1, "mukesh", 95));
147         al.add(new StudentTest(2, "logesh", 98));
148         al.add(new StudentTest(3, "lite mukesh", 100));
149         al.add(new StudentTest(4, "tej", 45));
150         al.add(new StudentTest(5, "mehck", 46));
151         Long ls= al.stream()
152             .filter(s->s.mark>50)
153             .collect(Collectors.counting());
154         System.out.println(ls);
155     }
156
157 }

```

```

158 -----
159 OUTPUT :
160 -----
161 3
162 -----

```

```

163
164 20-may-2022      StreamCollect()
165 *****          *****
166
167 package Java8Features;
168
169 import java.util.ArrayList;
170 import java.util.List;
171 import java.util.Map;
172 import java.util.Set;
173 import java.util.stream.Collectors;
174
175 class School {
176     String section;
177     String Depart;
178     int roll;
179     String name;

```

```

180
181 public School(String section, String depart, int roll, String name) {
182     super();
183     this.section = section;
184     Depart = depart;
185     this.roll = roll;
186     this.name = name;
187 }
188
189 @Override
190 public String toString() {
191     return "[section=" + section + ", Depart=" + Depart + ", roll=" + roll + ",
192         name=" + name + " ]";
193 }
194
195 }
196
197 public class CollectorsMethodTest {
198
199     public static void main(String[] args) {
200         ArrayList<School> al = new ArrayList<School>();
201         al.add(new School("A", "cse", 1, "mukesh"));
202         al.add(new School("A", "cse", 2, "basith"));
203         al.add(new School("B", "biology", 3, "ajmal"));
204         al.add(new School("C", "commerce", 4, "ajmal"));
205         System.out.println("          Displaying in List format          ");
206         System.out.println("-----");
207         List<String> lname = al.stream().filter(s -> s.roll < 4).map(s ->
            s.name.toUpperCase()).collect(Collectors.toList());
208         System.out.println(lname);
209         System.out.println("-----");
210         System.out.println("          Displaying in Set format          ");
211         System.out.println("-----");
212         Set<String> s = al.stream().map(st ->
            st.name.toUpperCase()).collect(Collectors.toSet());

```

```

213     System.out.println(s);
214     System.out.println("-----");
215     System.out.println("          Joing the String          ");
216     System.out.println("-----");
217     String allNames = al.stream().map(j ->
218     j.name.toUpperCase()).collect(Collectors.joining(" * "));
219     System.out.println(allNames);
220     System.out.println("-----");
221     System.out.println("          Grouping          ");
222     System.out.println("-----");
223     Map<String, List<School>> mapList =
224     al.stream().collect(Collectors.groupingBy(g -> g.section));
225     System.out.println(mapList);
226     mapList.forEach((k, v) -> System.out.println("Key " + k + "---" + " Value "
227     + v));
228     System.out.println("-----");
229     System.out.println("          Average Finding          ");
230     System.out.println("-----");
231     Double averageOfRoll=al.stream().collect(Collectors.averagingInt(a->a.roll));
232     System.out.println("The average is : " +averageOfRoll);
233     System.out.println("-----");
234     System.out.println("          Partision          ");
235     System.out.println("-----");
236     Map<Boolean,
237     List<School>>mapL=al.stream().collect(Collectors.partitioningBy(p->p.roll>0));
238     System.out.println(mapL);
239     System.out.println("-----");
240 }
241 }
242 OutPut:
243 -----
244          Displaying in List format
245 -----
246 [MUKESH, BASITH, AJMAL]

```

```
245 -----
246         Displaying in Set format
247 -----
248 [MUKESH, BASITH, AJMAL]
249 -----
250         Joing the String
251 -----
252 MUKESH * BASITH * AJMAL * AJMAL
253 -----
254         Grouping
255 -----
256 {A=[[section=A, Depart=cse, roll=1, name=mukesh ],
257 [section=A, Depart=cse, roll=2, name=basith ]],
258  B=[[section=B, Depart=biology, roll=3, name=ajmal ]],
259  C=[[section=C, Depart=commerce, roll=4, name=ajmal ]]}
260 Key A--- Value [[section=A, Depart=cse, roll=1, name=mukesh ],
261 [section=A, Depart=cse, roll=2, name=basith ]]
262 Key B--- Value [[section=B, Depart=biology, roll=3, name=ajmal ]]
263 Key C--- Value [[section=C, Depart=commerce, roll=4, name=ajmal ]]
264 -----
265         Average Finding
266 -----
267 The average is : 2.5
268 -----
269         Partision
270 -----
271 {false=[], true=[[section=A, Depart=cse, roll=1, name=mukesh ], [section=A,
272 Depart=cse, roll=2, name=basith ], [section=B, Depart=biology, roll=3, name=ajmal ],
273 [section=C, Depart=commerce, roll=4, name=ajmal ]]}
```