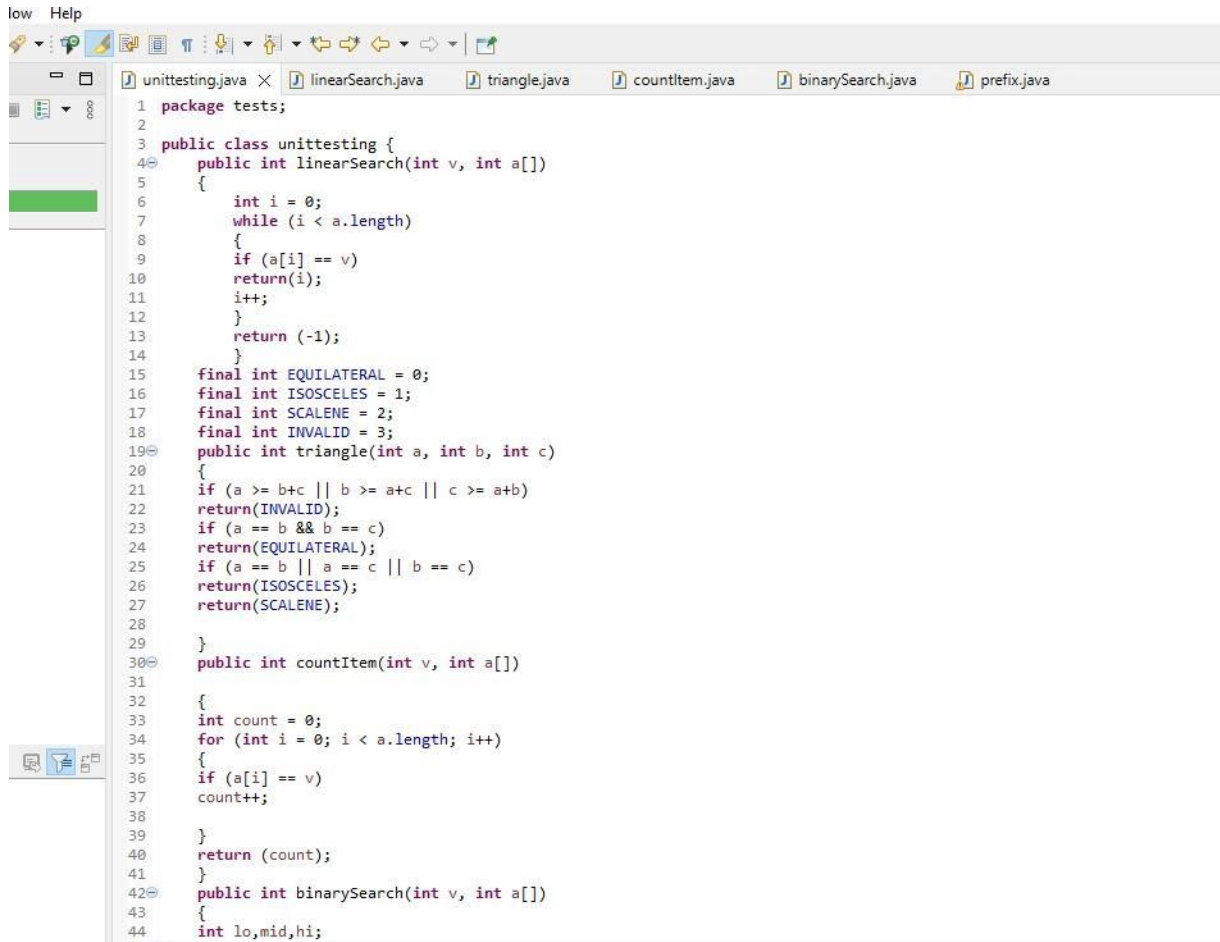


Name: Nandini
Chaudhary
ID: 202001090

Code:



```
low  Help
unittesting.java  linearSearch.java  triangle.java  countItem.java  binarySearch.java  prefix.java
1  package tests;
2
3  public class unittesting {
4  public int linearSearch(int v, int a[])
5  {
6      int i = 0;
7      while (i < a.length)
8      {
9          if (a[i] == v)
10         return(i);
11         i++;
12     }
13     return (-1);
14 }
15 final int EQUILATERAL = 0;
16 final int ISOSCELES = 1;
17 final int SCALENE = 2;
18 final int INVALID = 3;
19 public int triangle(int a, int b, int c)
20 {
21     if (a >= b+c || b >= a+c || c >= a+b)
22         return(INVALID);
23     if (a == b && b == c)
24         return(EQUILATERAL);
25     if (a == b || a == c || b == c)
26         return(ISOSCELES);
27     return(SCALENE);
28 }
29 }
30 public int countItem(int v, int a[])
31 {
32     int count = 0;
33     for (int i = 0; i < a.length; i++)
34     {
35         if (a[i] == v)
36             count++;
37     }
38     return (count);
39 }
40 }
41 public int binarySearch(int v, int a[])
42 {
43     int lo,mid,hi;
```

```
unittesting.java x linearSearch.java triangle.java countItem.java binarySearch.java prefix.java
33     int count = 0;
34     for (int i = 0; i < a.length; i++)
35     {
36         if (a[i] == v)
37             count++;
38     }
39     return (count);
40 }
41 }
42 public int binarySearch(int v, int a[])
43 {
44     int lo, mid, hi;
45     lo = 0;
46     hi = a.length-1;
47     while (lo <= hi)
48     {
49         mid = (lo+hi)/2;
50         if (v == a[mid])
51             return (mid);
52         else if (v < a[mid])
53             hi = mid-1;
54         else
55             lo = mid+1;
56     }
57     return(-1);
58 }
59 }
60 public static boolean prefix(char s1[], char s2[])
61 {
62     if (s1.length > s2.length)
63     {
64         return false;
65     }
66     for (int i = 0; i < s1.length; i++)
67     {
68         if (s1[i] != s2[i])
69         {
70             return false;
71         }
72     }
73     return true;
74 }
75 }
76
```

eclipse-workspace - JavaTesting/src/tests/linearSearch.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer JUnit ×

Finished after 0.012 seconds

Runs: 1/1 Errors: 0 Failures: 0

> tests.triangle [Runner: JUnit 4] (0.000 s)

```
1 package tests;
2
3 import static org.junit.Assert.*;
4
5
6 public class linearSearch {
7
8
9     @Test
10    public void test1() {
11        unittesting obj1= new unittesting();
12        int array[] = {1,2,3,4,5};
13        int output_f= obj1.linearSearch(4, array);
14        assertEquals(3,output_f);
15    }
16    public void test2() {
17        unittesting obj1= new unittesting();
18        int array[] = {1,2,3,'a',5};
19        int output_f= obj1.linearSearch(4, array);
20        assertEquals(-1,output_f);
21    }
22    public void test3() {
23        unittesting obj1= new unittesting();
24        int array[] = {1,2,3,5};
25        int output_f= obj1.linearSearch(4, array);
26        assertEquals(-1,output_f);
27    }
28    public void test4() {
29        unittesting obj1= new unittesting();
30        int array[] = {'\0'};
31        int output_f= obj1.linearSearch(4, array);
32        assertEquals(-1,output_f);
33    }
34 }
35
```

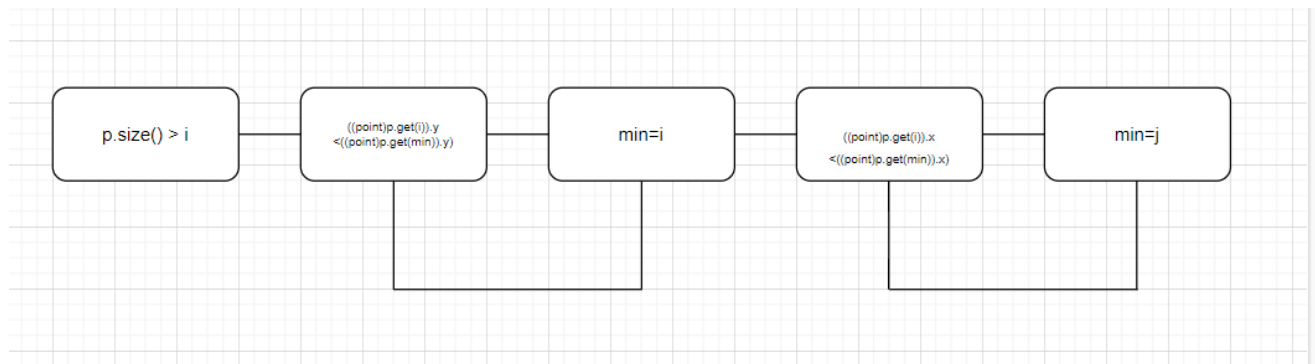
Values	Equivalent Output	Expected Output
4,{1,2,3,4}	3	3
4,{1,2,3,'a',5}	-1	-1
4, {1,2,3,5}	-1	-1
4, {'\0'}	-1	-1

Values	Equivalent Output	Expected Output
4, 4, 4	0	0
2, 1, 4	3	3
'a', 4, 4	3	3
3, 4, 5	2	2
3, 4, 4	1	1

Values	Equivalent Output	Expected Output
4, {1, 2, 3, 4, 5}	1	1
4, {1, 2, 3, 'a', 5}	0	0
4, {4, 4, 4, 4, 4}	5	5
4, {"\0"}	0	0

Values	Equivalent Output	Expected Output
4, {1, 2, 3, 4, 5}	3	1
-1, {1, 2, 3, 8, 5}	3	2
-1, {1, 2, 3, 'a', 5}	3	0
-1, {"0"}	3	3

Values	Equivalent Output	Expected Output
{h, e}, {h, e}	1	1
{e}, {h, e}	0	0
{h, t}, {h, e}	0	0
{h, e, d}, {h, e, 1}	0	0



2. a. Statement Coverage test set:

Test Case 1: `p.size() > point` i.e. 2 is false

Test Case 2: `p.size() > 2` is true

b. Branch Coverage Test Set:

Test Case 1: `p.size() > point` i.e. 2 is false

Test Case 2: `p.size() > 2` is true and loop is executed

Test Case 3: `p.size() > 2` is true and loop is not executed

c. Basic Condition Coverage Test Set:

Test Case 1: `p.size() > point` i.e. 2 is false

Test Case 2: `p.size() > 2` is true and loop is executed

Test Case 3: `p.size() > 2` is true and loop is not executed

Test Case 4: `p.size() > 2` is true and loop is executed twice