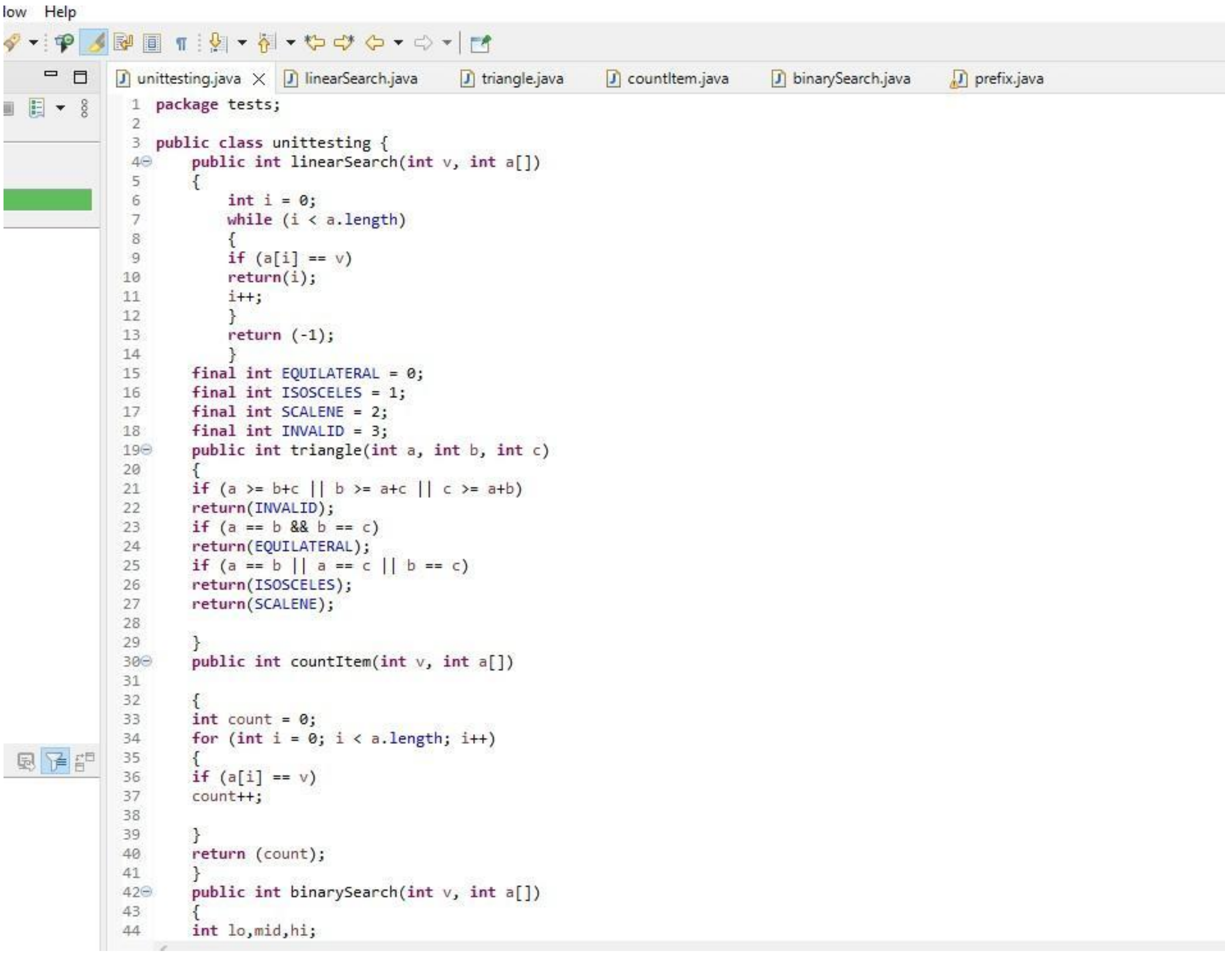
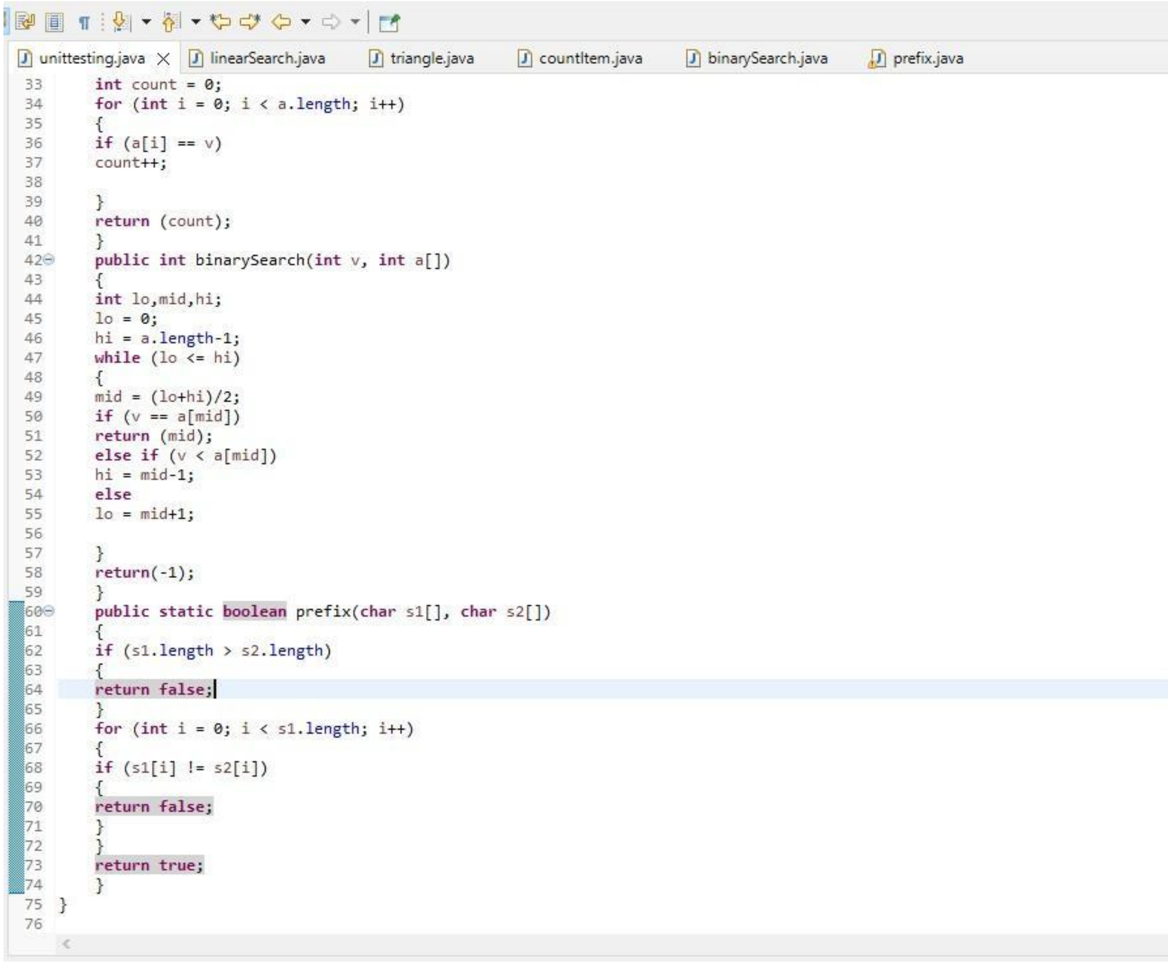
Name: Nandini Chaudhary
ID: 202001090

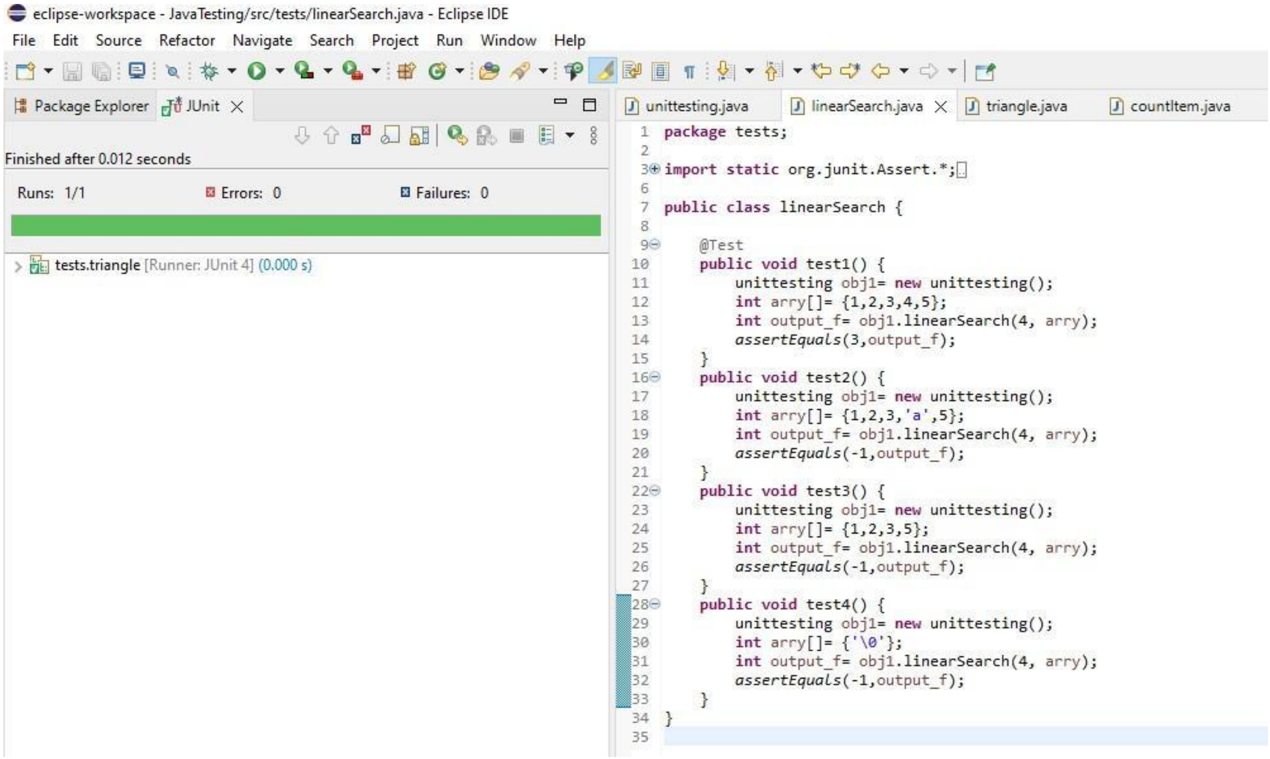
Code:



```java
package tests;

public class unittesting {
    public int linearSearch(int v, int a[])
    {
        int i = 0;
        while (i < a.length)
        {
        if (a[i] == v)
        return(i);
        i++;
        }
        return (-1);
        }
    final int EQUILATERAL = 0;
    final int ISOSCELES = 1;
    final int SCALENE = 2;
    final int INVALID = 3;
    public int triangle(int a, int b, int c)
    {
    if (a >= b+c || b >= a+c || c >= a+b)
    return(INVALID);
    if (a == b && b == c)
    return(EQUILATERAL);
    if (a == b || a == c || b == c)
    return(ISOSCELES);
    return(SCALENE);

    }
    public int countItem(int v, int a[])

    {
    int count = 0;
    for (int i = 0; i < a.length; i++)
    {
    if (a[i] == v)
    count++;

    }
    return (count);
    }
    public int binarySearch(int v, int a[])
    {
    int lo,mid,hi;
```



```
    int count = 0;
    for (int i = 0; i < a.length; i++)
    {
    if (a[i] == v)
    count++;

    }
    return (count);
    }
    public int binarySearch(int v, int a[])
    {
    int lo,mid,hi;
    lo = 0;
    hi = a.length-1;
    while (lo <= hi)
    {
    mid = (lo+hi)/2;
    if (v == a[mid])
    return (mid);
    else if (v < a[mid])
    hi = mid-1;
    else
    lo = mid+1;

    }
    return(-1);
    }
    public static boolean prefix(char s1[], char s2[])
    {
    if (s1.length > s2.length)
    {
    return false;
    }
    for (int i = 0; i < s1.length; i++)
    {
    if (s1[i] != s2[i])
    {
    return false;
    }
    }
    return true;
    }
}
```


eclipse-workspace - JavaTesting/src/tests/linearSearch.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
unittesting.java
linearSearch.java
triangle.java
countItem.java

```
Finished after 0.012 seconds

Runs: 1/1        Errors: 0        Failures: 0

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

```java
package tests;

import static org.junit.Assert.*;

public class linearSearch {

    @Test
    public void test1() {
        unittesting obj1= new unittesting();
        int arry[]= {1,2,3,4,5};
        int output_f= obj1.linearSearch(4, arry);
        assertEquals(3,output_f);
    }
    public void test2() {
        unittesting obj1= new unittesting();
        int arry[]= {1,2,3,'a',5};
        int output_f= obj1.linearSearch(4, arry);
        assertEquals(-1,output_f);
    }
    public void test3() {
        unittesting obj1= new unittesting();
        int arry[]= {1,2,3,5};
        int output_f= obj1.linearSearch(4, arry);
        assertEquals(-1,output_f);
    }
    public void test4() {
        unittesting obj1= new unittesting();
        int arry[]= {'\0'};
        int output_f= obj1.linearSearch(4, arry);
        assertEquals(-1,output_f);
    }
}
```

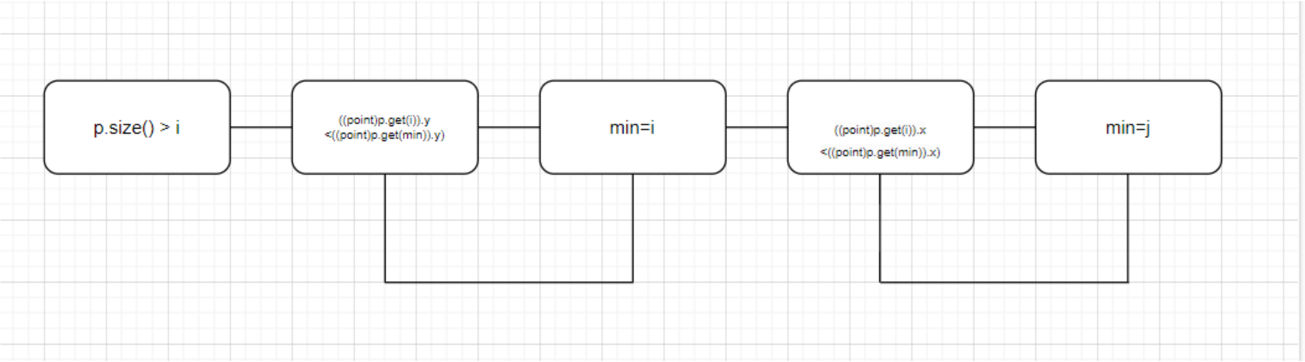
Values	Equivalent Output	Expected Output
4,{1,2,3,4}	3	3
4,{1,2,3,'a',5}	-1	-1
4, {1,2,3,5}	-1	-1
4, {'\0'}	-1	-1

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