

IT314 - Lab7

Name : Sumit Vaniya

ID : 202001085

Date : 12-04-2023

Section A :

Based on the input ranges, we can identify the following equivalence classes:

Valid dates: The input triple (day, month, year) that represent a valid date in the Gregorian calendar, such as (3, 4, 1995).

Invalid dates: The input triple (day, month, year) that represent an invalid date, such as (31, 2, 2022) or (29, 2, 1900).

Out of range dates: The input triple (day, month, year) that are outside the allowed ranges, such as (0, 5, 2010) or (15, 13, 2005). Based on these equivalence classes, we can design the following test cases:

Tester Action and Input Data Expected Outcome

Valid dates:

Calculate previous date for (15, 10, 2022) 14, 10, 2022

Calculate previous date for (1, 1, 2015) 31, 12, 2014

Calculate previous date for (31, 3, 2000) 30, 3, 2000

Invalid dates:

Calculate previous date for (29, 2, 2022) Invalid date

Calculate previous date for (31, 4, 2010) Invalid date

Calculate previous date for (30, 2, 2000) Invalid date

Out of range dates:

Calculate previous date for (0, 5, 2010) Invalid date

Calculate previous date for (15, 13, 2005) Invalid date

Calculate previous date for (31, 12, 1899) Invalid date

Boundary Value Analysis:

Using boundary value analysis, we can identify the following boundary test cases:

The earliest possible date: (1, 1, 1900)

The latest possible date: (31, 12, 2015)

The earliest day of each month: (1, 1, 2000), (1, 2, 2000), (1, 3, 2000),...,
(1, 12, 2000)

The latest day of each month: (31, 1, 2000), (28, 2, 2000), (31, 3, 2000),...,
(31, 12, 2000)

Leap year day: (29, 2, 2000)

Invalid leap year day: (29, 2, 1900)

One day before earliest date: (31, 12, 1899)

One day after latest date: (1, 1, 2016)

Based on these boundary test cases, we can design the following test cases:

Tester Action and Input Data Expected Outcome

Boundary Test Cases:

Calculate previous date for (1, 1, 1900) Invalid date

Calculate previous date for (31, 12, 2015) 30, 12, 2015

Calculate previous date for (1, 1, 2000) 31, 12, 1999

Calculate previous date for (31, 1, 2000) 30, 1, 2000

Calculate previous date for (29, 2, 2000) 28, 2, 2000

Calculate previous date for (29, 2, 1900) Invalid date
Calculate previous

Programs

(P1). The function linearSearch searches for a value v in an array of integers a. If v appears in the array a, then the function returns the first index i, such that a[i] == v; otherwise, -1 is returned.

```
int linearSearch(int v, int a[]){
    int i = 0;
    while (i < a.length){
        if (a[i] == v)
            return(i);
        i++;
    }
    return (-1);
}
```

Test Cases :

(1) V = 2 , a={4,3,2} expected output = 2

eclipse-workspace - Lab7_202001085/src/p1/linearsrch.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit lab7_p1.java linearsrch.java lab7_p2.java countnum.java
Outline View Problems JavaDoc Declaration
Failure Trace
1 package p1;
2
3 import static org.junit.Assert.*;
4
5 public class linearsrch {
6
7     @Test
8     public void test() {
9         lab7_p1 pl = new lab7_p1();
10        int v=2;
11        int a[]={new int[]{4,3,2}};
12        assertEquals(1,pl.linearSearch(v,a));
13    }
14
15 }
16
17 // @Test
18 // public void test2() {
19 //     lab7_p1 pl = new lab7_p1();
20 //     int v=2;
21 //     int a[]={new int[]{4,2,1}};
22 //     assertEquals(-1,pl.linearSearch(v,a));
23 // }
24
25
26
27 // @Test
28 // public void test3() {
29 //     lab7_p1 pl = new lab7_p1();
30 //     int v=20;
31 //     int a[]={new int[]{10,20,30,20,40}};
32 //     assertEquals(1,pl.linearSearch(v,a));
33 // }
34
35
36
37 // @Test
38 // public void test4() {
39 //     lab7_p1 pl = new lab7_p1();
40 //     int v=20;
41 //     int a[]={new int[]{10,20,30,20,40}};
42 //     assertEquals(-1,pl.linearSearch(v,a));
43 // }
44
45 }

```

9 errors, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (9 items)				
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 11	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 11	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 19	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 19	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 27	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 27	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 35	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 35	Java Problem

Writable Smart Insert 17:5 [1]

37°C Sunny ENG 12-04-2023

(2) $V = 2, a=\{4,2,1\}$ expected output = -1

eclipse-workspace - Lab7_202001085/src/p1/linearsrch.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit lab7_p1.java linearsrch.java lab7_p2.java countnum.java
Outline View Problems JavaDoc Declaration
Failure Trace
1 package p1;
2
3 import static org.junit.Assert.*;
4
5 public class linearsrch {
6
7     @Test
8     public void test() {
9         lab7_p1 pl = new lab7_p1();
10        int v=2;
11        int a[]={new int[]{4,3,2}};
12        assertEquals(1,pl.linearSearch(v,a));
13    }
14
15 }
16
17 // @Test
18 // public void test2() {
19 //     lab7_p1 pl = new lab7_p1();
20 //     int v=2;
21 //     int a[]={new int[]{4,2,1}};
22 //     assertEquals(-1,pl.linearSearch(v,a));
23 // }
24
25
26
27 // @Test
28 // public void test3() {
29 //     lab7_p1 pl = new lab7_p1();
30 //     int v=4;
31 //     int a[]={new int[]{}};
32 //     assertEquals(-1,pl.linearSearch(v,a));
33 // }
34
35
36
37 // @Test
38 // public void test4() {
39 //     lab7_p1 pl = new lab7_p1();
40 //     int v=20;
41 //     int a[]={new int[]{10,20,30,20,40}};
42 //     assertEquals(-1,pl.linearSearch(v,a));
43 // }
44
45 }

```

9 errors, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (9 items)				
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 11	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 11	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 19	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 19	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 27	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 27	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 35	Java Problem
lab7_p1 cannot be resolved to a type	binarySearch...	/Lab7_202001085/src/p1/linearsrch.java	line 35	Java Problem

Writable Smart Insert 19:1 [152]

37°C Sunny ENG 12-04-2023

(3) $V = 2, a=\{\}$ expected output = -1

The screenshot shows the Eclipse IDE interface with the following details:

- Top Bar:** File, Edit, Source, Refactor, Navigator, Search, Project, Run, Window, Help.
- Left Sidebar:** Package Explorer (JUnit X), Problems (0 errors, 9 warnings), and Failure Trace.
- Middle Area:** A large code editor window displaying Java code for a `linearsearch` class. The code includes several test methods using the `Assert` library to verify the correctness of the search algorithm.
- Right Sidebar:** Outline (p1 package), Help (Welcome, Contents, Search, Related Topics), and Java Editor (about Java support, see also links).
- Bottom Bar:** Problems (0 errors, 9 warnings, 0 others), a search bar, and system status indicators (37°C, 16:30, ENG, 12-04-2023).

Java Code in the Editor:

```
package p1;
import static org.junit.Assert.*;
public class linearsearch {
    @Test
    public void test() {
        lab7_p1 pi = new lab7_p1();
        int v=2;
        int []vnew = {1,3,2};
        assertEquals(2,pi.linearSearch(v,a));
    }
    @Test
    public void test2() {
        lab7_p1 pi = new lab7_p1();
        int v4;
        int []vnew = {1,2,1};
        assertEquals(-1,pi.linearSearch(v,a));
    }
}
// @Test
public void test3() {
    lab7_p1 pi = new lab7_p1();
    int v4;
    int []vnew = {};
    assertEquals(-1,pi.linearSearch(v,a));
}
// @Test
public void test4() {
    lab7_p1 pi = new lab7_p1();
    // int v=28;
    // int []vnew = {18,29,30,29,48};
    // assertEquals(1,pi.linearSearch(v,a));
}
// }
```

Problems View:

Description	Resource	Path	Location	Type
Errors (9 items)				
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 11		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 14		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 19		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 27		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 27		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 35		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/src/	line 35		Java Problem

(4) $V = 2$, $a=\{10,20,30,20,40\}$ expected output = 1

The screenshot shows the Eclipse IDE interface with the following details:

- File Path:** eclipse-workspace - Lab7_202001085/src/p1/linsearch.java - Eclipse IDE
- Toolbar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Package Explorer:** Shows the project structure with p1 and linsearch packages.
- Outline View:** Displays the class structure and methods: linearsearch, test(), test2(), test3(), and test4().
- Java Help:** Provides links to Java editor support, Java help, and Java editor concepts.
- Failure Trace:** Shows a single entry: lab7_p1 cannot be resolved to a type.
- Problems View:** Lists 9 errors related to the lab7_p1 type being unresolved across various lines of code.
- Bottom Status Bar:** Shows the date (12-04-2023), time (16:36), and system status (37°C Sunny).

```
1 package p1;
2
3 import static org.junit.Assert.*;
4
5 public class linsearch {
6
7     @Test
8     public void test() {
9         lab7_p1 p1 = new lab7_p1();
10        int v=2;
11        int []v=new int[] {4,3,2};
12        assertEquals(2,p1.linearSearch(v,v));
13    }
14
15 }
16
17
18
19     @Test
20     public void test2() {
21         lab7_p1 p1 = new lab7_p1();
22        int v=3;
23        int []v=new int[] {4,2,1};
24        assertEquals(-1,p1.linearSearch(v,v));
25    }
26
27
28     @Test
29     public void test3() {
30         lab7_p1 p1 = new lab7_p1();
31        int v=4;
32        int []v=new int[] {};
33        assertEquals(-1,p1.linearSearch(v,v));
34    }
35
36
37     @Test
38     public void test4() {
39         lab7_p1 p1 = new lab7_p1();
40        int v=2;
41        int []v=new int[] {10,20,30,20,40};
42        assertEquals(1,p1.linearSearch(v,v));
43    }
44
45 }
```

Description	Resource	Path	Location	Type
Errors (9 items)				
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 11		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 11		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 19		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 27		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 27		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 35		Java Problem
lab7_p1 cannot be resolved to a type	binarySearch... /Lab7_202001085/src/	line 35		Java Problem

(P2). The function countItem returns the number of times a value v appears in an array of integers a.

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

Test Cases :

(1) v=2 , a={4,2,3,2,1} expected output = 2

eclipse-workspace - Lab7_202001085/src/p2/countitem.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit lab7_p1.java linearsearch.java lab7_p2.java countitem.java
Finished after 0.012 seconds
Runs: 1/1 Errors: 0 Failures: 0
p2.countitem (Runner: JUnit 4) [0.000 s]
1 package p2;
2
3 import static org.junit.Assert.*;
4
5 import org.junit.Test;
6
7
8 public class countitem {
9
10    @Test
11    public void test() {
12
13        lab7_p2 p2 = new lab7_p2();
14        int v=2;
15        int a[]={new int[]{4,2,3,2,1}};
16        assertEquals(2,p2.countItem(v,a));
17    }
18
19
20    @Test
21    public void test2() {
22
23        lab7_p2 p2 = new lab7_p2();
24        int v=3;
25        int a[]={new int[]{4,2,3,2,1}};
26        assertEquals(3,p2.countItem(v,a));
27    }
28
29
30

```

Outline View

Help > Welcome

About Java Editor

See also:

- Java editor concepts
- Java editor reference
- Opening a Java editor
- Using context assist
- Identifying problems in your code
- Using code templates
- Organizing import statements
- Java editor preferences
- Quick Fixes
- Quick Assists
- Configuring accessibility options for textual editors

More results:

Search for Java Editor

Failure Trace

Problems > Javadoc Declaration

9 errors, 0 warnings, 0 others

Description	Resource	Path	Location	Type
↳ Error (9 items)				
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 11		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 11		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 19		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 19		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 27		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 27		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 35		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 35		Java Problem

Writable Smart Insert 20:2:265

37°C Sunny 16:30 ENG 12-04-2023

(2) $v=3$, $a=\{4,2,3\}$ expected output =1

eclipse-workspace - Lab7_202001085/src/p2/countitem.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit lab7_p1.java linearsearch.java lab7_p2.java countitem.java
Finished after 0.011 seconds
Runs: 2/2 Errors: 0 Failures: 0
p2.countitem (Runner: JUnit 4) [0.000 s]
1 package p2;
2
3 import static org.junit.Assert.*;
4
5 import org.junit.Test;
6
7
8 public class countitem {
9
10    @Test
11    public void test() {
12
13        lab7_p2 p2 = new lab7_p2();
14        int v=2;
15        int a[]={new int[]{4,2,3,2,1}};
16        assertEquals(2,p2.countItem(v,a));
17    }
18
19
20    @Test
21    public void test2() {
22
23        lab7_p2 p2 = new lab7_p2();
24        int v=3;
25        int a[]={new int[]{4,2,3,2,1}};
26        assertEquals(3,p2.countItem(v,a));
27    }
28
29
30

```

Outline View

Help > Welcome

About Java Editor

See also:

- Java editor concepts
- Java editor reference
- Opening a Java editor
- Using context assist
- Identifying problems in your code
- Using code templates
- Organizing import statements
- Java editor preferences
- Quick Fixes
- Quick Assists
- Configuring accessibility options for textual editors

More results:

Search for Java Editor

Failure Trace

Problems > Javadoc Declaration

9 errors, 0 warnings, 0 others

Description	Resource	Path	Location	Type
↳ Error (9 items)				
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 11		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 11		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 19		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 19		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 27		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 27		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 35		Java Problem
↳ lab7_p1 cannot be resolved to a type	binarySearch.../Lab7_202001085/...	line 35		Java Problem

Writable Smart Insert 21:26:294

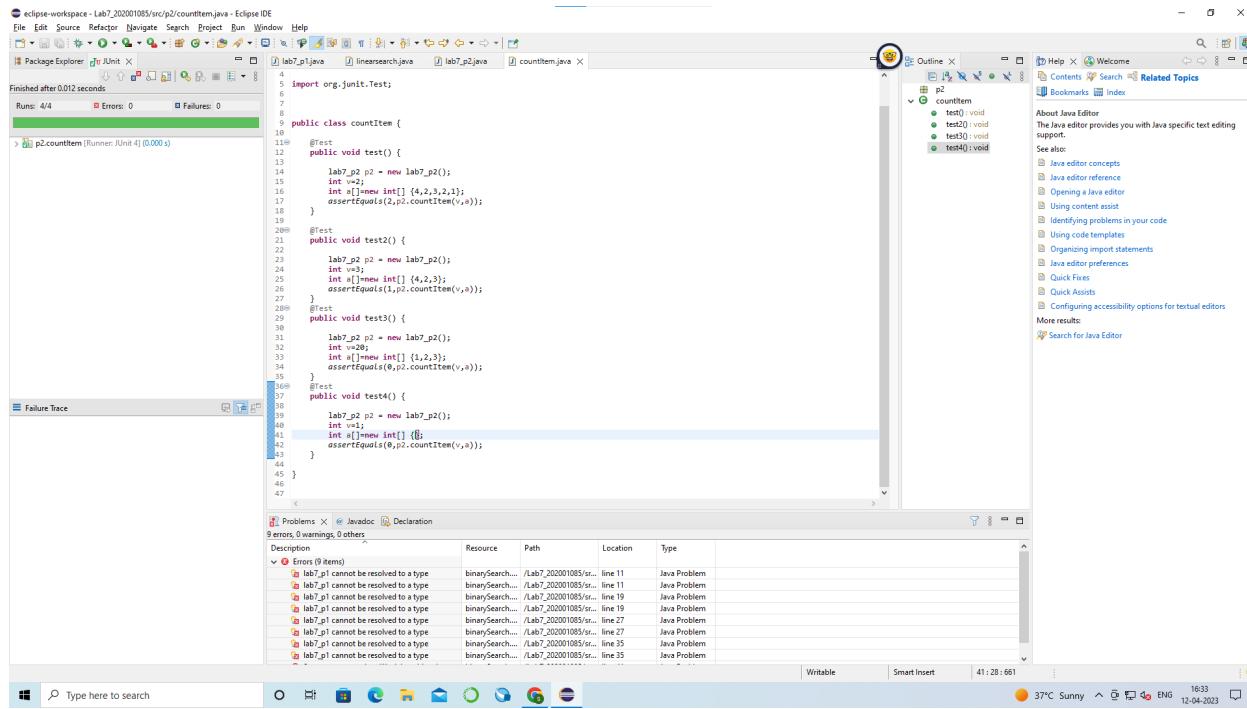
37°C Sunny 16:31 ENG 12-04-2023

(3) v= 20 , a= {1,2,3} expected output =0

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files lab7_p1.java, linearsearch.java, lab7_p2.java, and countitem.java.
- Java Editor:** Displays the Java code for the countitem class and its test methods. The code uses JUnit annotations (@Test) and assert statements to check if the countitem method returns 0 for different inputs.
- Outline View:** Shows the class structure with methods test0(), test2(), and test3().
- Problems View:** Lists 9 errors, all related to the unresolved reference lab7_p1. The errors are:
 - lab7_p1 cannot be resolved to a type
 - lab7_p1 cannot be resolved to a type
- Bottom Status Bar:** Shows system information: 37°C Sunny, ENG, 16:33, 12-04-2023.

(4) v=1 , a ={} , expected output = 0



(P3). The function `binarySearch` searches for a value `v` in an ordered array of integers `a`. If `v` appears in the array `a`, then the function returns an index `i`, such that `a[i] == v`; otherwise, `-1` is returned.

Assumption: the elements in the array `a` are sorted in non-decreasing order.

```
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])

```

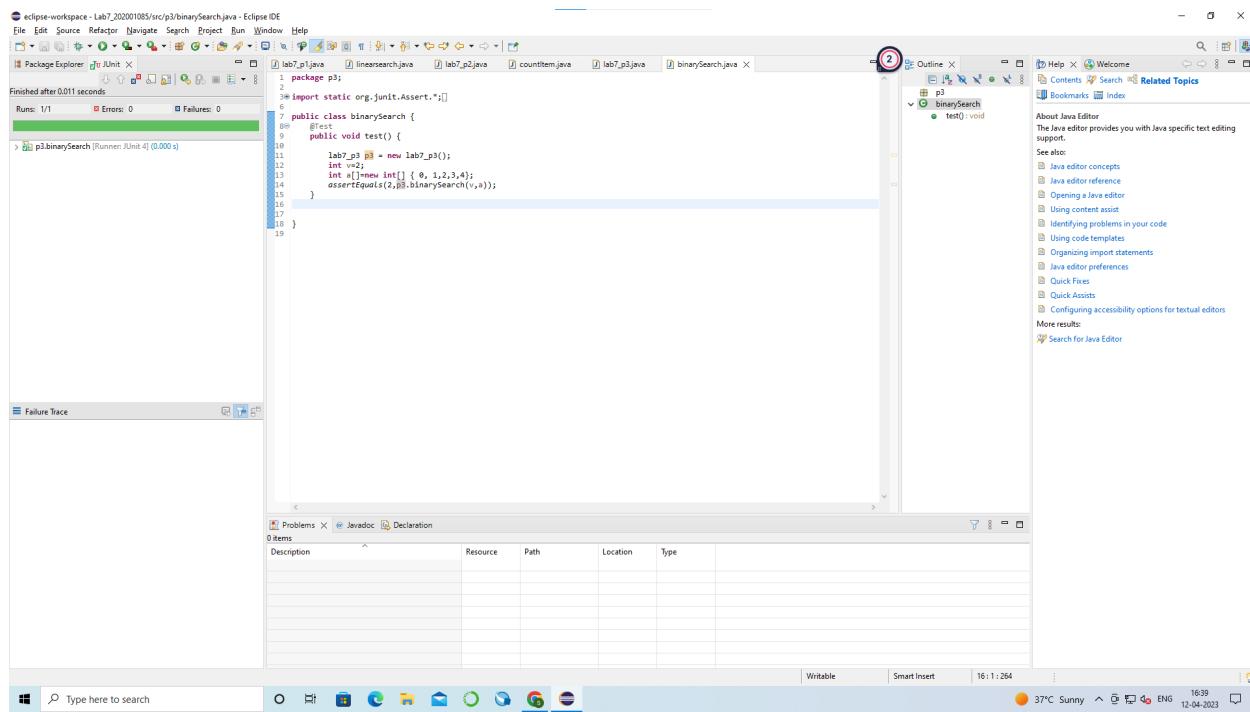
```

        hi = mid-1;
    else
        lo = mid+1;
    }
    return(-1);
}

```

Test Cases :

(1) v=2 , a= { 0, 1,2,3,4} expected output = 2



(2) v= -4 , a= {1,2,3,4,5 } expected output = -1

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Lab7_20200105/src/p3/binarySearch - Eclipse IDE
- Toolbar:** Standard Eclipse toolbar with icons for file operations, search, and navigation.
- Left Sidebar:** Package Explorer shows a single package "p3". JUnit View indicates 2 runs, 0 errors, and 0 failures.
- Central Area:** The code editor displays `binarySearch.java` containing Java code for a binary search algorithm and its unit tests. The code includes imports, class definitions, and two test methods: `test()` and `test2()`.
- Right Sidebar:** Outline view shows the project structure with `p3`, `binarySearch`, `test():void`, and `test2():void`. Help and Related Topics are visible at the top right.
- Bottom:** Status bar shows the date and time (16:10:270), and a system tray with icons for battery, temperature (37°C), and connectivity.

(3) v=5 , a= {2,3,4,5,5,6} expected output = 3 or 4

The screenshot shows the Eclipse IDE interface with the following details:

- Top Bar:** eclipse-workspace - Lab7_202001085/sc/p3/binarySearch.java - Eclipse IDE
- File Menu:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Includes icons for New, Open, Save, Cut, Copy, Paste, Find, Replace, and others.
- Package Explorer:** Shows p3 binarySearch (Runners: JUnit 4) with three tests: test (0.000 s), test2 (0.000 s), and test3 (0.001 s). Status: Finished after 0.06 seconds. Run: 3/3 Errors: 0 Failures: 1.
- Code Editor:** Displays the Java code for the binarySearch class. The code defines a binarySearch method that takes an array of integers and a target value, returning the index of the target or -1 if it's not found. It uses a recursive approach with base cases for empty arrays and arrays with one element. Test cases are provided for arrays of size 1, 2, 3, 4, and 5, comparing the result against expected values.
- Outline View:** Shows the structure of the p3 package, including the binarySearch class and its methods: test(), test2(), and test3().
- Help:** Provides links to Java editor concepts, Java editor reference, and other documentation.
- Failure Trace:** Shows a java.lang.AssertionError at line 31 of p3.binarySearch.test2().
- Bottom Bar:** Includes tabs for Problems, Javadoc, and Declaration. A search bar at the bottom left contains the placeholder "Type here to search".

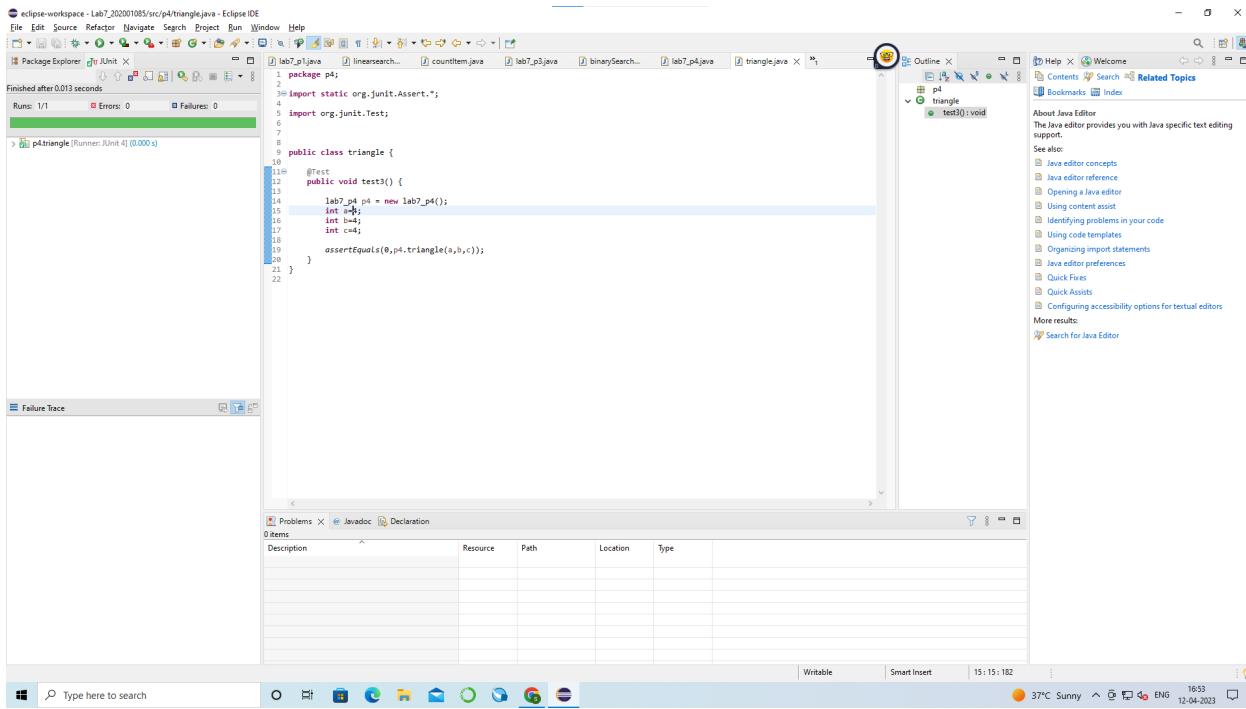
(P4). The following problem has been adapted from The Art of Software Testing, by G. Myers (1979). The function triangle takes three integer parameters that are interpreted as the lengths of the sides of a triangle. It returns whether the triangle is equilateral (three lengths equal), isosceles (two lengths equal), scalene (no lengths equal), or invalid (impossible lengths).

```
final int EQUILATERAL = 0;  
final int ISOSCELES = 1;  
final int SCALENE = 2;  
final int INVALID = 3;
```

```
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);  
}
```

Test Cases :

(1) a=4,b=4,c=4 expected output = EQUILATERAL



eclipse-workspace - Lab7_202001085/src/p4/triangle.java - Eclipse IDE

```

1 package p4;
2 import static org.junit.Assert.*;
3 import org.junit.Test;
4
5 public class triangle {
6
7     @Test
8     public void test3() {
9         lab7_p4 p4 = new lab7_p4();
10        int a=3;
11        int b=4;
12        int c=5;
13        assertEquals(0,p4.triangle(a,b,c));
14    }
15 }

```

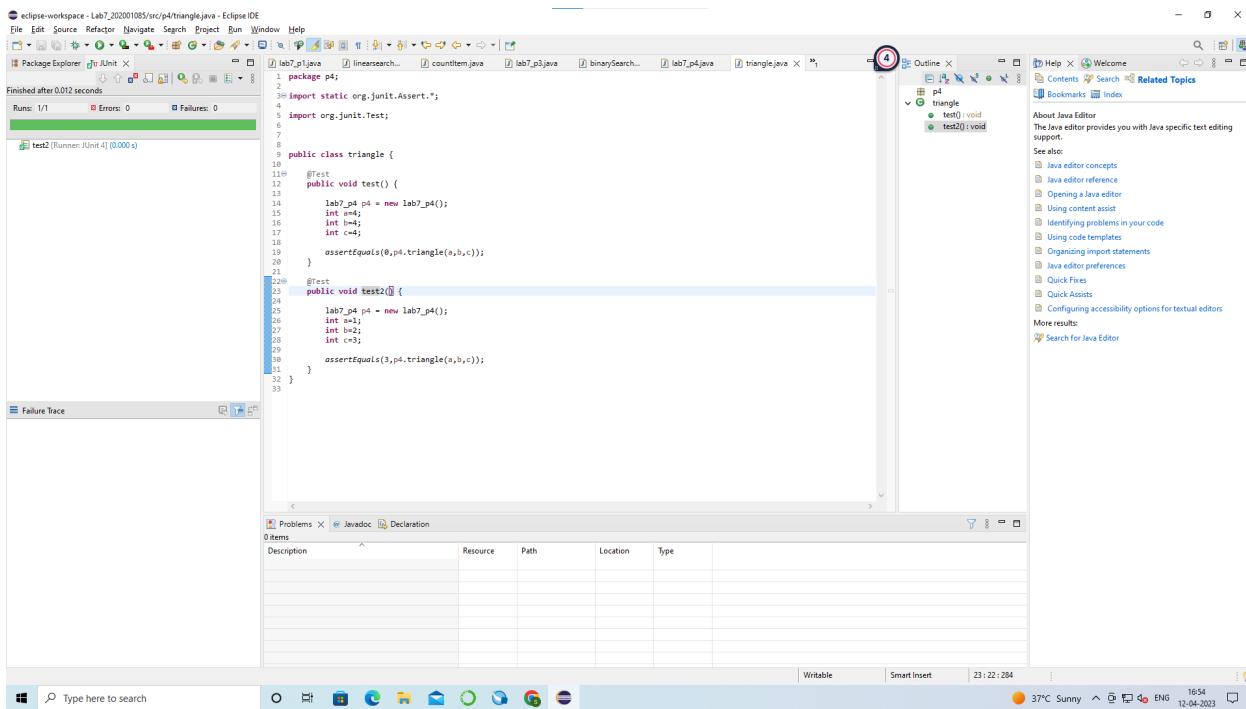
Outline View shows: triangle, test3();

Problems View: 0 items

Failure Trace: None

Java Editor Help: About Java Editor, See also: Java editor concepts, Java editor reference, Opening a Java editor, Using content assist, Identifying problems in your code, Using code templates, Organizing import statements, Java editor preferences, Quick Fixes, Quick Assists, Configuring accessibility options for textual editors.

(2) a=1,b=2,c=3 expected output= INVALID



eclipse-workspace - Lab7_202001085/src/p4/triangle.java - Eclipse IDE

```

1 package p4;
2 import static org.junit.Assert.*;
3 import org.junit.Test;
4
5 public class triangle {
6
7     @Test
8     public void test() {
9         lab7_p4 p4 = new lab7_p4();
10        int a=1;
11        int b=2;
12        int c=3;
13        assertEquals(0,p4.triangle(a,b,c));
14    }
15
16     @Test
17     public void test2() {
18         lab7_p4 p4 = new lab7_p4();
19        int a=1;
20        int b=2;
21        int c=2;
22        assertEquals(3,p4.triangle(a,b,c));
23    }
24 }

```

Outline View shows: triangle, test();, test2();

Problems View: 0 items

Failure Trace: None

Java Editor Help: About Java Editor, See also: Java editor concepts, Java editor reference, Opening a Java editor, Using content assist, Identifying problems in your code, Using code templates, Organizing import statements, Java editor preferences, Quick Fixes, Quick Assists, Configuring accessibility options for textual editors.

(3) a=-1,b=2,c=3 expected output = INVALID

The screenshot shows the Eclipse IDE interface with the following details:

- Top Bar:** File, Edit, Source, Refactor, Navigator, Search, Project, Run, Window, Help.
- Left Sidebar:** Package Explorer (JUnit), Problems, Javadoc, Declaration.
- Middle Area:** A code editor window containing Java code for a `triangle` class and its corresponding JUnit test cases (`test1`, `test2`, `test3`). The code includes imports for `org.junit.Assert` and `org.junit.Test`, and uses assertions like `assertEquals` and `assertTEquals`.
- Right Sidebar:** Outline view showing the project structure (`p4` and `triangle` package with methods `test1`, `test2`, `test3`).
- Help Bar:** Help, Welcome, Contents, Search, Related Topics, Bookmarks, Index.
- Bottom Status Bar:** Writable, Smart Insert, 36:16:480, 16:55, 37°C Sunny, ENG, 12-04-2023.

(4) a=3,b=4,c=5 expected output = SCALENE

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Lab7_202001085/src/p4/triangle.java - Eclipse IDE
- Toolbar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Package Explorer:** Shows p4 package with triangle.java and lab7_p4.java files.
- Outline View:** Shows the class structure with methods: p4, triangle, test1(), test2(), test3(), and test4().
- Java Editor:** Displays the Java code for triangle.java and its corresponding test methods (test1() through test4()). The code uses assertions to check the triangle function's results for different input values (a, b, c).
- Failure Trace:** Shows no errors or failures.
- Problems View:** Shows 0 items.
- Bottom Status Bar:** Writable, Smart Insert, 49:9:658, 16:56, 37°C Sunny, ENG, 12-04-2023.

(P6). Consider again the triangle classification program (P4) with a slightly different specification: The program reads floating values from the standard input. The three values A, B, and C are interpreted as representing the lengths of the sides of a triangle. The program then prints a message to the standard output that states whether the triangle, if it can be formed, is scalene, isosceles, equilateral, or right angled.

Determine the following for the above program:

a) Identify the equivalence classes for the system

Equivalence Classes:

EC1: All sides are positive, real numbers.

EC2: One or more sides are negative or zero.

EC3: The sum of the lengths of any two sides is less than or equal to the length of the remaining side (impossible lengths).

EC4: The sum of the lengths of any two sides is greater than the length of the remaining side (possible lengths).

b) Identify test cases to cover the identified equivalence classes.

Also, explicitly mention which test case would cover which equivalence class.

(Hint: you must need to be ensure that the identified set of test cases cover all identified equivalence classes)

Test cases:

TC1 (EC1): A=3, B=4, C=5 (right-angled triangle)

TC2 (EC1): A=5, B=5, C=5 (equilateral triangle)

TC3 (EC1): A=5, B=6, C=7 (scalene triangle)

TC4 (EC1): A=5, B=5, C=7 (isosceles triangle)

TC5 (EC2): A=-2, B=4, C=5 (invalid input)

TC6 (EC2): A=0, B=4, C=5 (invalid input)

c) For the boundary condition $A + B > C$ case (scalene triangle), identify test cases to verify the boundary.

Test cases for the boundary condition $A + B > C$:

TC7 (EC4): A=2, B=3, C=6 (sum of A and B is equal to C)

d) For the boundary condition $A = C$ case (isosceles triangle), identify test cases to verify the boundary.

Test cases for the boundary condition $A = C$:

TC8 (EC4): A=5, B=6, C=5 (A equals to C)

e) For the boundary condition $A = B = C$ case (equilateral triangle), identify test cases to verify the boundary.

Test cases for the boundary condition $A = B = C$:

TC9 (EC4): A=1, B=1, C=1 (all sides are equal)

f) For the boundary condition $A^2 + B^2 = C^2$ case (right-angle triangle), identify test cases to verify the boundary.

Test cases for the boundary condition $A^2 + B^2 = C^2$:

TC10 (EC4): A=3, B=4, C=5 (right-angled triangle)

g) For the non-triangle case, identify test cases to explore the boundary.

Test cases for the non-triangle case:

TC11 (EC3): A=2, B=2, C=4 (sum of A and B is less than C)

h) For non-positive input, identify test points.

Test points for non-positive input:

TP1 (EC2): A=0, B=4, C=5 (invalid input)

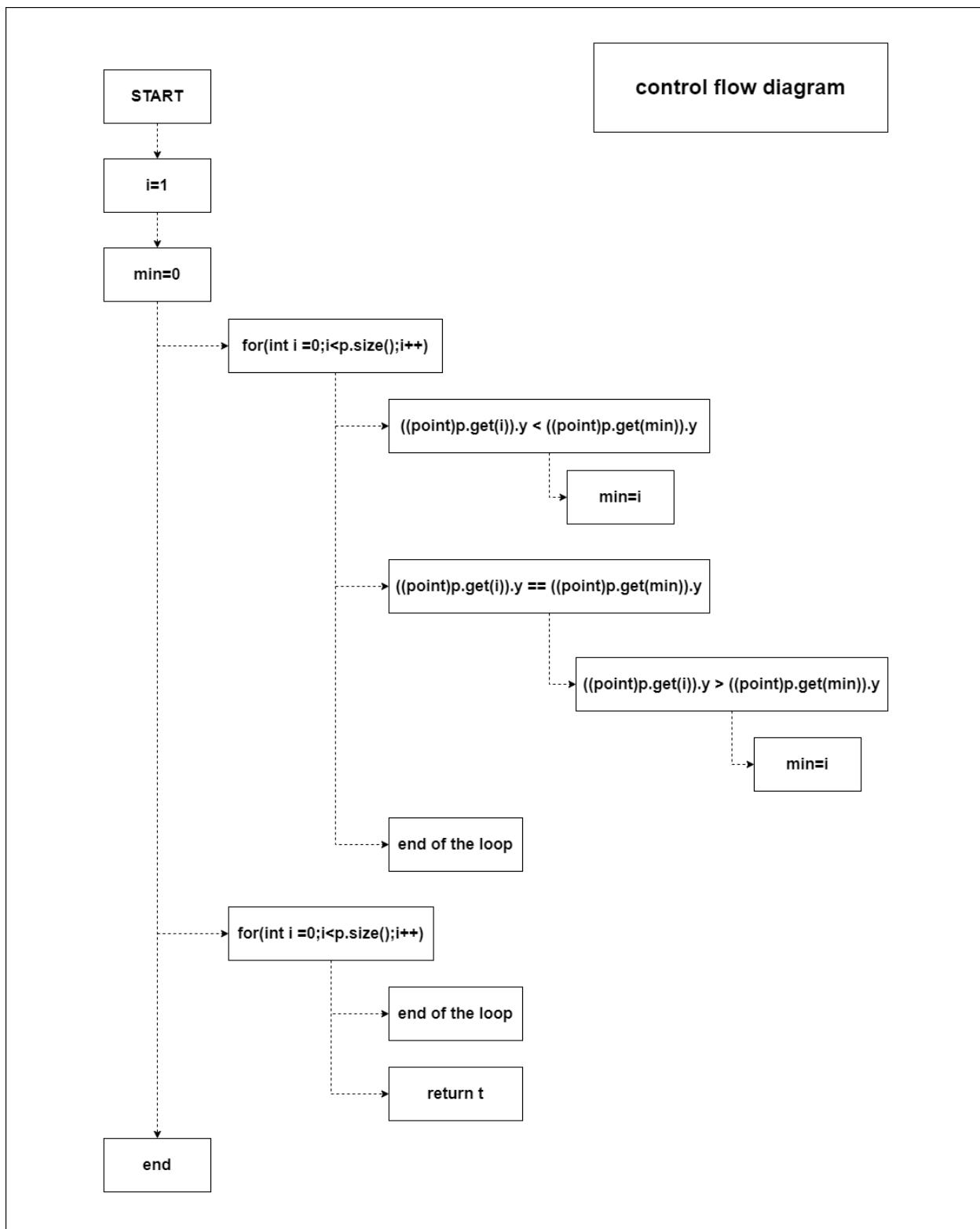
TP2 (EC2): A=-2, B=4, C=5 (invalid input)

Note: Test cases TC1 to TC10 covers all identified equivalence classes.

Section B :

- 1. Convert the Java code comprising the beginning of the doGraham method into a control flow graph (CFG).**

control flow diagram



2. Test sets:

a) Statement coverage test sets: To achieve statement coverage, we need to make sure that every statement in the code is executed at least once.

Test 1: p = empty vector

Test 2: p = vector with one point

Test 3: p = vector with two points with the same y component

Test 4: p = vector with two points with different y components

Test 5: p = vector with three or more points with different y components

Test 6: p = vector with three or more points with the same y component

b) Branch coverage test sets: To achieve branch coverage, we need to make sure that every possible branch in the code is taken at least once

Test 1: p = empty vector

Test 2: p = vector with one point

Test 3: p = vector with two points with the same y component

Test 4: p = vector with two points with different y components

Test 5: p = vector with three or more points with different y components, and none of them have the same x component

Test 6: p = vector with three or more points with the same y component, and some of them have the same x component

Test 7: p = vector with three or more points with the same y component, and all of them have the same x component

c) Basic condition coverage test sets: To achieve basic condition coverage, we need to make sure that every basic condition in the

code (i.e., every Boolean subexpression) is evaluated as both true and false at least once

Test 1: $p = \text{empty vector}$

Test 2: $p = \text{vector with one point}$

Test 3: $p = \text{vector with two points with the same y component, and the first point has a smaller x component}$

Test 4: $p = \text{vector with two points with the same y component, and the second point has a smaller x component}$

Test 5: $p = \text{vector with two points with different y components}$

Test 6: $p = \text{vector with three or more points with different y components, and none of them have the same x component}$

Test 7: $p = \text{vector with three or more points with the same y component, and some of them have the same x component}$

Test 8: $p = \text{vector with three or more points with the same y component, and all of them have the same x component.}$