

Part 1: Test Cases:

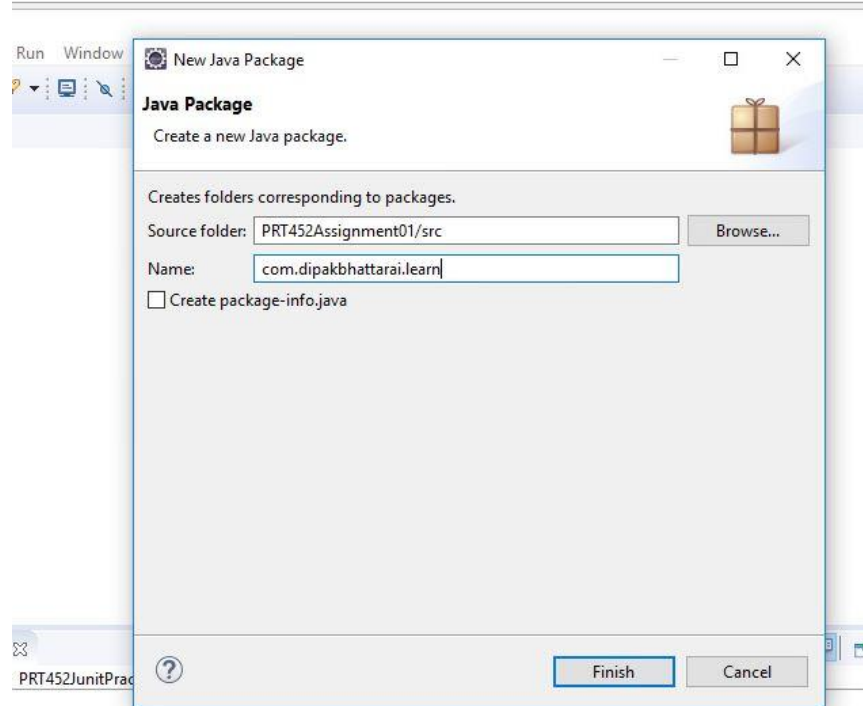
TDD Test Cases for gradient, distance and equation of line between two points:

S.N	Test Case Description	Input				Expected Output			Remarks
		X1	Y1	X2	Y2	Gradient (y2-y1)/ (x2-x1)	Distance $\sqrt{(x2-x1)^2 + (y2-y1)^2}$	Equation $y-y1=m(x-x1)$ or $y=mx +c$	
1	User enters the coordinates A(0,0), B(0,0)	0	0	0	0	0	0	0	
2	User enters the coordinates A(1,1), B(1,1)	1	1	1	1	unidentified	0	Y=0	
3	User enters the coordinates A(-2,1), B(-1,1)	-2	1	-1	1	1	0	Y=1	
4	User enters the coordinates A(4,3), B(6,7)	4	3	6	7	2	$\sqrt{20}$	Y=2x - 5	
6	User enters the coordinates A(0,0), B(0,0)	2	4	4	6	1	$\sqrt{8}$	$y = x + 2$	

Part 2: Program developing screenshots:

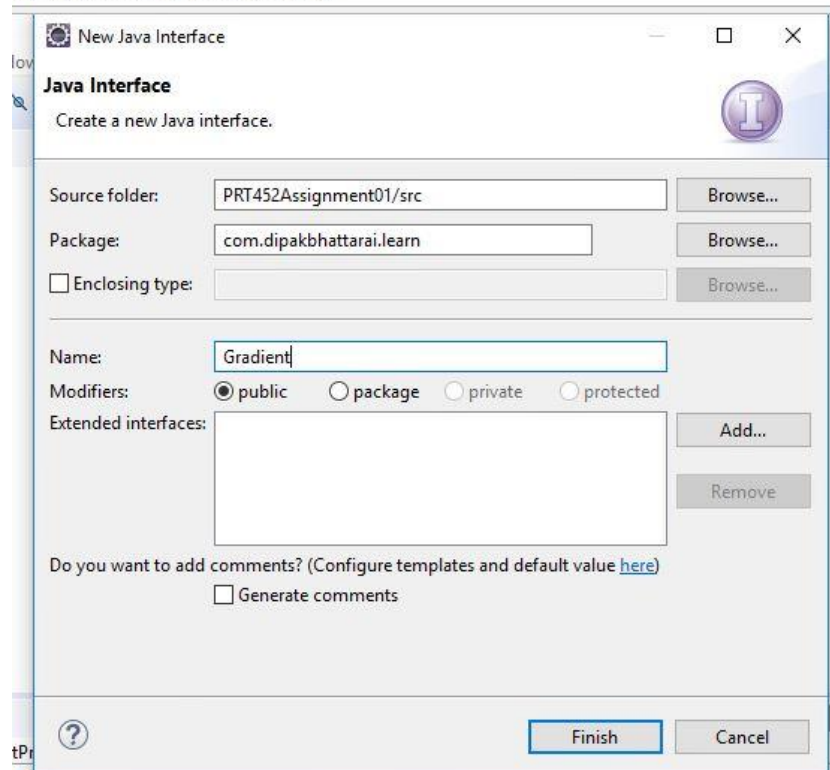
Step 1: The project PRT452Assignment01 has been created and the package com.dipakbhattarai.learn is created.

edu.au/portal/webclient/index.html#/desktop



Step 2: The Interface Gradient is created. Similarly, the interfaces Distance and Equation are also created.

tal/webclient/index.html#/desktop



Step 3: The interfaces are created with the methods with their respective parameters.

Interface1: Gradient and its method with parameters.

m/dipakbhattarai/learn/Gradient.java - Eclipse

```
Project Run Window Help
[Icons]
Gradient.java Distance.java Equation.java Mathematics.java
1 package com.dipakbhattarai.learn;
2
3 public interface Gradient {
4
5     double gardient (double x1, double y1, double x2, double y2);
6 }
7
```

Interface2: Distance and its method with parameters.

m/dipakbhattarai/learn/Distance.java - Eclipse

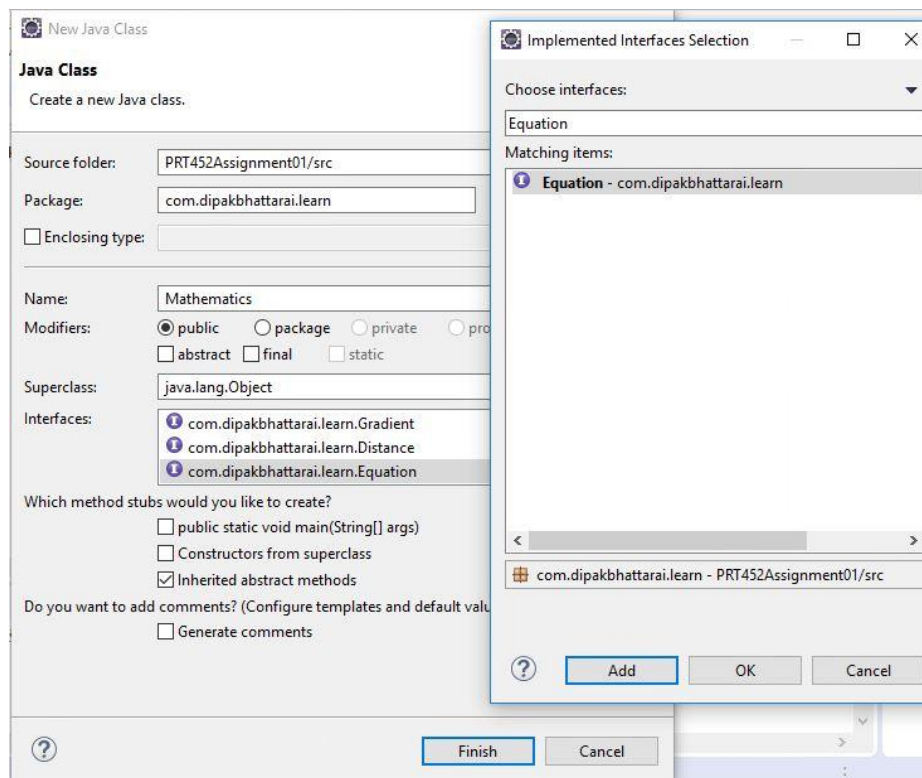
```
Project Run Window Help
[Icons]
Gradient.java Distance.java Equation.java Mathematics.java
1 package com.dipakbhattarai.learn;
2
3 public interface Distance {
4
5     double distance (double x1, double y1, double x2, double y2);
6 }
7
```

Interface3: Equation and its method with parameters.

uation.java - Eclipse

```
v Help
[Icons]
Gradient.java Distance.java Equation.java Mathematics.java
1 package com.dipakbhattarai.learn;
2
3 public interface Equation {
4
5     String equation (double x1, double y1, double gradient);
6 }
7
```

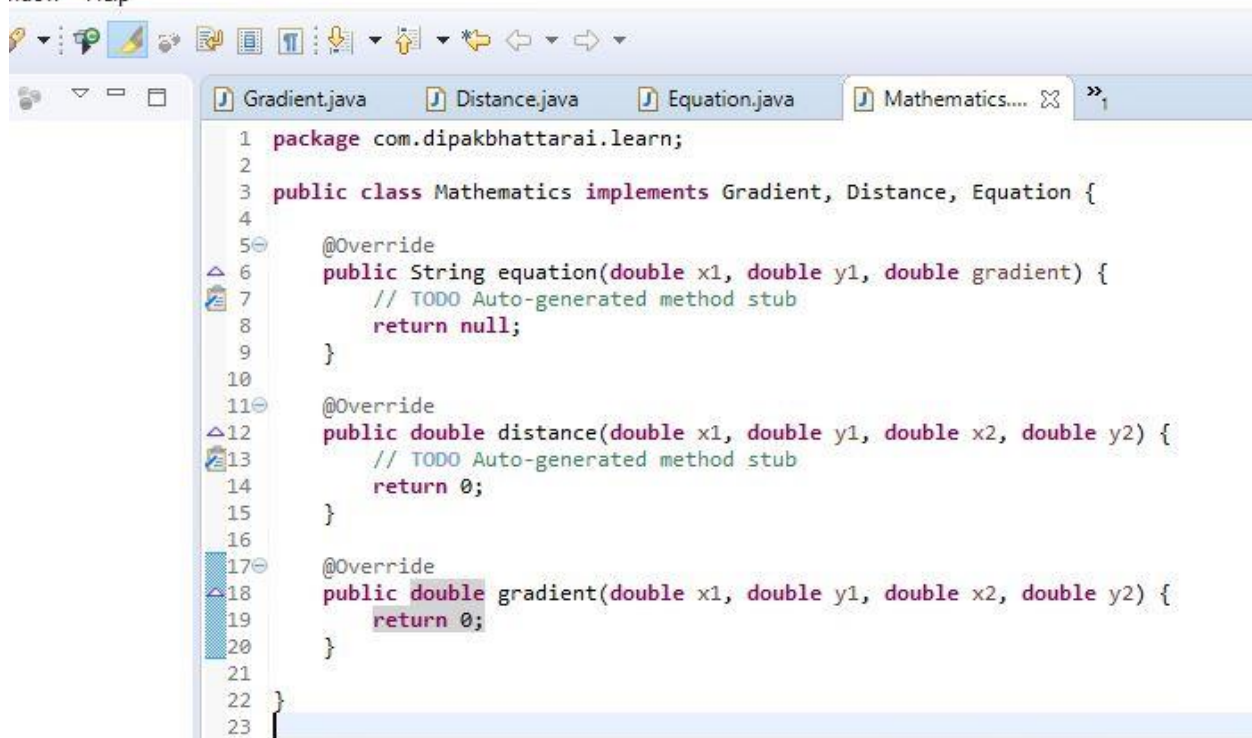
Steps 4: The java file "Mathematics.java" is created and the three interfaces are also implemented as shown in the picture:



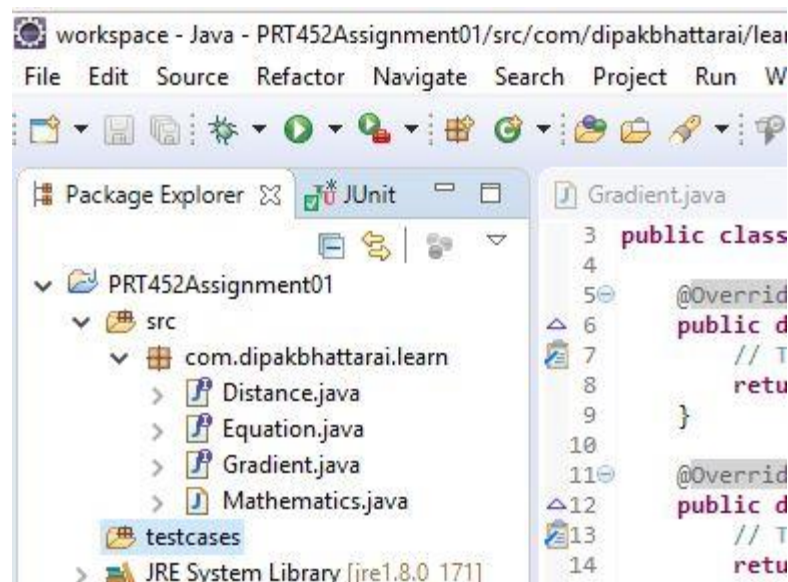
Output: The class Mathematics.java is created with its methods implemented from the interfaces

n/Mathematics.java - Eclipse

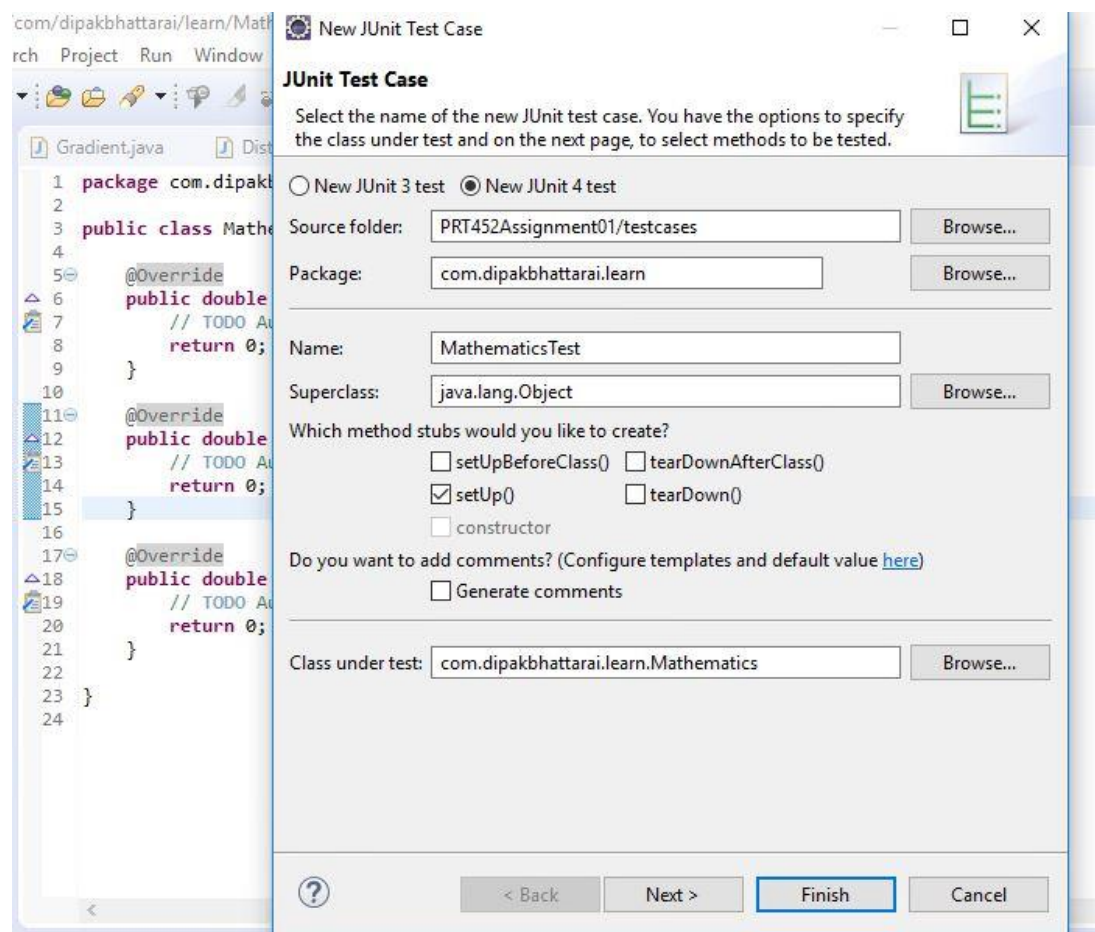
indow Help



Steps 5: Creating folder testcases in order to separate the test file.



Steps 6: Creating the JUnit Testing of the file Mathematics.java



The follow picture shows the JunitTesting of the methods gradient, distance and equation:

```

1
2
3 import static org.junit.Assert.*;
4
5
6
7
8 public class MathematicsTest {
9
10     @Before
11     public void setUp() throws Exception {
12     }
13
14     @Test
15     public void test() {
16         fail("Not yet implemented");
17     }
18
19     @Test
20     public void testGradient(){
21         fail("Not yet implemented");
22     }
23
24     @Test
25     public void testDistance(){
26         fail("Not yet implemented");
27     }
28
29     @Test
30     public void testEquation(){
31         fail("Not yet implemented");
32     }
33
34 }
35

```

Steps 7: The testing fail showing the unmatched result.

```

13
14 @Before
15 public void setUp() throws Exception {
16     classUnderTest = new Mathematics();
17 }
18
19 @Test
20 public void testGradient(){
21     double x1 = 10;
22     double y1 = 12;
23     double x2 = 12;
24     double y2 = 15;
25
26     double result = (y2 - y1)/(x2 - x1);
27
28     //compare to Mathematics java
29     assertEquals(result, classUnderTest.gradient(10, 12, 12, 15), 0.001);
30 }
31
32 @Test
33 public void testDistance(){
34     fail("Not yet implemented");
35 }
36
37 @Test
38 public void testEquation(){
39     fail("Not yet implemented");
40 }
41

```

Package Explorer: JUnit

Finished after 0.021 seconds

Runs: 3/3 Errors: 0 Failures: 3

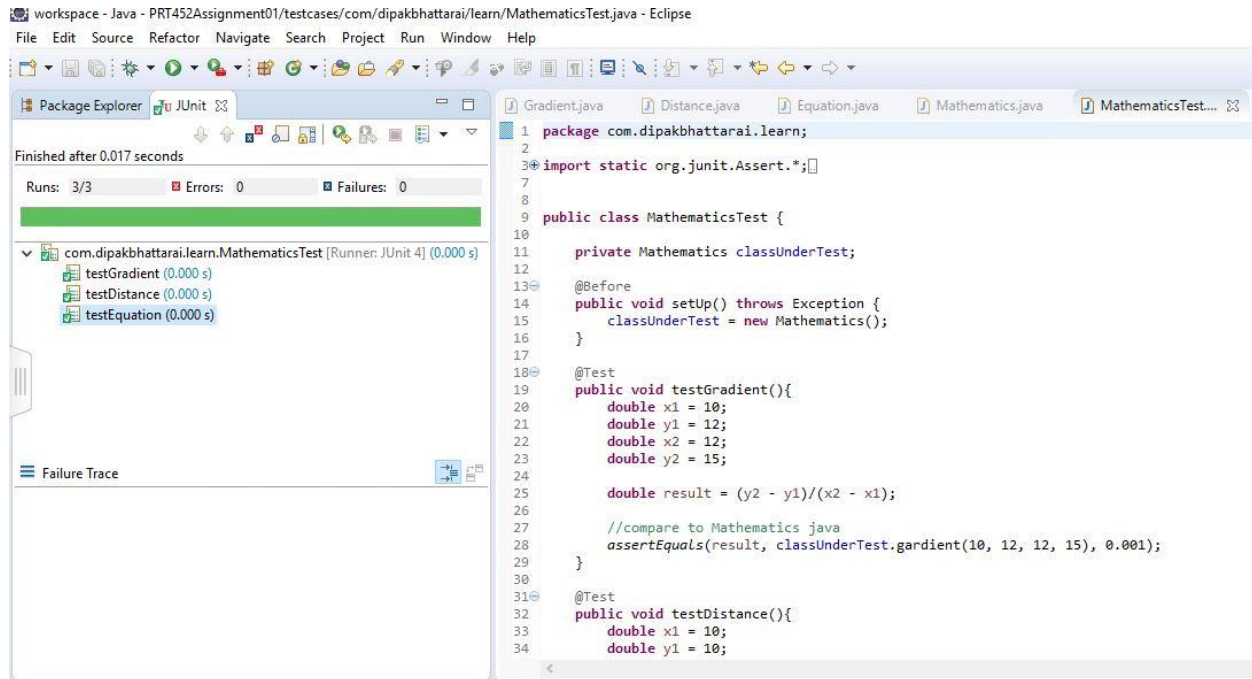
com.dipakbhattarai.learn.MathematicsTest [Runner: JUnit 4] (0.000 s)

- testGradient (0.000 s)
- testDistance (0.000 s)
- testEquation (0.000 s)

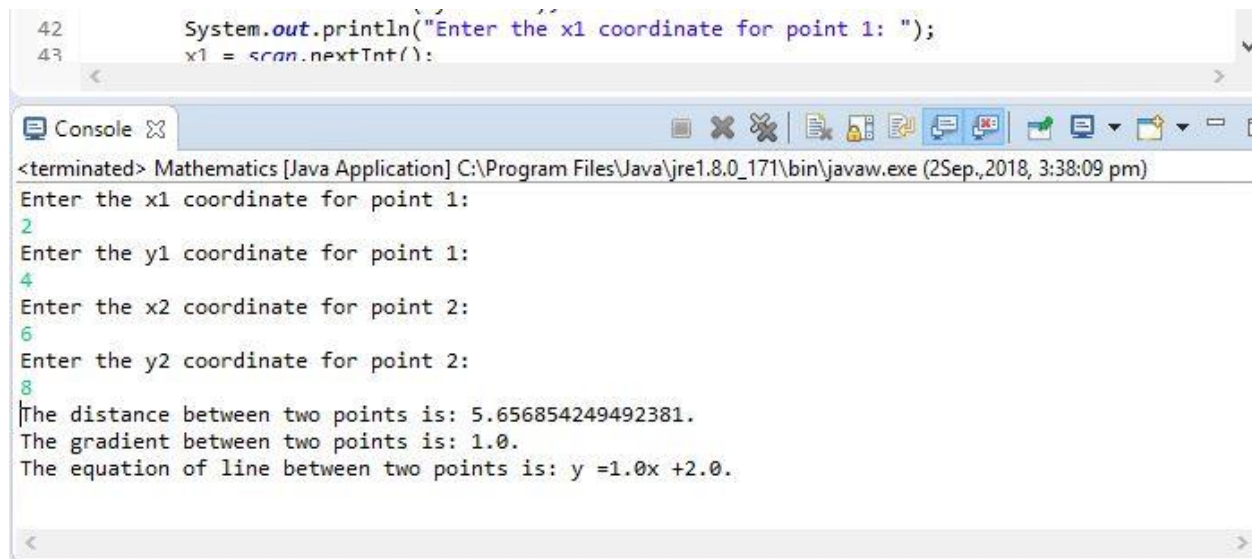
Failure Trace

java.lang.AssertionError: expected:<1.5> but was:<0.0>
 at com.dipakbhattarai.learn.MathematicsTest.testGradient(MathematicsTest.java:29)

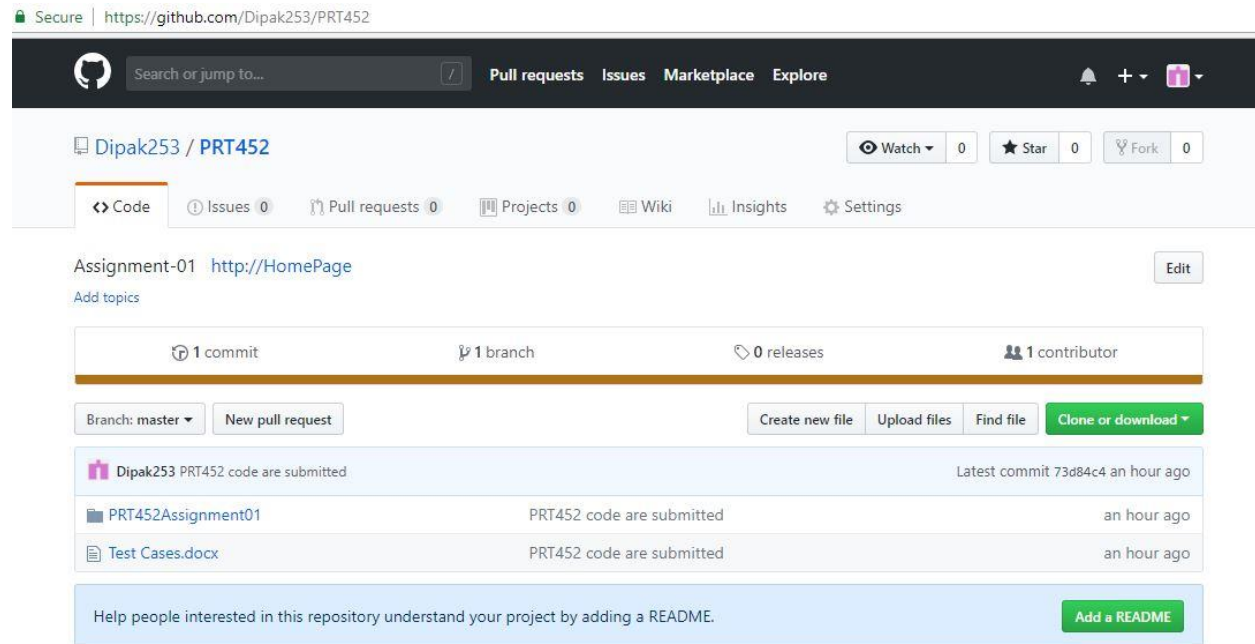
Steps 8: After implementing the codes the test is finally passed.



Steps 9: The final output of the program when user enters the two points.



Steps 10: Creating the github repository for the assignment.



Part 3: Code Smells

The five issues related to code smells are identified and the solution for them are discussed below as:

- 1) **Lazy Class:** Creating a class costs money. If the class which is not doing enough should be eliminated.

Solution:

- **Collapse Hierarchy:** It will collapse all the subclasses that aren't doing enough.
- **Inline Component:** The useless components should be subjected using this method.

- 2) **Temporary Field:** An object needs its variables to be instantiated but sometimes some variables are only set for certain circumstances. Such code is difficult to understand.

Solution:

- **Extract Component:** It helps to collect all the concerns variables in the component.
- **Introduce Null Object:** It helps to eliminate conditional code and create an alternative component for invalid variables.

3) **Middle Man:** Sometimes the internal details of objects are encapsulated.

Solution:

- **Move Method** and **Move Field:** These methods are used to move features out of the middle man into the other objects making the middle man empty.

4) **Large Class:** In a large class, there are a number of objects and its variables but all these variables are not used at all.

Solution:

- **Extract Component:** It helps to bundle up the variables and
- **Extract subclass:** If the subset of variables are constant then Extract subclass can be used to overcome this problem

5) **Duplicated code:** It means repetitive use of the code in a program which is not a good practice.

Solution:

- **Extract Method:** This method is used to separate the similar codes and invoke the code from other place.