**Assignment 4**

**Rasika Mohod**

rmohod@gmu.edu

G01044774

**Source code files:**

**LineSegment.java**

**package** com.gmu;

**import** java.awt.Point;

**public** **class** LineSegment {

/\*

\* This class represents the mathematical concept of a line segment.

\*

\* Specification fields

\* Point(x,y) represents a point with integer x and y coordinates.

\* @ startPoint : Point // The starting point of the line segment.

\* @ endPoint : Point // The ending point of the line segment.

\*

\* Derived specification fields:

\* @length : real //length = sqrt((startPoint.x - endPoint.x)^2 + (startPoint.y - endPoint.y)^2)

\* // The length of the line segment.

\*/

/\* Abstraction Function:

\* AF(c) = a line segment l,

\* where l.startPoint = (c.startPointx, c.startPointy)

\* l.endPoint = (c.endPointx, c.endPointy)

\*

\* Representation Invariant:

\* (startPointx != endPointx && startPointy != endPointy)

\* Both startPoint and endPoint should not be same

\*

\*/

**private** Point startPoint,endPoint;

//Constructor

**public** LineSegment(**int** x1,**int** y1,**int** x2,**int** y2)

{

//EFFECTS: Initializes the seatPoint and endPoint

// & Sets the value of startPoint and endPoint of a line segment.

startPoint = **new** Point();

endPoint = **new** Point();

Point tempStartPoint = **new** Point (x1,y1);

Point tempEndPoint = **new** Point (x2,y2);

setStartPoint(tempStartPoint);

setEndPoint(tempEndPoint);

}

//Method : to check if representation is correct

**public** **void** RepOk() **throws** Exception

{

//EFFECTS: Nothing if rep invariant is satisfied else throws an exception

Point tempStartPoint = getStartPoint();

Point tempEndPoint = getEndPoint();

**if**((tempStartPoint.x == tempEndPoint.x) && (tempStartPoint.y == tempEndPoint.y))

{

**throw** **new** Exception("Rep is not correct!! Both the start and end points are same.");

}

}

//Setter Methods

**public** **void** setStartPoint(Point tempPoint)

{

//MODIFIES: startPoint

//EFFECTS: set the coordinates of a startPoint of line segment

startPoint=tempPoint;

}

**public** **void** setEndPoint(Point tempPoint)

{

//MODIFIES: endPoint

//EFFECTS: set the coordinates of a endPoint of line segment

endPoint=tempPoint;

}

//Getter Methods

**public** Point getStartPoint()

{

//MODIFIES: Nothing

//EFFECTS: returns the startPoint of line segment

**return** startPoint;

}

**public** Point getEndPoint()

{

//MODIFIES: Nothing

//EFFECTS: returns the endPoint of line segment

**return** endPoint;

}

//Method: To calculate length of a line segment

**public** **double** lineSegmentLength() **throws** Exception

{

//REQUIRES: rep invariant to be satidfied

//MODIFIES: Nothing

//EFFECTS: returns the length of line segment

//calling RepOk() before length calculation to check if rep invariant is satisfied

RepOk();

**double** length;

Point tempStartPoint = getStartPoint();

Point tempEndPoint = getEndPoint();

//Calculating the length of a line segment

length = ((tempStartPoint.x - tempEndPoint.x)\*(tempStartPoint.x - tempEndPoint.x)) +

((tempStartPoint.y - tempEndPoint.y)\*(tempStartPoint.y - tempEndPoint.y));

length = (**double**)Math.*sqrt*(length);

**return** length;

}

//Method: To print the length of the line segment

**public** **void** printSegmentLength() **throws** Exception

{

//MODIFIES: Nothing

//EFFECTS: Prints the length of line segment

System.***out***.println("Length of the line segment is: "+lineSegmentLength());

}

}

**LineSegmentClient.java**

**package** com.gmu;

//Client class used to input the data required to calculate the length of a line segment

**public** **class** LineSegmentClient {

**public** **static** **void** main(String args[])

{

**try**{

//Test input: with 2 different points

System.***out***.println("LineSegment 1=> ");

LineSegment lineSegment1 = **new** LineSegment(1,3,2,5);

lineSegment1.printSegmentLength();

//Test input: with 2 different points

System.***out***.println("\nLineSegment 2=> ");

LineSegment lineSegment2 = **new** LineSegment(-2,4,2,9);

lineSegment2.printSegmentLength();

//Test input: with 2 same points

System.***out***.println("\nLineSegment 3=> ");

LineSegment lineSegment3 = **new** LineSegment(-2,4,-2,4);

lineSegment3.printSegmentLength();

}

**catch**(Exception e)

{

System.***out***.println("Exception occured: "+e);

}

}

}

**Output:**

LineSegment 1=>

Length of the line segment is: 2.23606797749979

LineSegment 2=>

Length of the line segment is: 6.4031242374328485

LineSegment 3=>

Exception occured: java.lang.Exception: Rep is not correct!! Both the start and end points are same.

**JUnitTest.java**

package test;

import static org.junit.Assert.\*;

import org.junit.Test;

import com.gmu.LineSegment;

public class JUnitTest {

@Test

//Test case written to be successful

public void test1() throws Exception {

LineSegment lineSegment = new LineSegment(-2,4,2,9);

double calculatedLength = lineSegment.lineSegmentLength();

double expectedLength = 6.4031242374328485;

if(calculatedLength!=expectedLength)

{

assertTrue(false);

fail("Test case Failed as the computed length of line segment is not correct.");

}

}

@Test

//Test case written to be unsuccessful

public void test2() throws Exception {

LineSegment lineSegment = new LineSegment(1,3,2,5);

double calculatedLength = lineSegment.lineSegmentLength();

double expectedLength = 6.4031242374328485;

if(calculatedLength!=expectedLength)

{

assertTrue(false);

fail("Test case Failed as the computed length of line segment is not correct.");

}

}

}