CSC108 – Introduction to Programming

Lab09A

**Point Class:**

Design a class named **Point** to represent a point with x- and y- coordinates. The class contains:

The private data fields **x** and **y** that represent the coordinates with getter methods. Use the “this” keyword to reference the object’s instance members (e.g. this.X, this.Y) and to invoke another constructor of the same class, for example,

public Point(){

this(0,0);

}

* A no-arg constructor that creates a point (0,0).
* A constructor that constructs a point with specific coordinates.
* A static method named **distance** that returns the distance from this point object to another point object with specific x and y values.

Open Microsoft Word, and create the UML diagram for the class and then implement the class. Write a test program called **TestPoint** that asks for the coordinates of two points, creates two point objects, and then displays the distance between them. You should use the Pythagorean Theorem to calculate this distance (see below). You should use the following showInputDialog method and the showMessageDialog method in the JOptionPane class and the StringTokenizer class from the java.util package along with the parseInt method from the Integer wrapper class (see example below):

String input = JOptionPane.showInputDialog("Enter x1 y1 x2 y2:");

StringTokenizer st = new StringTokenizer(input, " ");

int x1 = Integer.parseInt(st.nextToken());

String output = "The distance between the two points…

JOptionPane.showMessageDialog(null, output);

(x2,y2)

For a right triangle:

(4,3)

c

b

(0,0)

a

(x1,y1)

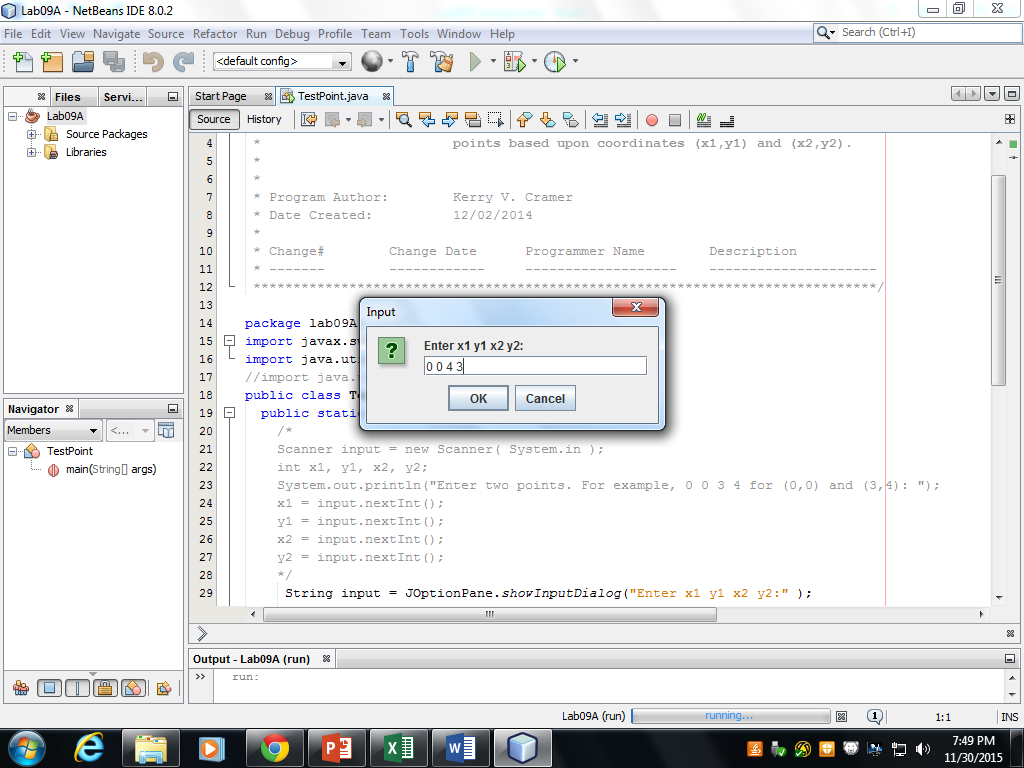
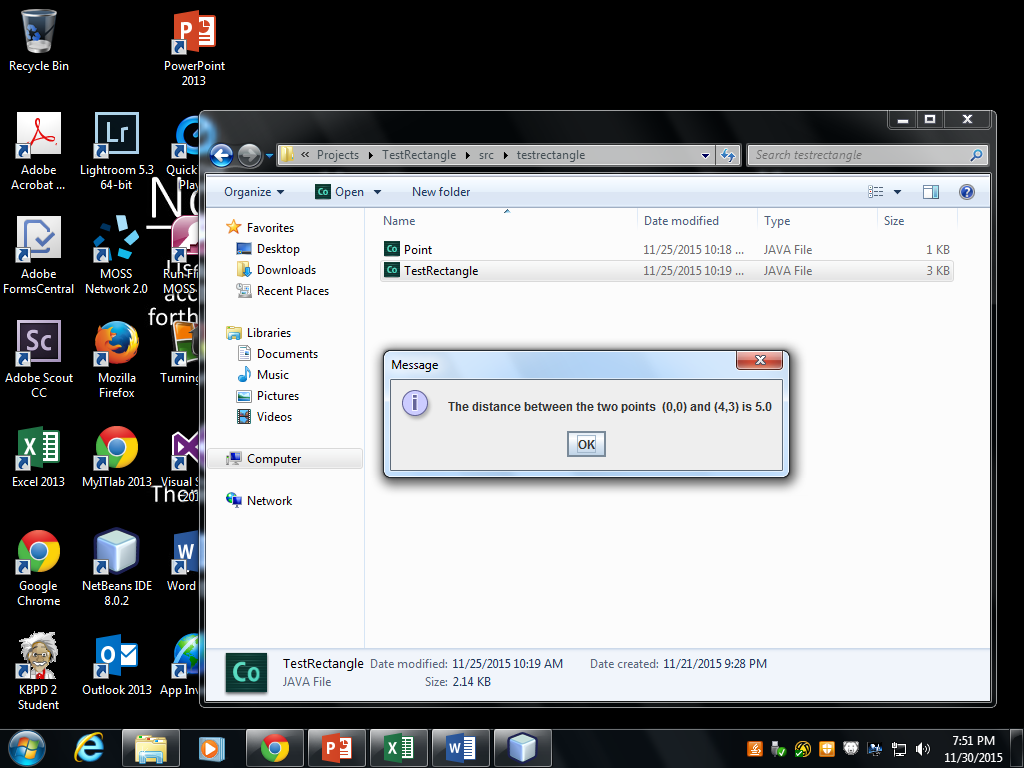
c2 = a2 + b2

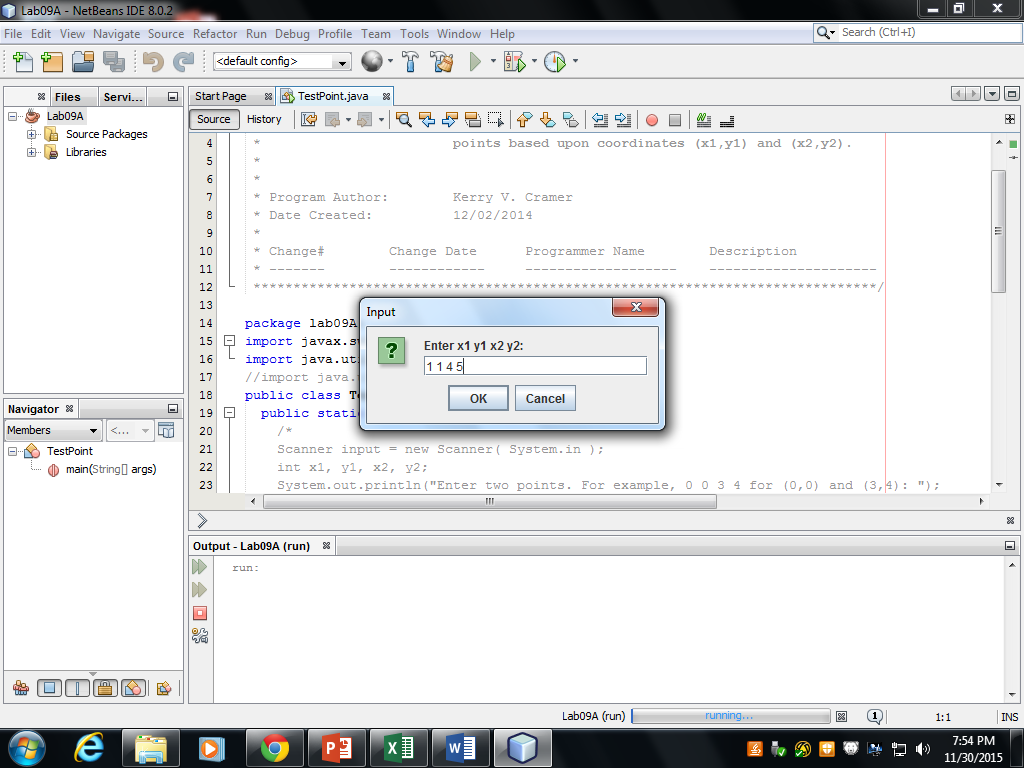
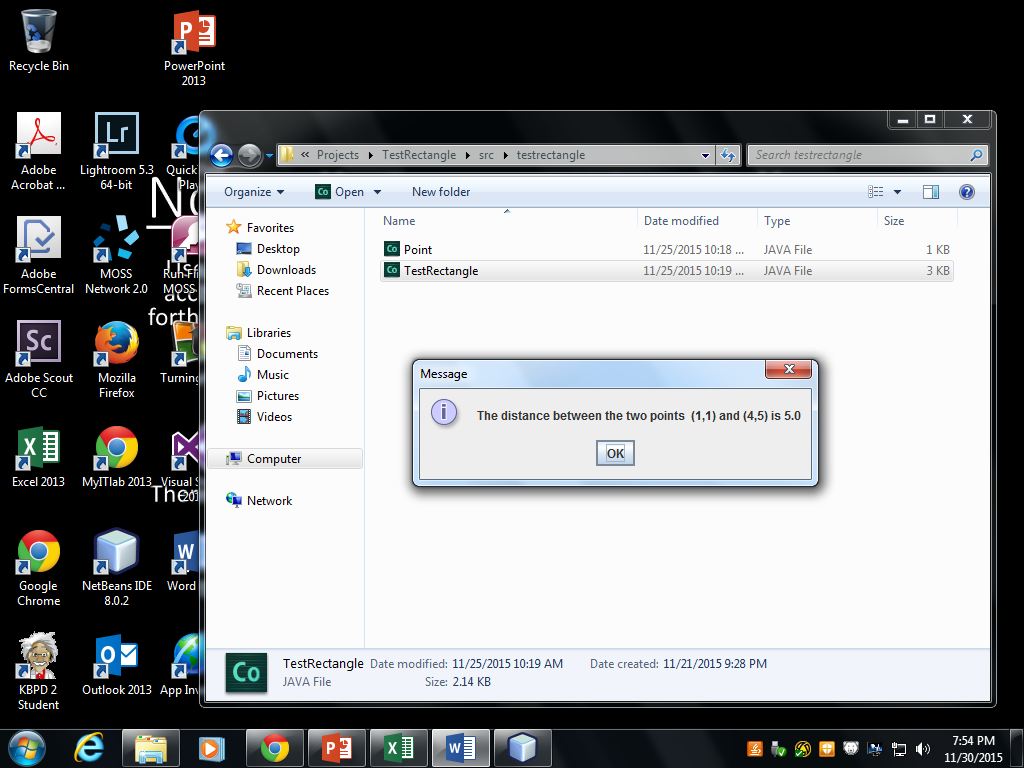
The distance between the two points: c = = = = 5

c =

c = = = = 5

Sample Java output:

***Include the UML diagram*** with the zipped Lab09A folder and send it to me as an attachment in a Blackboard message.

public class main {

public static void main (String[] args) {

MyPoint a = new MyPoint(0, 0);

MyPoint b = new MyPoint(10, 30.5);

System.out.printf(

"The distance between the points :" x, y, y.distance(a));

}

}

class MyPoint {

private double \_\_x;

private double \_\_y;

public Point() {

this(0, 0);

}

public Point(double x, double y) {

this.\_\_x = x;

this.\_\_y = y;

}

/\*\*

\* Calculate the distance between this instance of Point and the

\* provided coordinates.

\*

\*/

public double distance(double x, double y) {

// d = sqrt((x2 - x1)^2 + (y2 - y1)^2)

return Math.sqrt(

(x - this.\_\_x)\*(x - this.\_\_x) +

(y - this.\_\_y)\*(y - this.\_\_y));

}

/\*\*

\* Calculate the distance between this and another instance of

\* Point.

\*

\* Calls the other Point instance's distance(double, double)

\* method with the calling instance's coordinates.

\* I.E: `p1.distance(p2)` is equivalent to

\* `p2.distance(p1.\_\_x, p1.\_\_y)`.

\*

\*/

public double distance(Point other) {

return other.distance(this.\_\_x, this.\_\_y);

}

/\*\*

\* Distance from (0, 0)

\*

\*/

public double distance() {

return this.distance(0.0, 0.0);

}

/\*\*

\* Returns a constructor statement for this instance of Point.

\*

\*/

public String \_\_repr\_\_() {

return String.format("MyPoint(%f, %f)", this.\_\_x, this.\_\_y);

}

/\*\*

\* Returns a human readable representation of Point.

\*

\*/

public String toString() {

return String.format("(%f, %f)", this.\_\_x, this.\_\_y);

}

/\*\*

\* Completely pointlessly fetch the x-coordinate of Point.

\*

\*/

//public double getX() {

// return this.\_\_x;

//}

/\*\*

\* Completely pointlessly fetch the y-coordinate of Point.

\*

\*/

//public double getY() {

// return this.\_\_y;

//}

}