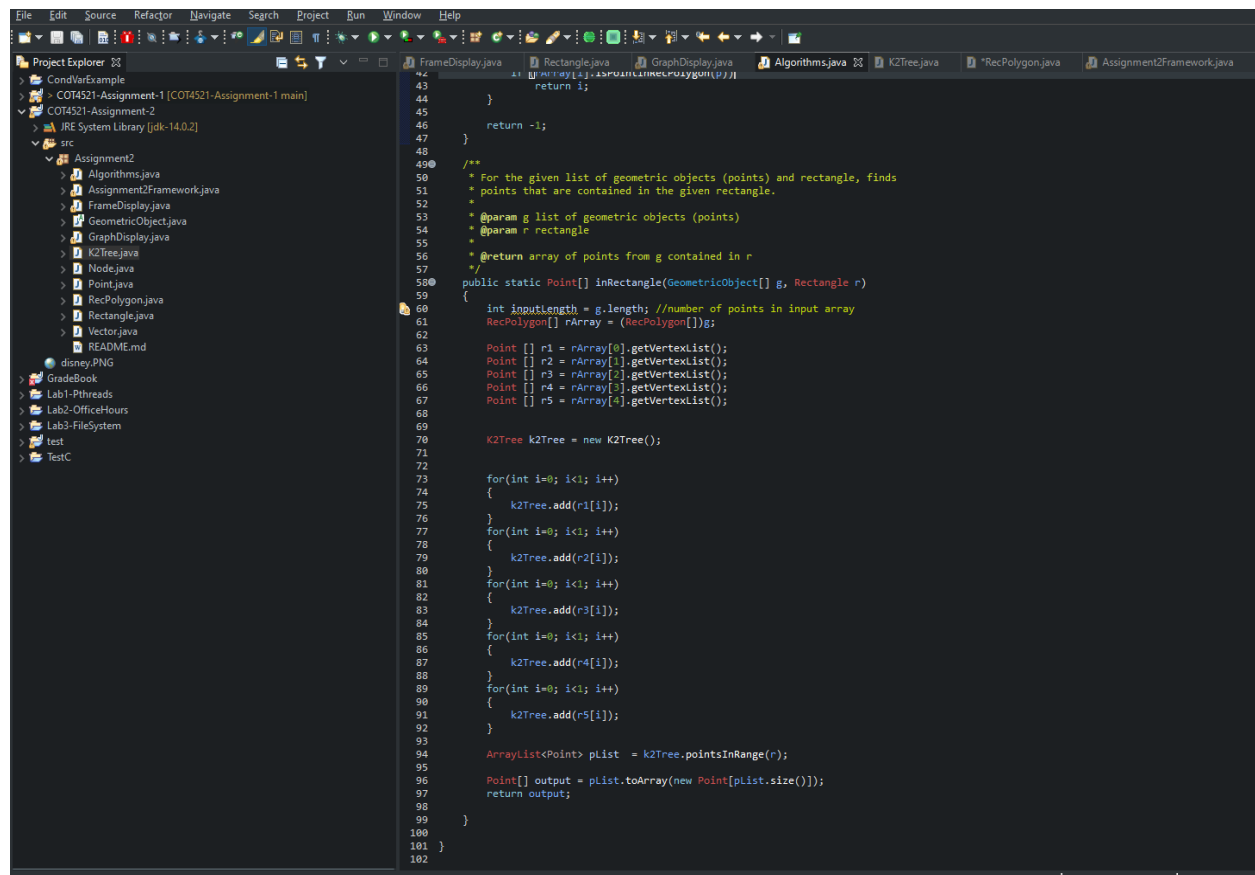


Project 2 was full of challenges that I had while implementing the solution. The most challenging component of the project was figuring out how the k2tree, that stored all the rectilinear polygons, knew from which rectilinear polygon object it came from. That is to say when the point search was done on the k2tree, it would need to differentiate specific points that would come from 1 rectilinear polygon geometric object. I would say that reviewing the lecture slides and the in class lectures helped with this problem to an extent. Below are three images, two show the project 2 code in the Eclipse IDE, and the other shows the map when the program is executed to do the orthogonal search on the rectilinear polygon objects.



```
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
  CondVarExample
  COT4521-Assignment-1 [COT4521-Assignment-1 main]
  COT4521-Assignment-2
    JRE System Library [jdk-14.0.2]
    src
      Assignment2
        Algorithms.java
        Assignment2Framework.java
        FrameDisplay.java
        GeometricObject.java
        GraphDisplay.java
        K2Tree.java
        Node.java
        Point.java
        RectPolygon.java
        Rectangle.java
        Vector.java
        README.md
      disney.PNG
      GradeBook
      Lab1-Pthreads
      Lab2-OfficeHours
      Lab3-Filesystem
      test
      TestC
  K2Tree.java
  RectPolygon.java
  Assignment2Framework.java

42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102

/**
 * For the given list of geometric objects (points) and rectangle, finds
 * points that are contained in the given rectangle.
 *
 * @param g list of geometric objects (points)
 * @param r rectangle
 *
 * @return array of points from g contained in r
 */
public static Point[] inRectangle(GeometricObject[] g, Rectangle r)
{
    int inputLength = g.length; //number of points in input array
    RectPolygon[] rArray = (RectPolygon[])g;

    Point[] r1 = rArray[0].getVertexList();
    Point[] r2 = rArray[1].getVertexList();
    Point[] r3 = rArray[2].getVertexList();
    Point[] r4 = rArray[3].getVertexList();
    Point[] r5 = rArray[4].getVertexList();

    K2Tree k2Tree = new K2Tree();

    for(int i=0; i<l; i++)
    {
        k2Tree.add(r1[i]);
    }
    for(int i=0; i<l; i++)
    {
        k2Tree.add(r2[i]);
    }
    for(int i=0; i<l; i++)
    {
        k2Tree.add(r3[i]);
    }
    for(int i=0; i<l; i++)
    {
        k2Tree.add(r4[i]);
    }
    for(int i=0; i<l; i++)
    {
        k2Tree.add(r5[i]);
    }

    ArrayList<Point> pList = k2Tree.pointsInRange(r);

    Point[] output = pList.toArray(new Point[pList.size()]);
    return output;
}
```

