HW4-Observation Stations

2024.3.18

112-2 PDSA HW4

Release date: 03/18 16:00
Due date: 03/27 21:00

HW4 - Observation Stations

In this assignment, we delve into a scenario where a group of ecologists are assessing potential sites for establishing observation stations. The primary goal is to determine how these stations should be strategically placed to facilitate ecological studies.

Description

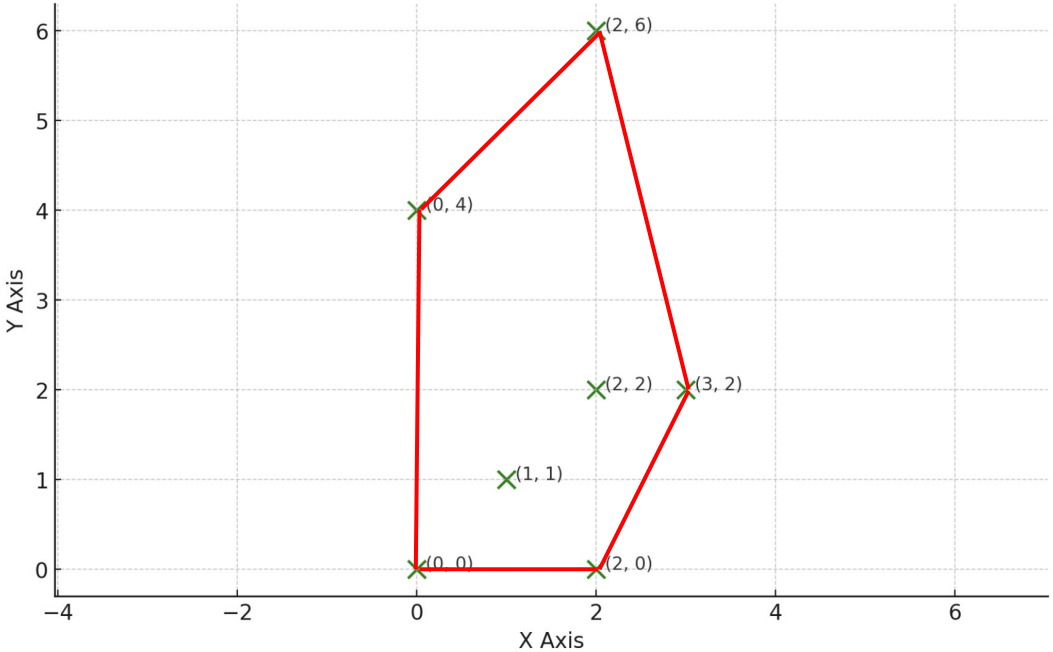
Your mission is to assist these ecologists doing their research. You will receive coordinates representing potential stations and are required to investigate:

- **Farthest Observation Stations:** Determine the two station locations that are geographically most distant from each other. This analysis will help in assessing the spatial extent of the ecological study area.
- **Ecological Area Coverage:** Calculate the coverage area for the convex polygon formed by existing stations. This will help in understanding how effectively the network of stations can monitor and represent the region's ecological diversity.
- **A New Observation Station:** Assess how integrating a new station into the existing network alters the overall spatial arrangement and ecological coverage. This evaluation will provide insights into enhancing the network for comprehensive ecological observation.

Example

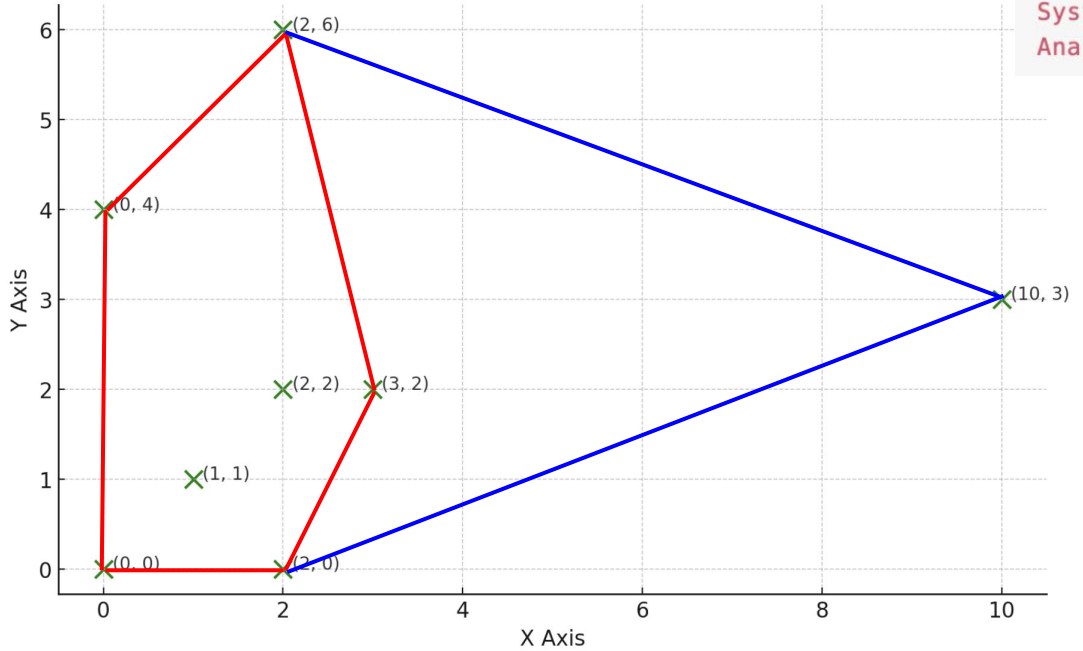
```
public static void main(String[] args) throws Exception {

    ArrayList<Point2D> stationCoordinates = new ArrayList<>();
    stationCoordinates.add(new Point2D(0, 0));
    stationCoordinates.add(new Point2D(2, 0));
    stationCoordinates.add(new Point2D(3, 2));
    stationCoordinates.add(new Point2D(2, 6));
    stationCoordinates.add(new Point2D(0, 4));
    stationCoordinates.add(new Point2D(1, 1));
    stationCoordinates.add(new Point2D(2, 2));

    ObservationStationAnalysis Analysis = new ObservationStationAnalysis(stationCoo
    System.out.println("Farthest Station A: "+Analysis.findFarthestStations()[0]);
    System.out.println("Farthest Station B: "+Analysis.findFarthestStations()[1]);
    System.out.println("Coverage Area: "+Analysis.coverageArea());

    System.out.println("Add Station (10, 3): ");
    Analysis.addNewStation(new Point2D(10, 3));

    System.out.println("Farthest Station A: "+Analysis.findFarthestStations()[0]);
    System.out.println("Farthest Station B: "+Analysis.findFarthestStations()[1]);
    System.out.println("Coverage Area: "+Analysis.coverageArea());
}
```

Example




```
Farthest Station A: (0.0, 0.0)
Farthest Station B: (2.0, 6.0)
Coverage Area: 13.0
```

Example




```
System.out.println("Add Station (10, 3): ");
Analysis.addNewStation(new Point2D(10, 3));
```

```
Farthest Station A: (0.0, 0.0)
Farthest Station B: (10.0, 3.0)
Converage Area: 34.0
```

Template

```
import java.util.ArrayList;
import java.util.List;
import java.util.Arrays;
import edu.princeton.cs.algs4.Point2D;

class ObservationStationAnalysis {

    public ObservationStationAnalysis(ArrayList<Point2D> stations) {
        // you can do something in Constructor or not
    }

    public Point2D[] findFarthestStations() {
        Point2D[] farthest = new Point2D[]{new Point2D(0,0), new Point2D(1,1)}; //
        // find the farthest two stations
        return farthest; // it should be sorted (ascendingly) by polar radius please sort (ascendingly) by y coordinate if there are ties
    }

    public double coverageArea() {
        double area = 0.0;
        // calculate the area surrounded by the existing stations
        return area;
    }

    public void addNewStation(Point2D newStation) {

    }
```

```
public final class Point2D implements Comparable<Point2D> {

    /**
     * Compares two points by x-coordinate.
     */
    public static final Comparator<Point2D> X_ORDER = new XOrder();

    /**
     * Compares two points by y-coordinate.
     */
    public static final Comparator<Point2D> Y_ORDER = new YOrder();

    /**
     * Compares two points by polar radius.
     */
    public static final Comparator<Point2D> R_ORDER = new ROrder();

    private final double x;    // x coordinate
    private final double y;    // y coordinate
```