ECON-381 SEMESTER PROJECT

1. What kind of coordinate system can we use to denote the cells in a pointy top hex grid? If there are alternatives, which one provides the easiest method to compute distances, or perform intersections on ranges as depicted above?

   ANSWER:

   - **Axial Coordinates (q,r)**
   - **Cube Coordinates (x,y,z)**
   - **Offset Coordinates (col,row)**

2. Which data structure is better suited to store the entire map?
   - Perfect for sparse maps
   - O(1) lookup
   - Handles negative coordinates
   - Memory efficient
   - Simple to implement

3. Which data structure is better suited to store a region defined by the sensor reading? Does it matter if the region is a circle or a ring?
   - Fast membership testing O(1)
   - Easy intersection/union operations
   - Memory efficient
   - Only stores coordinates, no need for cell data
   - Works identically for both circles and rings

4. Implement with Java the coordinate system, the map, and finding the intersection. Your program should get inputs as ◦ Number of cells in map, or an indicator of its dimensions (ie. rows, columns, etc) ◦ Number of cells with radar responses ◦ Coordinates of cells with radar responses (repeats until the number indicated is satisfied) Then your program should output ◦ The number of cells in the intersection, ◦ Their coordinates.

   ANSWER:

```
import java.util.Scanner;
import java.util.HashSet;

public class SimpleHexIntersection {
```

```
static class Coordinate {
  int q;
  int r;

  public Coordinate(int q, int r) {
    this.q = q;
    this.r = r;
  }

  @Override
  public boolean equals(Object o) {
    if (this == o) return true;
    if (!(o instanceof Coordinate)) return false;
    Coordinate that = (Coordinate) o;
    return q == that.q && r == that.r;
  }

  @Override
  public int hashCode() {
    return 31 * q + r;
  }
}

public static void main(String[] args) {
  Scanner scanner = new Scanner(System.in);

  int width = scanner.nextInt();
  int height = scanner.nextInt();
  int numberOfRadars = scanner.nextInt();

  HashSet<Coordinate> intersection = new HashSet<>();

  int cellCount = scanner.nextInt();

  for (int i = 0; i < cellCount; i++) {
    int q = scanner.nextInt();
    int r = scanner.nextInt();
    intersection.add(new Coordinate(q, r));
  }

  for (int radar = 1; radar < numberOfRadars; radar++) {
      HashSet<Coordinate> currentRadar = new HashSet<>();

      cellCount = scanner.nextInt();

      for (int i = 0; i < cellCount; i++) {
        int q = scanner.nextInt();
        int r = scanner.nextInt();
        currentRadar.add(new Coordinate(q, r));
      }

      intersection.retainAll(currentRadar);
    }

    System.out.println(intersection.size());
    for (Coordinate coord : intersection) {
      System.out.println(coord.q + " " + coord.r);
    }
  }
}
```

Sample input:

```
5 5
2
3
0 0
1 0
1 1
2
0 0
1 0
```

Output:

```
2
0 0
1 0
```

5. Your report should also include a single test case ◦ A sketch (hand made and photographed is acceptable) which marks the towers and the regions, and the intersection. ◦ Screen shot of your program working, taking the inputs ◦ Screen shot of your program working, showing the output.

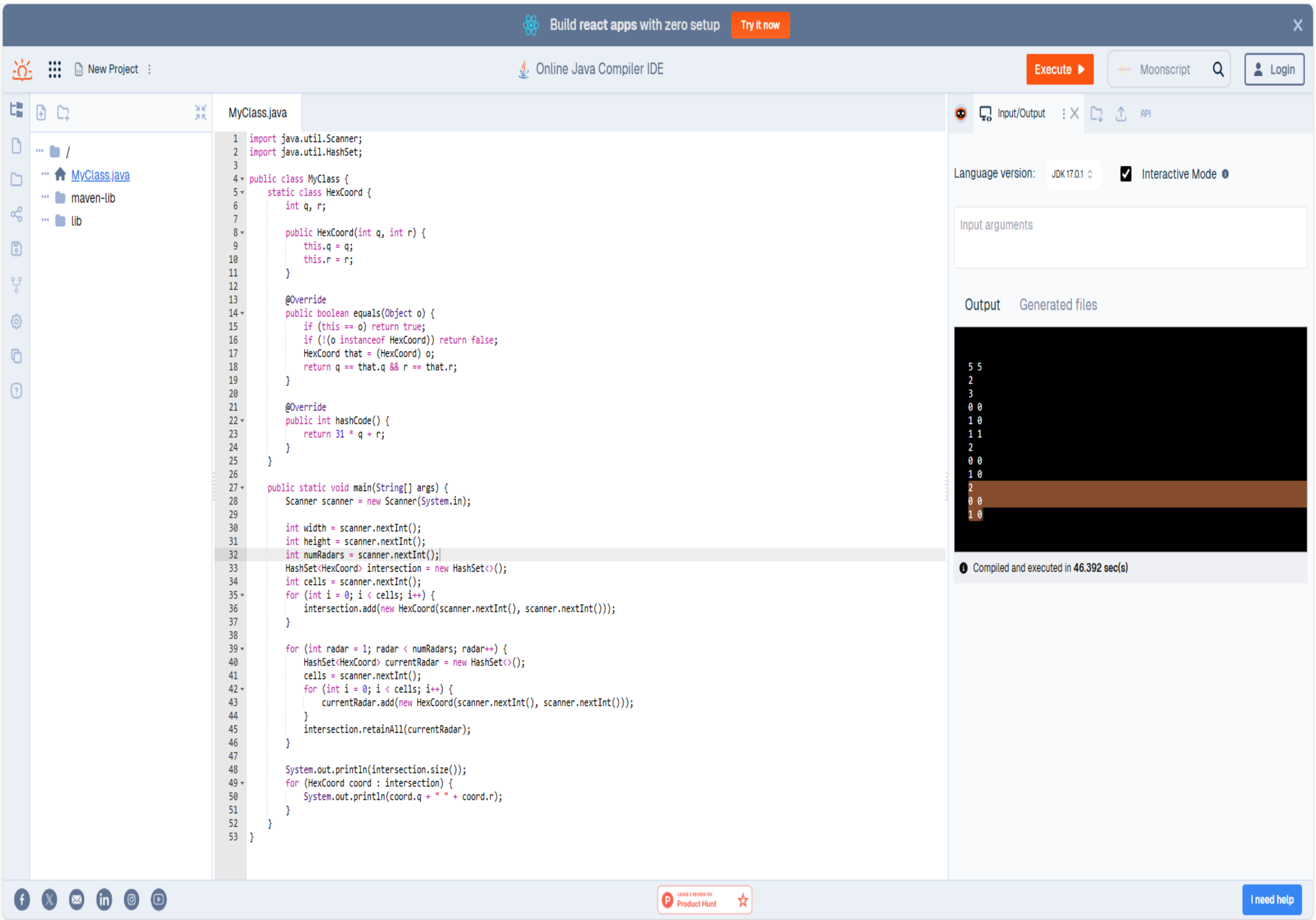
ANSWER:

Sample input:

5 5 // width height
2 // number of radars
3 // first radar cells
0 0
1 0
1 1
2 // second radar cells
0 0
1 0

Sample output:

2 // size of intersection
0 0 // coordinates
1 0



```java
import java.util.Scanner;
import java.util.HashSet;

public class MyClass {
    static class HexCoord {
        int q, r;

        public HexCoord(int q, int r) {
            this.q = q;
            this.r = r;
        }

        @Override
        public boolean equals(Object o) {
            if (this == o) return true;
            if (!(o instanceof HexCoord)) return false;
            HexCoord that = (HexCoord) o;
            return q == that.q && r == that.r;
        }

        @Override
        public int hashCode() {
            return 31 * q + r;
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int width = scanner.nextInt();
        int height = scanner.nextInt();
        int numRadars = scanner.nextInt();
        HashSet<HexCoord> intersection = new HashSet<>();
        int cells = scanner.nextInt();
        for (int i = 0; i < cells; i++) {
            intersection.add(new HexCoord(scanner.nextInt(), scanner.nextInt()));
        }

        for (int radar = 1; radar < numRadars; radar++) {
            HashSet<HexCoord> currentRadar = new HashSet<>();
            cells = scanner.nextInt();
            for (int i = 0; i < cells; i++) {
                currentRadar.add(new HexCoord(scanner.nextInt(), scanner.nextInt()));
            }
            intersection.retainAll(currentRadar);
        }

        System.out.println(intersection.size());
        for (HexCoord coord : intersection) {
            System.out.println(coord.q + " " + coord.r);
        }
    }
}
```

Language version: JDK 17.0.1

☑ Interactive Mode

Input arguments

```
5 5
2
3
0 0
1 0
1 1
2
0 0
1 0
2
0 0
1 0
```

Compiled and executed in 46.392 sec(s)

Elif Işıl Çiçek 2323281007