Student: Michael Grossman

Due Date: 5/19/2022

Algorithm steps for Main given an input file and an output file:

1. inFile, outFile ← args[]
2. numRows, numCol, minVal, maxVal ←inFile
3. diagonal ←sqrt(numRows\*numRows + numCols\*numCols)
4. offset ← diagonal
5. imgAry ←dynamically allocated size of numRows by numCols
6. houghAry ← dynamically allocated size of 2\*diagonal by 180
7. loadImage()
8. For-Each row and column in imgAry:
9. If imgAry[row][col] > 0:
10. buildHoughSpace(row, col)
11. End-if
12. End-For-Each
13. prettyPrint(houghAry, outFile)
14. close all files

Algorithm steps for buildHoughSpace given a row and col into imgAry:

1. angleInD ← 0
2. angleInR ← (angleInD / 180) \*PI
3. dist ←( row\*cos(angleInR) + col\*sin(angleInR))
4. houghAry[dist][angleInD]++
5. angleInD++
6. repeat 2 to 5 while angleInD < 180

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.util.StringTokenizer;

public class Q2\_Main{

    public static int numRow, numCol, minVal, maxVal;

    public static int[][] imgAry;

    public static int[][] houghAry;

    public static int angleInD;

    public static double angleInR;

    public static int offset;

    public static int diagonal;

    public static int dist;

    public static void prettyPrint(int[][] ary, BufferedWriter outp){

        int houghR = 2\*diagonal;

        int houghC = 180;

        try{

            for(int row = 0; row < houghR; row++){

                for(int col = 0; col < houghC; ++col){

                    if(ary[row][col] > 0){

                        outp.write(Integer.toString(ary[row][col]));

                        //banking that no hough space with > 2 digits

                        if(ary[row][col] < 10) outp.write("  ");

                        else outp.write(" ");

                    }

                    else outp.write(".  ");

                }

                outp.write("\n");

            }

        }catch(Exception e){

            System.out.println(e.getMessage());

        }

    }

    public static void loadImage(int[][] ary, BufferedReader input){

        StringTokenizer st;

        try{

            for(int i = 0; i < numRow; ++i){

                st = new StringTokenizer(input.readLine());

                for(int j = 0; j < numRow; ++j){

                    ary[i][j] = Integer.parseInt(st.nextToken());

                }

            }

        }catch(Exception e){

            System.out.println(e.getMessage());

        }

    }

    public static void buildHoughSpace(int r, int c){

        angleInD = 0;

        while(angleInD < 180){

            angleInR = (Math.PI \* angleInD)/180;

            dist = (int)(r \* Math.cos(angleInR) + c\*Math.sin(angleInR) + offset);

            houghAry[dist][angleInD]++;

            angleInD++;

        }

    }

    public static void main(String[] args){

        try{

            String input = args[0];

            String output = args[1];

            BufferedReader br = new BufferedReader(new FileReader(input));

            BufferedWriter bw = new BufferedWriter(new FileWriter(output));

            StringTokenizer st = new StringTokenizer(br.readLine());

            numRow = Integer.parseInt(st.nextToken());

            numCol = Integer.parseInt(st.nextToken());

            minVal = Integer.parseInt(st.nextToken());

            maxVal = Integer.parseInt(st.nextToken());

            diagonal = (int)Math.sqrt(numRow\*numRow + numCol\*numCol);

            offset = diagonal;

            imgAry = new int[numRow][numCol];

            houghAry = new int[2\*diagonal][180];

            int hRow = 2\*diagonal;

            for(int i = 0; i < hRow; i++){

                for(int j = 0; j < 180; ++j){

                    houghAry[i][j] = 0;

                }

            }

            loadImage(imgAry, br);

            for(int row = 0; row < numRow; ++row){

                for(int col = 0; col < numCol; ++col){

                    if(imgAry[row][col] > 0){

                        buildHoughSpace(row, col);

                    }

                }

            }

            prettyPrint(houghAry, bw);

            br.close();

            bw.close();

        }catch(Exception e){

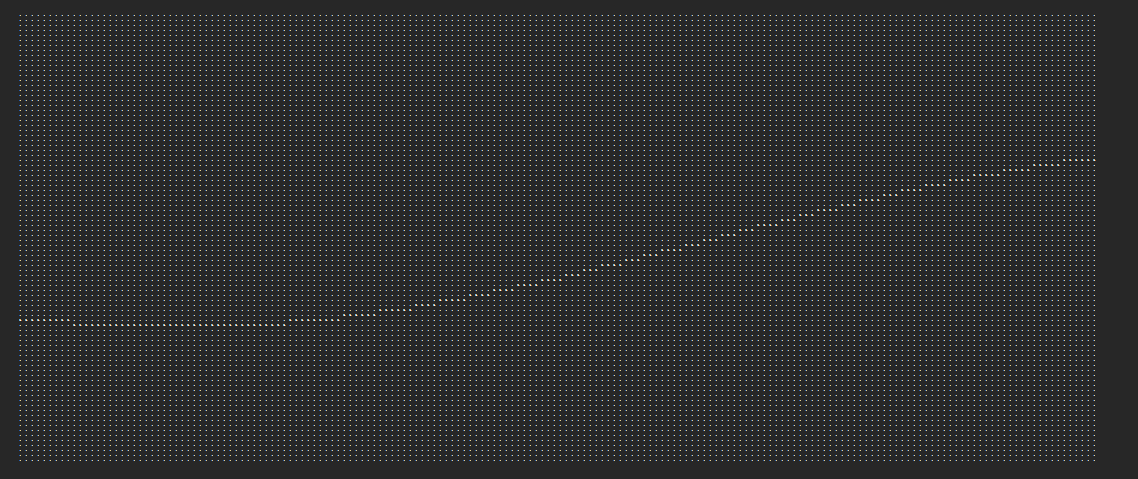
            System.out.println(e.getMessage());

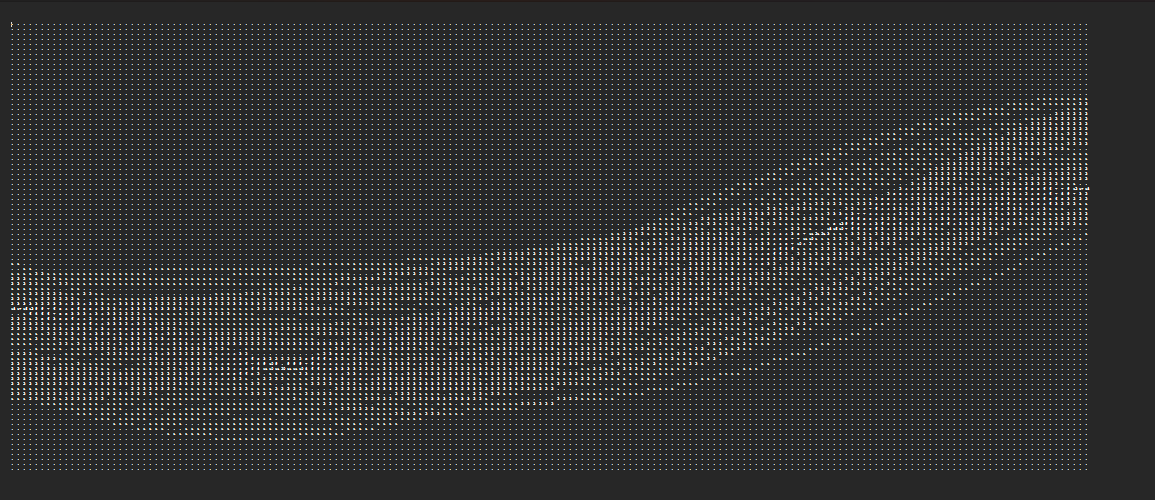
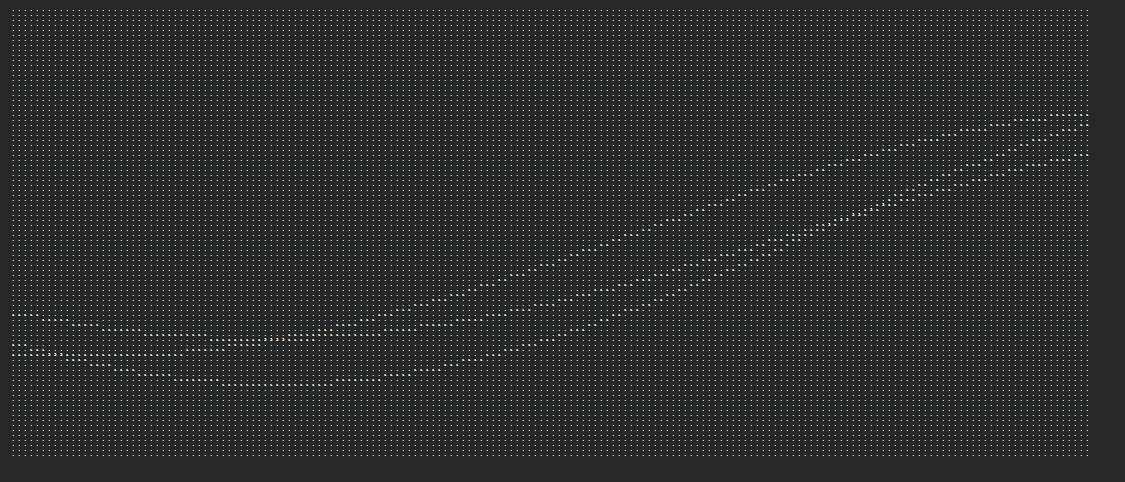
        }

    }

}

**OUTPUTS 1, 2, and then 3**

****

****