CV Project7: Hough Transform Java Student: Zeehan Rahman

Project Due Date:

Algorithmic Steps:

1. Open the input fil, outputf2 and the output1 file
2. Assign all the variables numsrow, numscol, imgmax and imgmin from the input file. Assign the variable HoughAngle to 180.
3. Calculate the diagonal (hypotenuse) of the image using the image rows and cols (Math.sqrt(rows^2 + col^2))
4. Dynamically allocate the image array of size rows by columns
5. Dynamically allocate the Hough Array of size diagonal(from step 3) * 2 by houghangle (180) 6. Initialize both the 2D array to 0, in the constructor
7. Call the function loadimage to load the image to the image array
8. Call the function BuildHoughSpace that lays down the hough space of the image inside the hougharray. The function BuildHoughSpace calls the function computesinusoid that calculates the polar distance (dist) of each non zero pixel (via CalcPolarDistance function), and increments the hougharray[dist] [angle] , where angle loops from 0 to 179
9. Call the function prettyprint to output the contents of the hougharray to outfile1 (output "1 for hougharray[i][j] > 0, or output ". " for hougharray[i][j] < 0
10. Call the function determinminmax to find out the max and min of the hougharray 11. Output the value of houghdist, houghangle, houghmin, houghmax, to outputfile2 12. Call the function arraytofile2 output the contents of the hougharray to outputfile2 13. Finally, close all files

Source Code

//Main

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class main {

	public static void main(String[] args) throws IOException
	{
		// TODO Auto-generated method stub
		String inputfilename = args[0];
		FileReader inputImage = null;
		BufferedReader buffinImage = null;
		Scanner input = null;

		String outputname = args[1];
		FileWriter outputwriter = null;
		BufferedWriter output = null;

		String outputname2 = args[2];
		FileWriter outputwriter2 = null;
		BufferedWriter output2 = null;

		try
		{
			inputImage = new FileReader(inputfilename);
			buffinImage = new BufferedReader(inputImage);
```

```
                                        input = new Scanner(buffinImage);

                                        outputwriter = new FileWriter(outputname);
                                        output = new BufferedWriter(outputwriter);

                                        outputwriter2 = new FileWriter(outputname2);
                                        output2 = new BufferedWriter(outputwriter2);

                                        int numrows = 0, numcols = 0, minval = 0, maxval = 0;

                                        if(input.hasNextInt()) numrows = input.nextInt();
                                        else System.out.println("Invalid format of header");

                                        if(input.hasNextInt()) numcols = input.nextInt();
                                        else System.out.println("Invalid format of header");

                                        if(input.hasNextInt()) minval = input.nextInt();
                                        else System.out.println("Invalid format of header");

                                        if(input.hasNextInt()) maxval = input.nextInt();
                                        else System.out.println("Invalid format of header");

                                        HTrans obj = new HTrans(numrows, numcols, minval, maxval);

                                        obj.loadimage(input);

                                        obj.BHSpace();



                                        obj.prettyprint(output, "PrettyPrint");

                                        obj.DetermineMinMax();

                                        output2.write(obj.HDist +" " + obj.HAngle + " " + obj.HMin + " " + obj.HMax);

                                        obj.Array2File(output2, "");


                                }

                                finally
                                {
                                if(input!=null) input.close();
                                if(output!=null) output.close();
                                if(output2!=null) output2.close();
                                }
                        }

}
```

//HTrans

```
import java.io.BufferedWriter;
import java.io.IOException;
import java.util.Scanner;

public class HTrans
{
	public int numrows = 0, numcols = 0, minval = 0, maxval = 0, HAngle = 180, HDist = 0,
				HMin = 9999, HMax = 0;

	public int imagearray[][], HArray[][];

	public HTrans(int nr, int nc, int minv, int maxv)
	{

		this.numrows = nr;
		this.numcols = nc;
		this.minval = minv;
		this.maxval = maxv;

		HDist = (int) Math.sqrt(nr * nr + nc * nc);

		this.imagearray = new int[numrows] [numcols];
		this.HArray = new int[2*HDist] [HAngle];

		for(int i = 0; i < this.numrows; i++)
		{
			for(int j = 0; j < this.numcols; j++)
			{
				this.imagearray [i][j] = 0;
				this.HArray [i][j] = 0;
			}
		}
	}

	public void loadimage(Scanner m)
	{


		for(int i = 0; i < this.numrows; i++)
		{
			for(int j = 0; j < this.numcols; j++)
			{
				if(m.hasNextInt())
					this.imagearray[i][j] = m.nextInt();

			}


		}
	}

	public void BHSpace()
	{
		for(int i = 0; i < this.numrows; i++)
		{
			for(int j = 0; j < this.numcols; j++)
			{
				if(this.imagearray[i][j]>0)
					ComputeSinusoid(i,j);


			}


		}
	}



	public void ComputeSinusoid(int i, int j)
	{
		int DAngle = 0;

		while(DAngle < 180)
		{
		double RAngle = (double)DAngle/180.0 * Math.PI;

		int dist = (int)CalcPolarDist(i, j, RAngle);
```

```
                this.HArray[dist][DAngle] = this.HArray[dist][DAngle] + 1;
                DAngle++;

                }

}

public double CalcPolarDist(int i, int j, double Radians)
{
                double dist = (double)i * Math.cos(Radians) + (double)j * Math.sin(Radians) + this.HDist;
                return dist;
}

public void DetermineMinMax()
{
                for(int i = 0; i < 2*this.HDist; i++)
                {
                                for(int j = 0; j < this.HAngle; j++)
                                {
                                                if(this.HArray[i][j] < HMin)
                                                                HMin = this.HArray[i][j];

                                                if(this.HArray[i][j] > HMax)
                                                                HMax = this.HArray[i][j];

                                }


                }
}

public void Array2File(BufferedWriter outputimage, String caption)
{
                try
                {
                                outputimage.write(caption+"\n");

                                for(int i = 0; i < 2*this.HDist; i++)
                                {
                                                for(int j = 0; j < this.HAngle; j++)
                                                {
                                                                outputimage.write(this.HArray[i][j]+" ");

                                                }
                                                               outputimage.write("\n");
```

```
                                        }
                                }

                                catch(IOException e)
                                {
                                        e.printStackTrace();
                                }
                }


                public void prettyprint(BufferedWriter outputimage, String caption)
                {
                                try
                                {
                                        outputimage.write(caption+"\n");

                                        for(int i = 0; i < 2*this.HDist; i++)
                                        {
                                                for(int j = 0; j < this.HAngle; j++)
                                                {

                                                        if(this.HArray[i][j] > 0)
                                                                outputimage.write(this.HArray[i][j] + " ");

                                                        else
                                                                outputimage.write(". ");
                                                }
                                                        outputimage.write("\n");
                                        }
                                }

                                catch(IOException e)
                                {
                                        e.printStackTrace();
                                }
                }

                public void showArray()
                {
                                for(int i = 0; i < 2*this.HDist; i++)
                                {
                                        for(int j = 0; j < this.HAngle; j++)
                                        {
                                                System.out.print(this.HArray[i][j]+" ");

                                        }
                                        System.out.println();
                                }
                }

}
```

Output

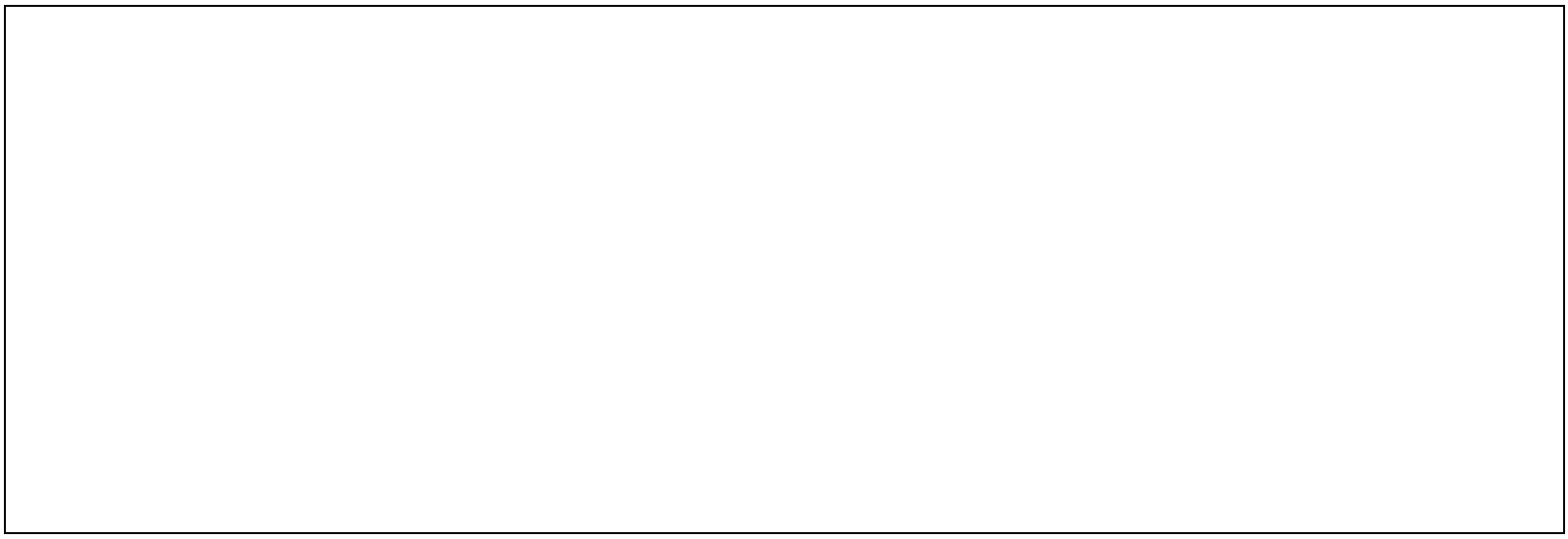
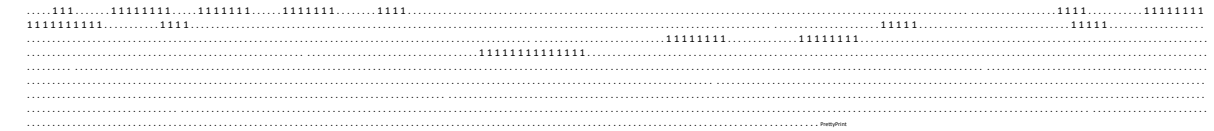


Image5:

Outfile1:

PrettyPrint



Outfile2:

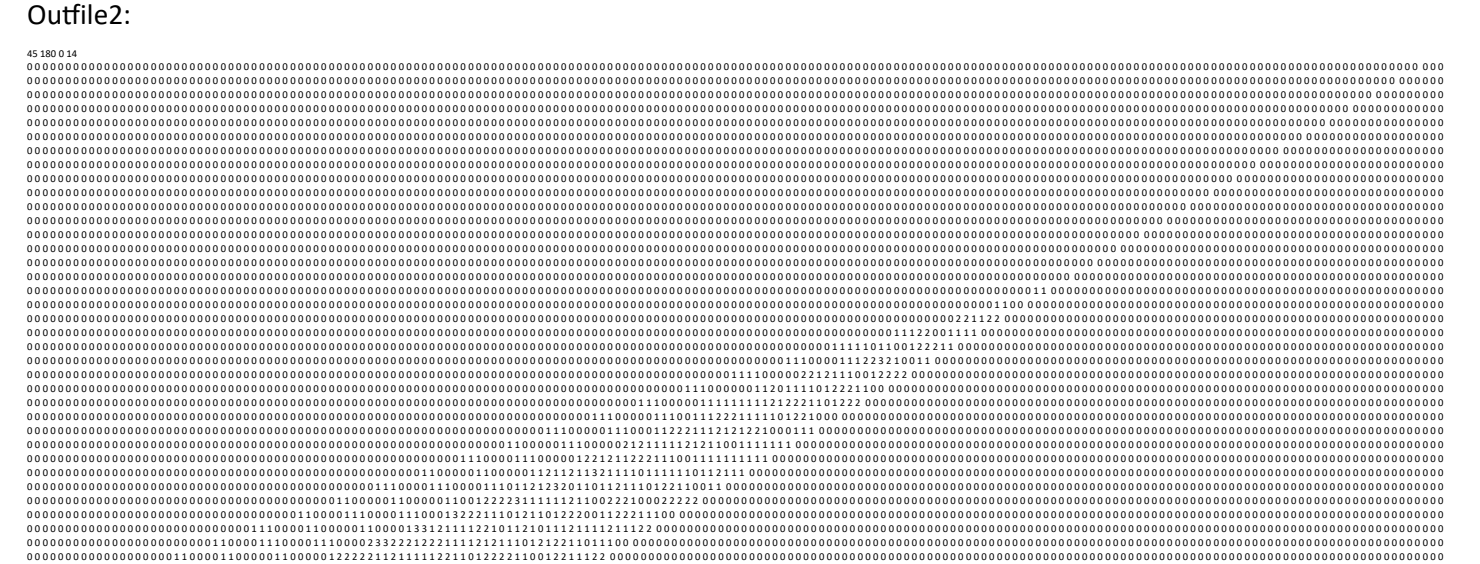
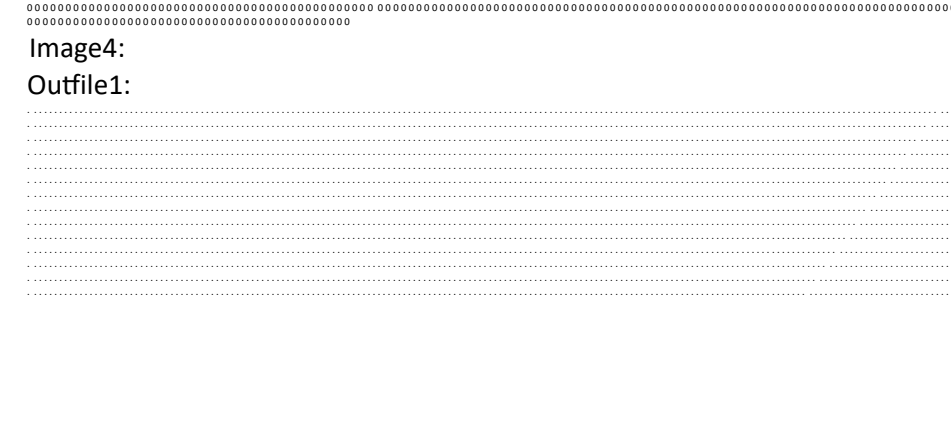


Image4:

Outfile1:



Outfile2:

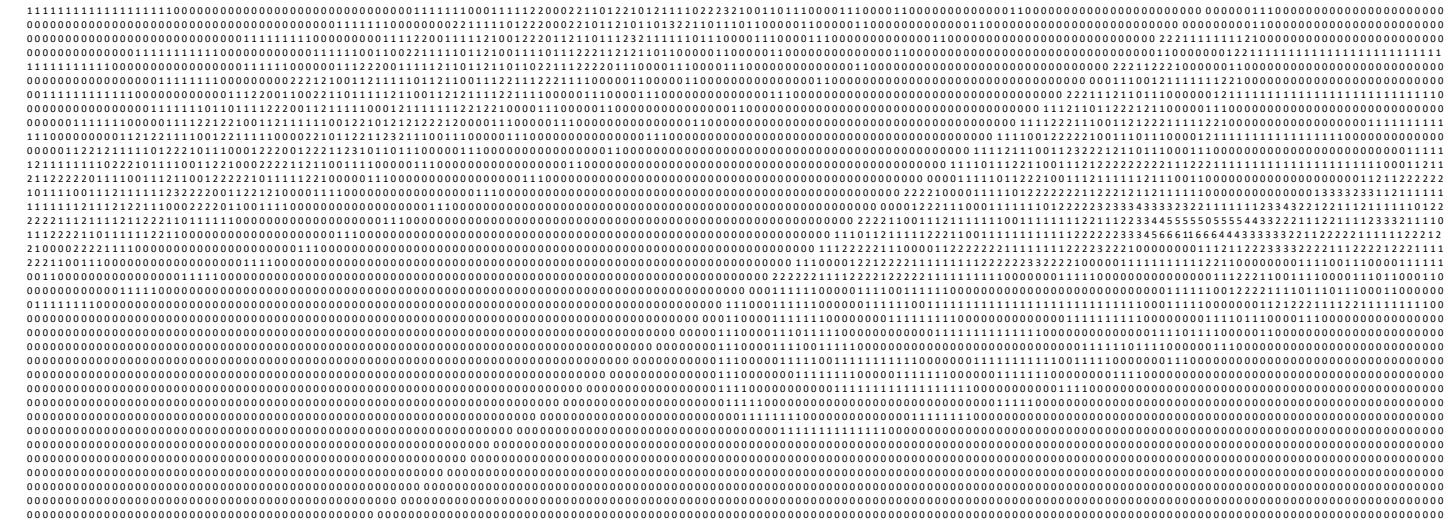
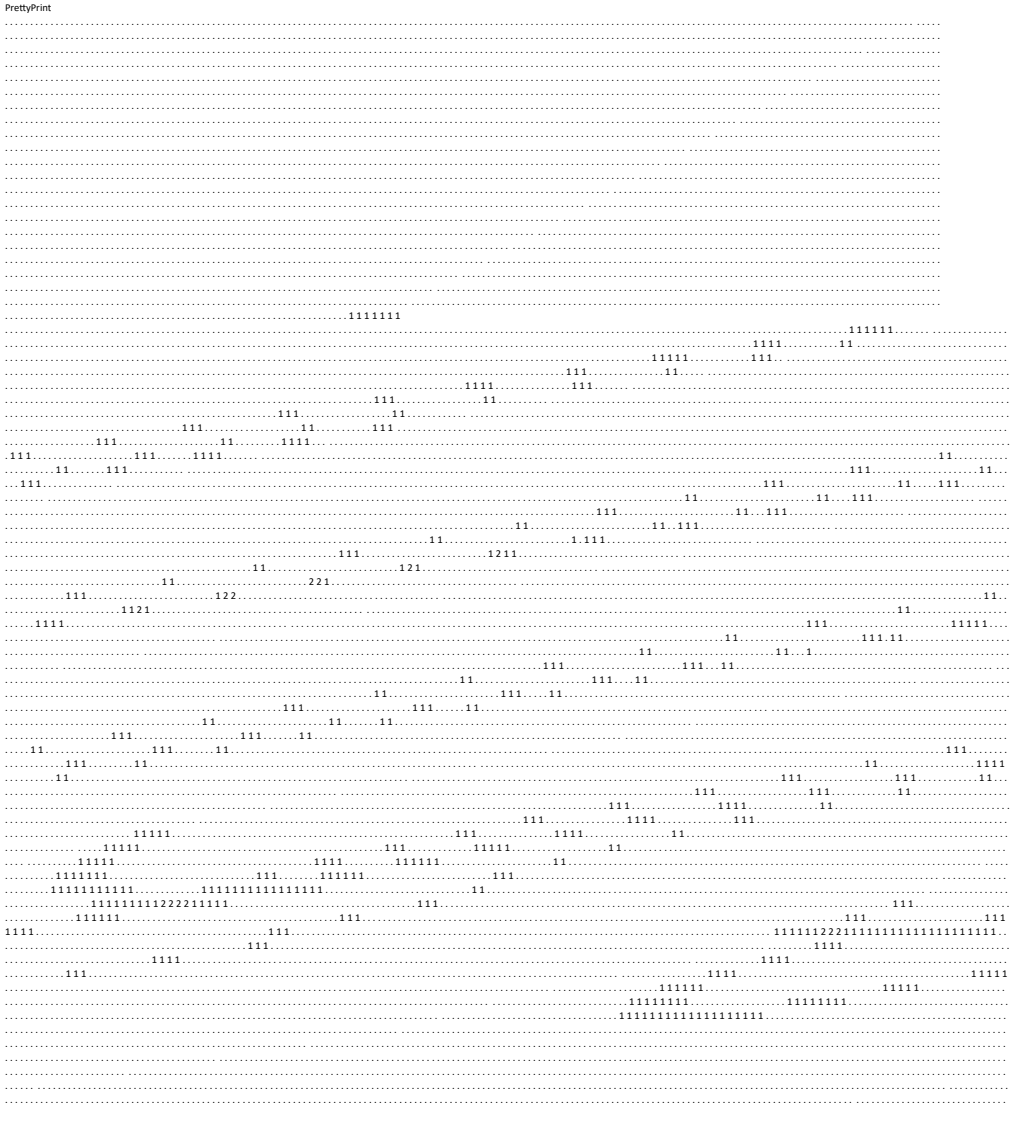
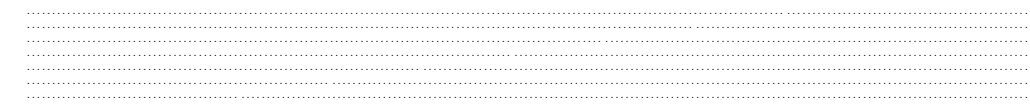


Image3:

Outfile1:

PrettyPrint





Outfile2:

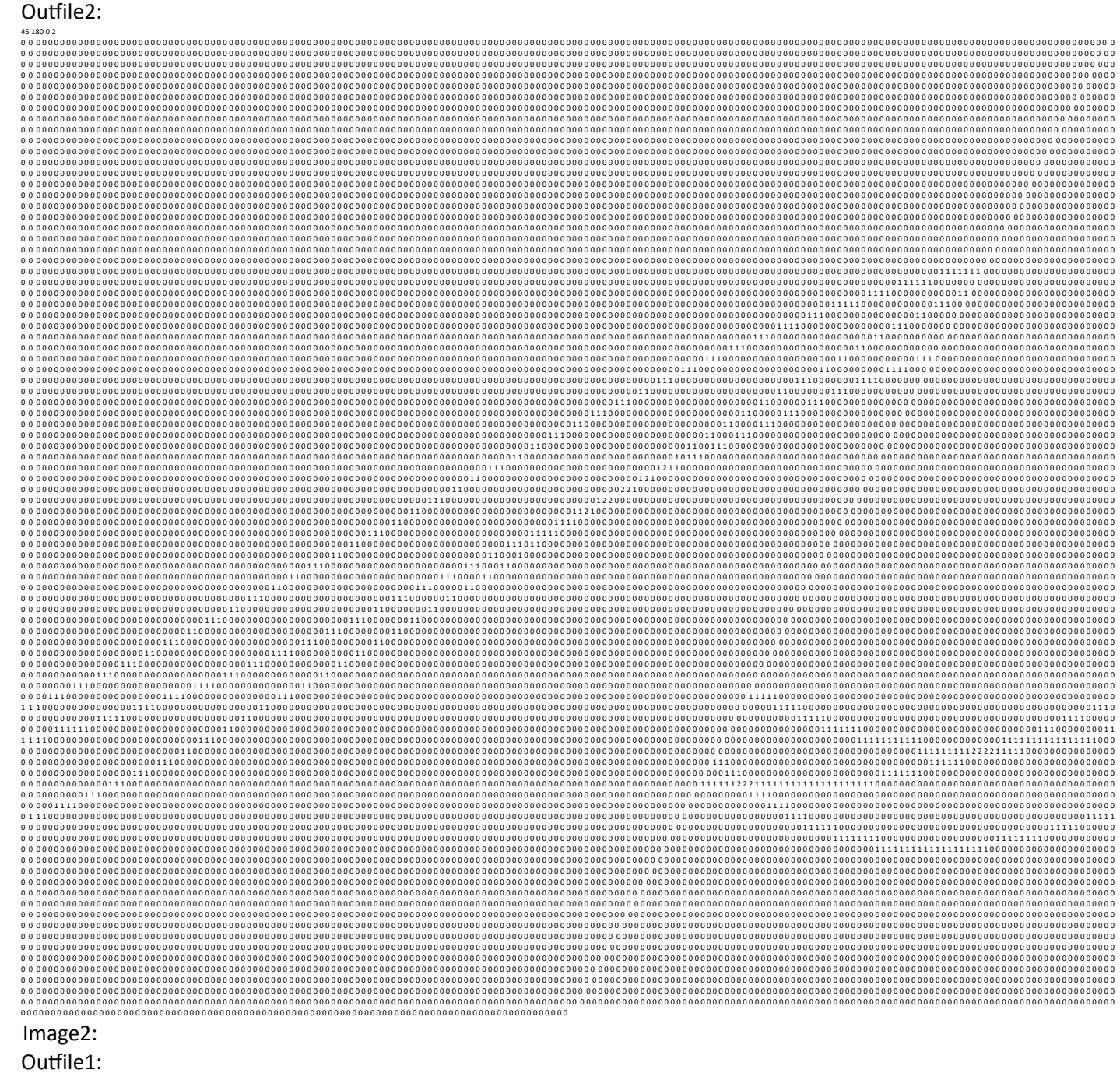
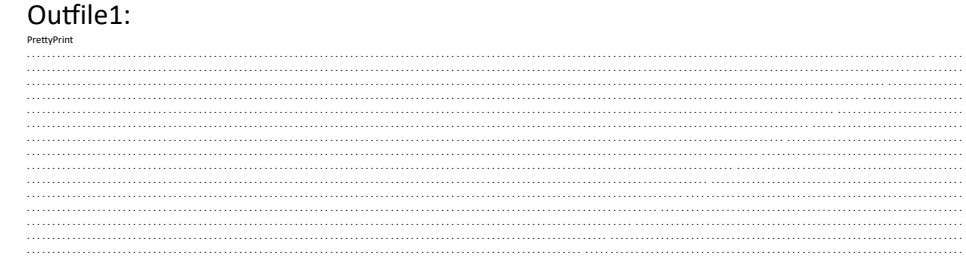
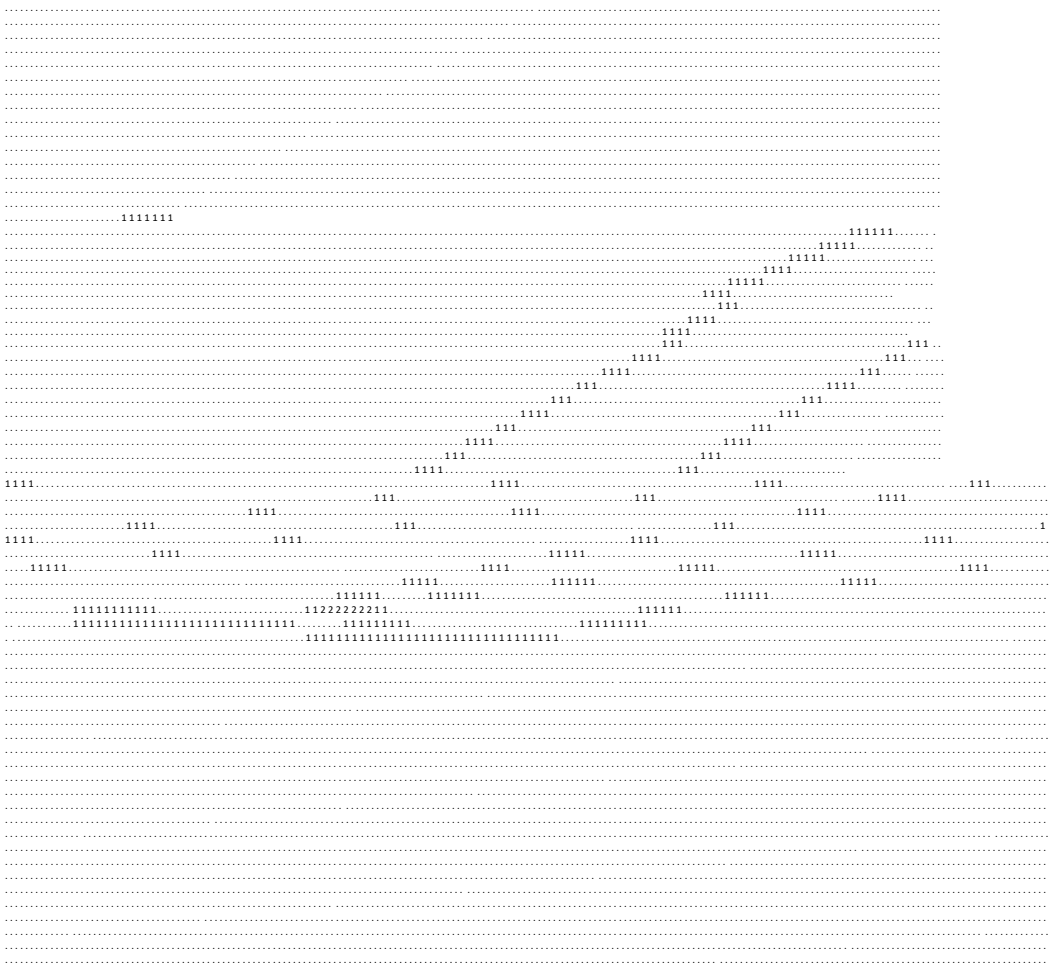


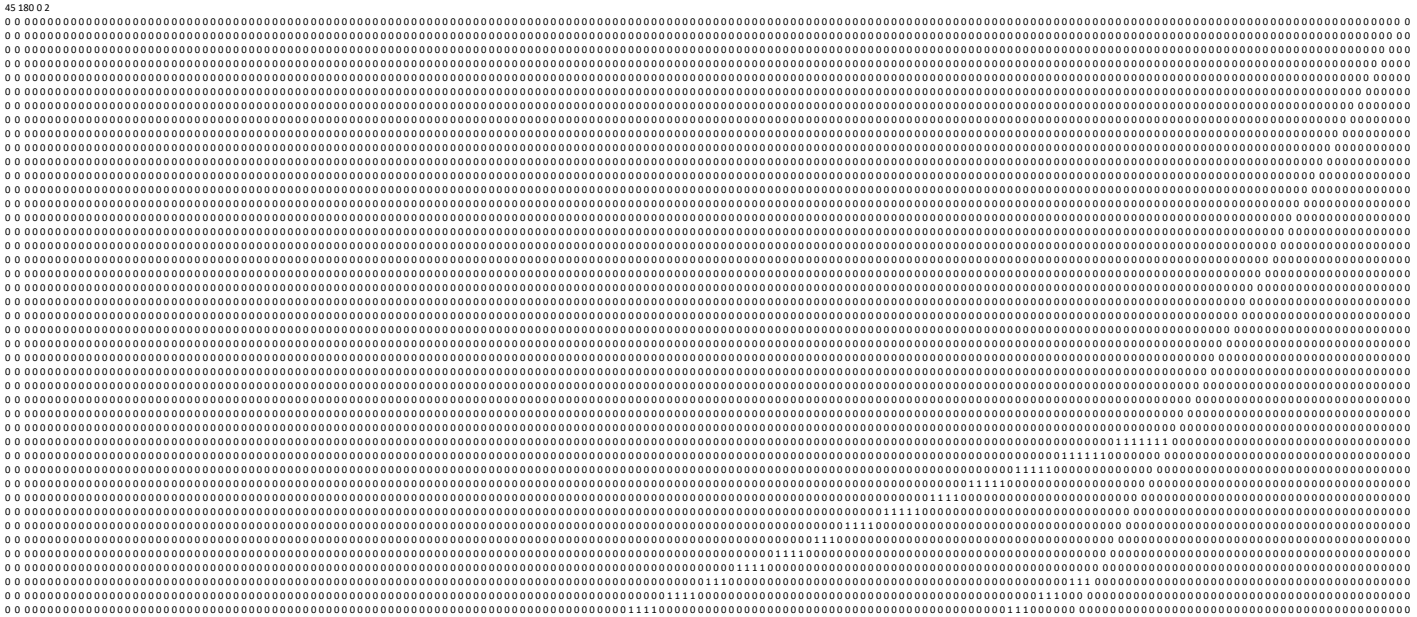
Image2:

Outfile1:





Outfile2:



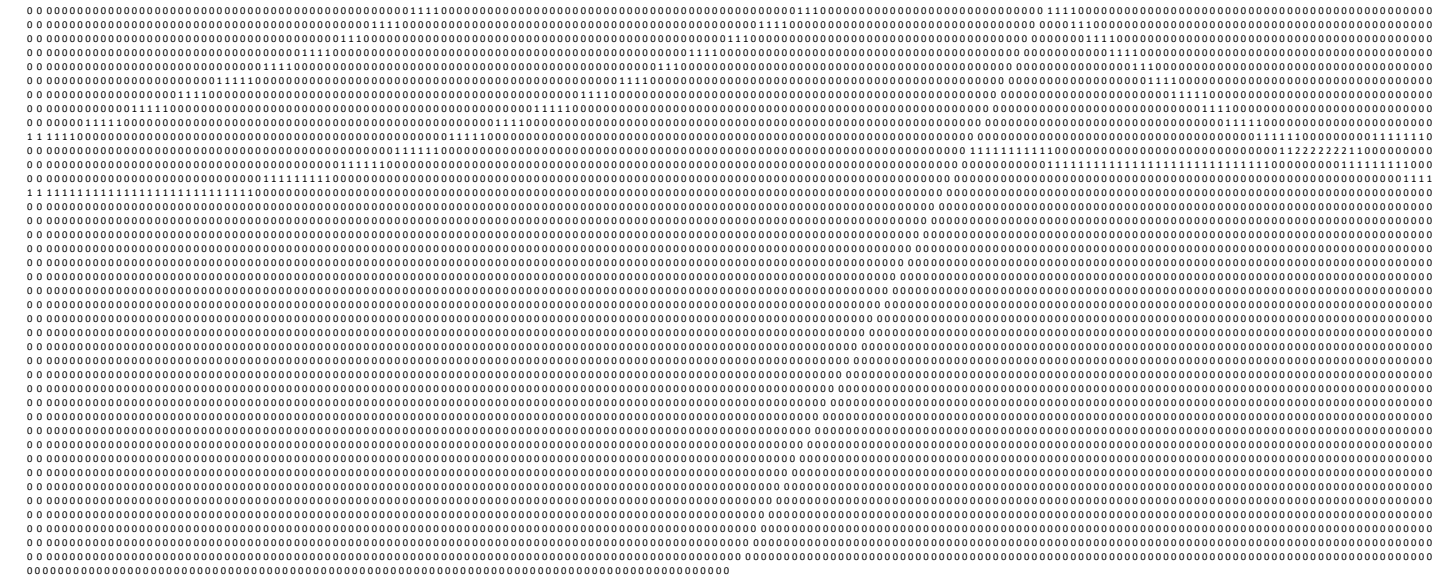
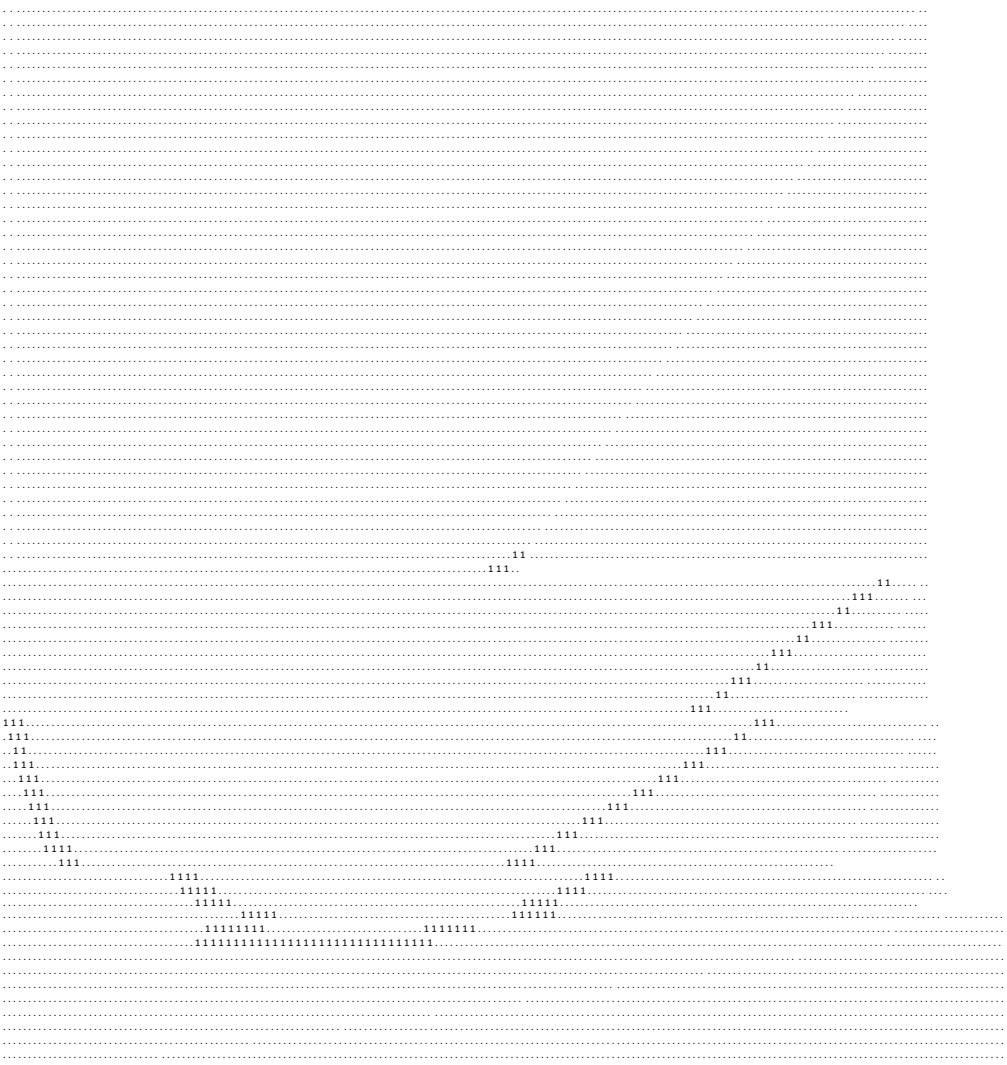


Image1:

Outfile1:

PrettyPrint



Outfile2: