CV **Kcurvature** C++

Student: Shuhua Song

Project Due Date: 05/11/2021

(1) Algorithm steps for initialization(inFile):
Step 1: numPts <- countPts(inFile)
Step 2: close inFile
Step 3: index <- 0
Step 4: (x, y) <- read from inFile
Step 5: storePt(x,y, index) // store x,y to PtAry[index]
Step 6: index++
Step 7: Repeat step 4- step 6 until inFile is empty

(2) Algorithm steps for cornerDetection()
Step1: Q <- 0
P <- K
R <- 2 * K
i = 0

Step2: index <- P
Step3: curvature <- computeCurvature(Q, P, R)
Step4: PtAry[index%numPts] -> curvature <- curvature
Step5: outFile3 <- print Q, P, R, index x, y, curvature of PtAry[index]
Step6: Q = (Q+1)%numPts
P = (P+1)%numPts
R = (R+1)%numPts
index++
i++
Step7: repeat step3 to step 6 untile i<numPts

Source Code:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

class kCurvature;
class Image{
    friend class kCurvature;
public:
    int numRows, numCols;
    int minVal, maxVal;
    int label; //
    int** imgAry;
    Image(){ }
    Image(int numRows, int numCols, int minVal, int maxVal, int label){
        this->numRows = numRows;
```

```
        this->numCols = numCols;
        this->minVal = minVal;
        this->maxVal = maxVal;
        this->label = label;
        imgAry = new int*[numRows];
        for(int i=0; i<numRows; i++){
            imgAry[i] = new int[numCols];
            for(int j=0; j<numCols; j++){
                imgAry[i][j] = 0;
            }
        }
    }

    //void plotPt2Img(int** imgAry){ }

    void reformatPrettyPrint(int** inArray, ofstream& outFile){
        for(int i=0; i<numRows; i++){
            for(int j=0; j<numCols; j++){
                if(inArray[i][j] != 0){
                    outFile << inArray[i][j] << " ";
                }else{
                    outFile << "." << " ";
                }
            }
            outFile << endl;
        }
    }

    int countPts(ifstream& inFile){
        int numPts = 0;
        while(!inFile.eof()){
            int x = -1, y = -1;
            inFile >> x >> y;
            if(x!=-1 && y!=-1){
                numPts++;
            }
        }
        return numPts;
    }

    void print(int** imgAry, ofstream& outFile){
        for(int i=0; i<numRows; i++){
            for(int j=0; j<numCols; j++){
                outFile << imgAry[i][j] << " ";
            }
            outFile << endl;
        }
    }
};

class BoundaryPt{
    friend class kCurvature;
public:
    int x, y;
    double curvature;
    int localMax;
    int corner; // 1 = non-corner,  or 9 = corner
    BoundaryPt(int x, int y, double curvature, int localMax, int corner){
        this->x = x;
        this->y = y;
        this->curvature = curvature;
```

```
        this->localMax = localMax;
        this->corner = corner;
        //cout << "Point created" << endl;
    }
};

class kCurvature{
public:
    int K, numPts;
    int Q, P, R;
    BoundaryPt** PtAry;
    int** boundAry;
    kCurvature(int K, int numPts){
        this->K = K;
        this->numPts = numPts;
        Q = 0; P = K; R = 2 * K;
        PtAry = new BoundaryPt*[numPts];
        boundAry = new int*[numPts];
        for(int i=0; i<numPts; i++){
            boundAry[i] = new int[2];
        }

    }
    kCurvature(int K, int numPts, int Q, int P, int R){
        this->K = K;
        this->numPts = numPts;
        this->Q = Q;
        this->P = P;
        this->R;
    }
    void initialization(ifstream& inFile, ofstream& outFile){
        int index = 0;
        while(!inFile.eof() && index < numPts){
            int  x = -1, y = -1;
            inFile >> x >> y;
            if(x!=-1 && y!=-1){
                boundAry[index][0] = x;
                boundAry[index][1] = y;
            }
            index++;
        }
        storePt(outFile);
    }
    void storePt(ofstream& outFile){
        outFile << "Print Boundary Points Array: " << endl;
        for(int i=0; i<numPts; i++){
            int curX = boundAry[i][0];
            int curY = boundAry[i][1];
            //cout << "get: " << curX << " " << curY << endl;
            PtAry[i] = new BoundaryPt(curX, curY, 0.0, -1,-1);
            outFile <<  PtAry[i]->x << " " << PtAry[i]->y << endl;
            //cout << "Array : " <<  PtAry[i]->x << " " << PtAry[i]->y << endl;
        }
        /*
        for(int i=0; i<numPts; i++) {
            cout << "Array : " << PtAry[i]->x << " " << PtAry[i]->y << endl;
        }*/
    }

    void cornerDetection(BoundaryPt** PtAry, ofstream& outFile){
        int Q = 0, P = K, R = 2 * K;
```

```
        //cout << "Origin: " << Q << " " << P << " " << R << endl;
        int index = P;
        double curva = 0.0;
        int i = 0;
        outFile << "\nrow " << "col " << "curvature" << endl;
        while(i<numPts){
            curva = computeCurvature(PtAry, Q, P, R);
            PtAry[index%numPts]->curvature = curva;
            //cout << " index = " << index << ", i =  " << i  << " Origin: " << Q << "
" << P << " " << R << endl;
            //cout << "PtAry[" << (index%numPts) << "].curvature = " <<
PtAry[index%numPts]->curvature <<  endl;
           // cout << " x = " << PtAry[index%numPts]->x <<  " y = " <<
PtAry[index%numPts]->y << endl;
            outFile << PtAry[index%numPts]->x <<  " " << PtAry[index%numPts]->y << " "
<< PtAry[index%numPts]->curvature << endl;
            Q = (Q+1) % numPts;
            P = (P+1) % numPts;
            R = (R+1) % numPts;
            index++;
            i++;
        }
    }
    //double computeCurvature(int Q, int P, int R);
    double computeCurvature(BoundaryPt** PtAry, int Q, int P, int R){
        double result = 0.0, slop1 = 0.0, slop2 = 0.0;
        double r1 =-1, c1 = -1, r2 = -1, c2 = -1, r3 = -1, c3 = -1;
        if(P < numPts && Q < numPts && R < numPts){
            //Q
            r1 = PtAry[Q]->x;
            c1 = PtAry[Q]->y;
            //P
            r2 = PtAry[P]->x;
            c2 = PtAry[P]->y;
            //R
            r3 = PtAry[R]->x;
            c3 = PtAry[R]->y;
            // cout << "\nData: " << r1 <<  " " << c1 << " " << r2 << " " << c2 << " "
<< r3 << " " << c3 << endl;
            if(r1-r2==0){
                slop1 = (c1-c2)/0.5;
            }else{
                slop1 = (c1-c2)/(r1-r2);
            }
            if(r2-r3==0){
                slop2 = (c2-c3)/0.5;
            }else{
                slop2 = (c2-c3)/(r2-r3);
            }
            if(slop1 > slop2){
                result = slop1 - slop2;
            }else{
                result = slop2 - slop1;
            }
            //cout << "result[" << P << "] = " << result << endl;
        }
        return result;
    }

    int computeLocalMaxima(BoundaryPt** PtAry, int i){
            int neib1 = (i-2) < 0 ? (numPts-2) : (i-2);
```

```
            int neib2 = (i-1) < 0 ? (numPts-1) : (i-1);
            int neib3 = (i+1) >= numPts ? (i+1)%numPts : (i+1);
            int neib4 = (i+2) >= numPts ? (i+2)%numPts : (i+2);
            if(PtAry[i]->curvature > PtAry[neib1]->curvature &&
               PtAry[i]->curvature > PtAry[neib2]->curvature &&
               PtAry[i]->curvature > PtAry[neib3]->curvature &&
               PtAry[i]->curvature > PtAry[neib4]->curvature){
                return 1;
            }
            return 0;
    }

    void markCorner(BoundaryPt** PtAry){
        int isLocalMax = 0, preMax = 0, nextMax = 0;
        int pre2Max = 0, next2Max = 0;
        int index = 0;
        for(int i=0; i<numPts; i++){
            index = i;
            isLocalMax = computeLocalMaxima(PtAry, i);
            index = (i-1) < 0 ? (numPts-1) : (i-1);
            if(i-1 < 0){
                preMax = computeLocalMaxima(PtAry, numPts-1);
            }else{
                preMax = computeLocalMaxima(PtAry, i-1);
            }
            if(i+1 >= numPts){
                nextMax = computeLocalMaxima(PtAry, (i+1)%numPts);
            }else{
                nextMax = computeLocalMaxima(PtAry, i+1);
            }

            if(i-2 < 0){
                pre2Max = computeLocalMaxima(PtAry, numPts-2);
            }else{
                pre2Max = computeLocalMaxima(PtAry, i-2);
            }
            if(i+2 >= numPts){
                next2Max = computeLocalMaxima(PtAry, (i+2)%numPts);
            }else{
                next2Max = computeLocalMaxima(PtAry, i+2);
            }

            if(isLocalMax==1){
                if( preMax==1 || nextMax==1 || pre2Max==0 && next2Max==0){
                    PtAry[i]->corner = 9;
                }
            }else{
                PtAry[i]->corner = 1;
            }
        }
    }

    void printBoundary(BoundaryPt** PtAry, ofstream& outFile){
         for(int i=0; i<numPts; i++){
             outFile << PtAry[i]->x << " " << PtAry[i]->y << " " << PtAry[i]->corner
<< endl;
         }
    }

    void plotPt2Img(BoundaryPt** PtAry, int** imgAry){
        for(int i=0; i<numPts; i++){
```

```
            int x = PtAry[i]->x;
            int y = PtAry[i]->y;
            int corner = PtAry[i]->corner;
            imgAry[x][y] = corner;
        }
    }

};

int main(int argc, const char *argv[]){
    ifstream inFile(argv[1]);
    ofstream outFile1(argv[2]); //The result of the K-curvature of the object boundary
points.
    ofstream outFile2(argv[3]); //Pretty print (displaying)of the result of the K-
curvature corner detection
    ofstream outFile3(argv[4]); //all debugging output
    int K = stoi(argv[5]);
    int numRows = 0, numCols = 0, minVal = 0, maxVal = 0, label = 0;
    inFile >> numRows;
    inFile >> numCols;
    inFile >> minVal;
    inFile >> maxVal;
    inFile >> label;
    cout << K << endl;
    cout << numRows << " " << numCols << " " << minVal << " "<< maxVal << endl;
    cout <<  label << endl;
    cout << "Points: " << endl;
    outFile1 << numRows << " " << numCols << " " << minVal << " "<< maxVal << endl;
    outFile1 << label << endl;
    Image  *image  = new Image(numRows, numCols, minVal, maxVal, label);
    int** imgAry = image->imgAry;
    int numPts = image->countPts(inFile);
    cout << "numPts = " << numPts << endl;
    outFile1 << numPts << endl;
    outFile2 << numRows << " " << numCols << " " << minVal << " "<< maxVal << endl;
    outFile2 << label << endl;
    inFile.close();
    //inFile(argv[1]);
    inFile.open(argv[1]);
    int numRows1 = 0, numCols1 = 0, minVal1 = 0, maxVal1 = 0, label1 = 0;

    inFile >> numRows1;
    inFile >> numCols1;
    inFile >> minVal1;
    inFile >> maxVal1;
    inFile >> label1;
    cout << "label1 = " << label1 << endl;
    //cout << numRows1 << " " << numCols1 << " " << minVal1 << " "<< maxVal1 << " " <<
label1 << endl;
    kCurvature  *kCur = new kCurvature(K, numPts);
    BoundaryPt** PtAry = kCur->PtAry;
    kCur->initialization(inFile, outFile3);
   /* for(int i=0; i<numPts; i++) {
        cout << "PtAry-" << i << " : " << PtAry[i]->x << " " << PtAry[i]->y << endl;
    }*/
    kCur->cornerDetection(PtAry,outFile3);
    kCur->markCorner(PtAry);
    kCur->printBoundary(PtAry, outFile1);
    kCur->plotPt2Img(PtAry, imgAry);
    //image->print(imgAry, outFile2);
    image->reformatPrettyPrint(imgAry, outFile2);
```

```
    inFile.close();
    outFile1.close();
    outFile2.close();
    outFile3.close();
    return 0;
}
```

- outFile1 for dataA

When K = 1

```
30 40 0 1
1
96
6 6 9
7 6 1
8 6 1
9 6 1
10 6 1
11 6 1
12 6 1
13 6 1
14 6 1
15 6 1
16 6 1
17 6 1
18 6 1
19 6 1
20 6 1
21 6 1
22 6 1
23 6 1
24 6 1
25 6 9
25 7 1
25 8 1
25 9 1
25 10 1
25 11 1
25 12 1
25 13 1
25 14 1
25 15 1
25 16 1
25 17 1
25 18 1
25 19 1
25 20 1
25 21 1
25 22 1
25 23 1
25 24 1
25 25 1
25 26 1
25 27 1
```

```
25 28 1
25 29 1
25 30 1
25 31 1
25 32 1
25 33 1
25 34 1
25 35 9
24 35 1
23 35 1
22 35 1
21 35 1
20 35 1
19 35 1
18 35 1
17 35 1
16 35 1
15 35 1
14 35 1
13 35 1
12 35 1
11 35 1
10 35 1
9 35 1
8 35 1
7 35 1
6 35 9
6 34 1
6 33 1
6 32 1
6 31 1
6 30 1
6 29 1
6 28 1
6 27 1
6 26 1
6 25 1
6 24 1
6 23 1
6 22 1
6 21 1
6 20 1
6 19 1
6 18 1
```

```
6 17 1
6 16 1
6 15 1
6 14 1
6 13 1
6 12 1
6 11 1
6 10 1
6 9 1
6 8 1
6 7 1
```

- outFile2 for dataA

When K = 4

```
30 40 0 1
1
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . 1 1 1 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 1 1 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

When K = 2

```
30 40 0 1
1
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . 1 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

When K = 1

```
30 40 0 1
1
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . .
. . . . . . 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

- outFile3 for dataA

When K = 1

```
Print Boundary Points Array:
6 6
7 6
8 6
9 6
10 6
11 6
12 6
13 6
14 6
15 6
16 6
17 6
18 6
19 6
20 6
21 6
22 6
23 6
24 6
25 6
25 7
25 8
25 9
25 10
25 11
25 12
25 13
25 14
25 15
25 16
25 17
25 18
25 19
25 20
25 21
25 22
25 23
25 24
25 25
25 26
25 27
25 28
25 29
25 30
25 31
25 32
25 33
25 34
25 35
24 35
23 35
22 35
21 35
20 35
19 35
18 35
17 35
16 35
15 35
14 35
13 35
12 35
11 35
10 35
9 35
8 35
7 35
6 35
6 34
6 33
6 32
6 31
6 30
6 29
6 28
6 27
6 26
6 25
6 24
6 23
6 22
6 21
6 21
6 20
6 19
6 18
6 17
6 16
6 15
6 14
6 13
6 12
6 11
6 10
6 9
6 8
6 7

row col curvature
7 6 0
8 6 0
9 6 0
10 6 0
11 6 0
12 6 0
13 6 0
14 6 0
15 6 0
16 6 0
17 6 0
18 6 0
19 6 0
20 6 0
21 6 0
22 6 0
23 6 0
24 6 0
25 6 2
25 7 0
25 8 0
25 9 0
25 10 0
25 11 0
25 12 0
```

```
25 12 0
25 13 0
25 14 0
25 15 0
25 16 0
25 17 0
25 18 0
25 19 0
25 20 0
25 21 0
25 22 0
25 23 0
25 24 0
25 25 0
25 26 0
25 27 0
25 28 0
25 29 0
25 30 0
25 31 0
25 32 0
25 33 0
25 34 0
25 35 2
24 35 0
23 35 0
22 35 0
21 35 0
20 35 0
19 35 0
18 35 0
17 35 0
16 35 0
15 35 0
14 35 0
13 35 0
12 35 0
11 35 0
10 35 0
9 35 0
8 35 0
7 35 0
```

```
6 35 2
6 34 0
6 33 0
6 32 0
6 31 0
6 30 0
6 29 0
6 28 0
6 27 0
6 26 0
6 25 0
6 24 0
6 23 0
6 22 0
6 21 0
6 20 0
6 19 0
6 18 0
6 17 0
6 16 0
6 15 0
6 14 0
6 13 0
6 12 0
6 11 0
6 10 0
6 9 0
6 8 0
6 7 0
6 6 2
```

- outFile1 for dataB
When K = 1

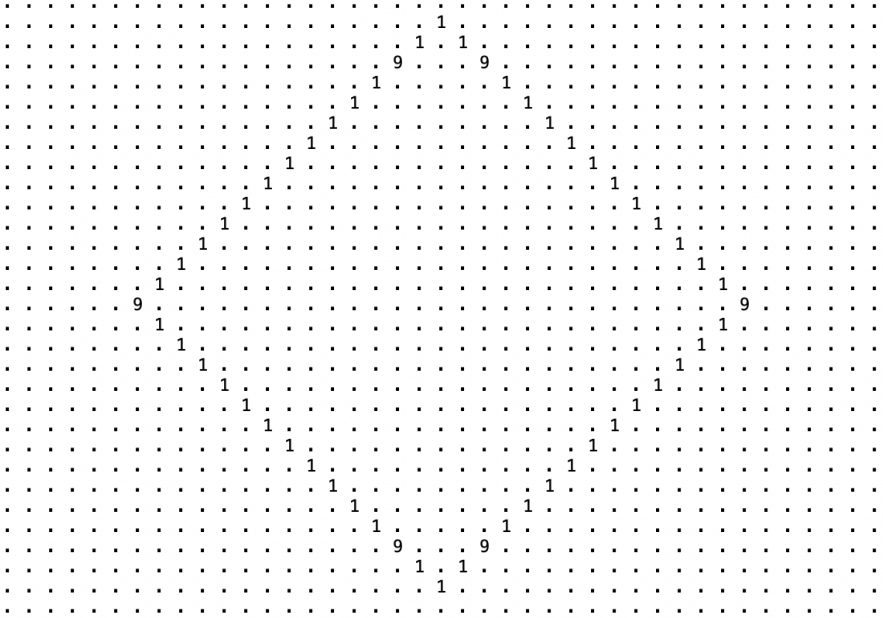
```
31 41 0 1
1
56
1 20 9
2 19 1
3 18 1
4 17 1
5 16 1
6 15 1
7 14 1
8 13 1
9 12 1
10 11 1
11 10 1
12 9 1
13 8 1
14 7 1
15 6 9
16 7 1
17 8 1
18 9 1
19 10 1
20 11 1
21 12 1
22 13 1
23 14 1
24 15 1
25 16 1
26 17 1
27 18 1
28 19 1
29 20 9
```

```
28 21 1
27 22 1
26 23 1
25 24 1
24 25 1
23 26 1
22 27 1
21 28 1
20 29 1
19 30 1
18 31 1
17 32 1
16 33 1
15 34 9
14 33 1
13 32 1
12 31 1
11 30 1
10 29 1
9 28 1
8 27 1
7 26 1
6 25 1
5 24 1
4 23 1
3 22 1
2 21 1
```

- outFile2 for dataB

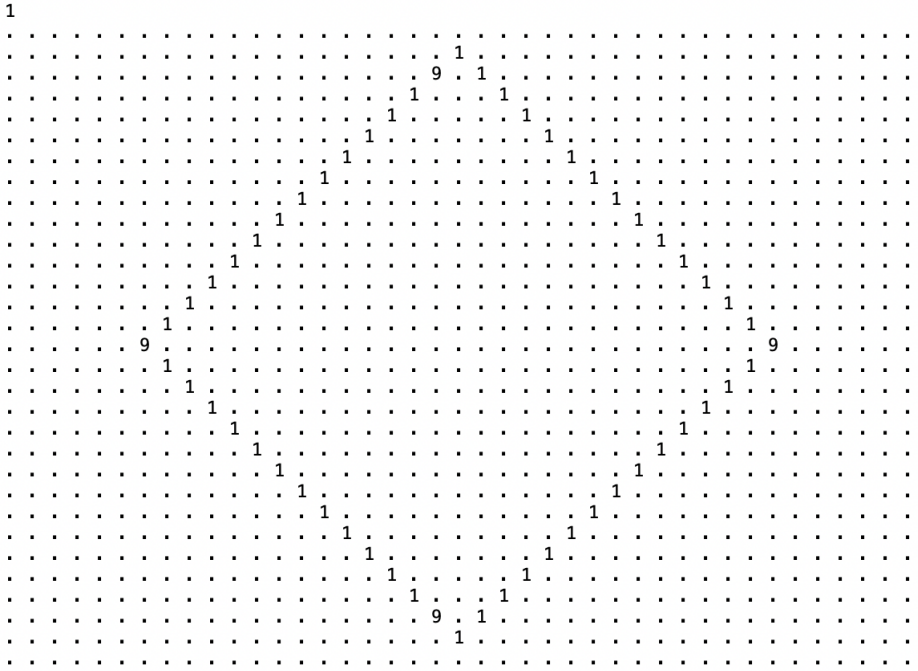
When K = 4

```
31 41 0 1
1
```

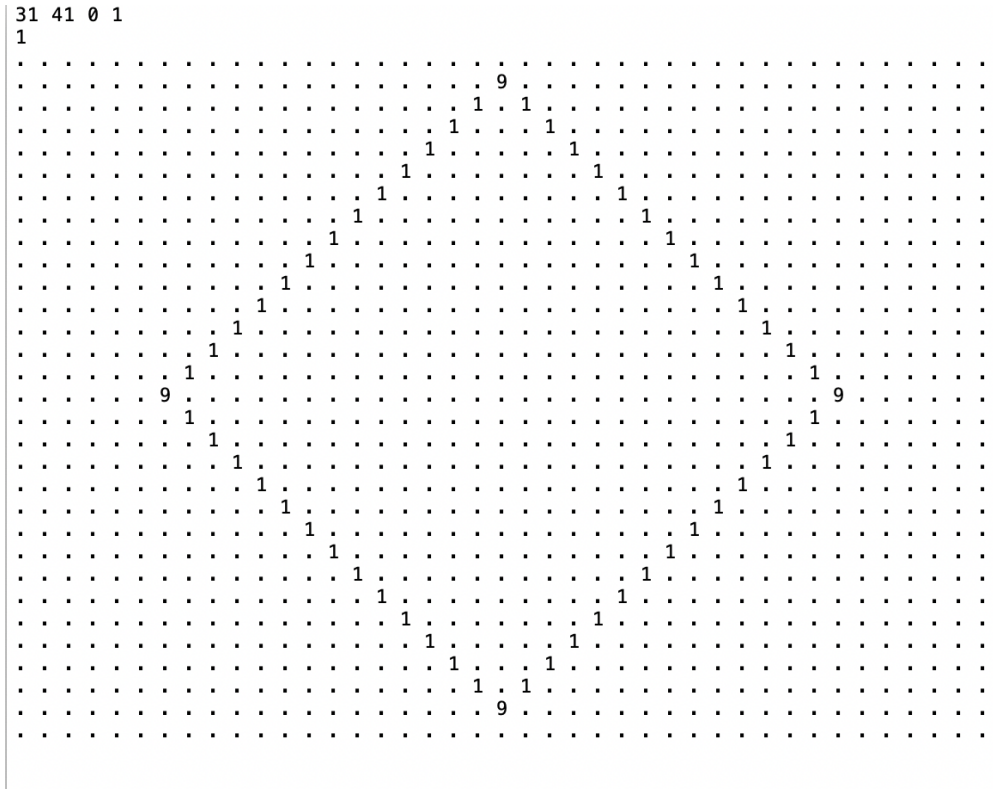


When K = 2

```
31 41 0 1
1
```



When K = 1



- outFile3 for dataB
When K = 1

```
Print Boundary Points Array:
1 20
2 19
3 18
4 17
5 16
6 15
7 14
8 13
9 12
10 11
11 10
12 9
13 8
14 7
15 6
16 7
17 8
18 9
19 10
20 11
21 12
22 13
23 14
24 15
25 16
26 17
27 18
28 19
29 20
28 21
27 22
26 23
25 24
24 25
23 26
22 27
21 28
20 29
19 30
18 31
17 32
16 33
15 34
14 33
13 32
12 31
11 30
10 29
9 28
8 27
7 26
6 25
5 24
4 23
3 22
2 21

row col curvature
2 19 0
3 18 0
4 17 0
5 16 0
6 15 0
7 14 0
8 13 0
9 12 0
10 11 0
11 10 0
12 9 0
13 8 0
14 7 0
15 6 2
16 7 0
17 8 0
18 9 0
19 10 0
20 11 0
21 12 0
22 13 0
22 13 0
23 14 0
24 15 0
25 16 0
26 17 0
27 18 0
28 19 0
29 20 2
28 21 0
27 22 0
26 23 0
25 24 0
24 25 0
23 26 0
22 27 0
21 28 0
20 29 0
19 30 0
18 31 0
17 32 0
16 33 0
15 34 2
14 33 0
13 32 0
12 31 0
11 30 0
10 29 0
9 28 0
8 27 0
7 26 0
6 25 0
5 24 0
4 23 0
3 22 0
2 21 0
1 20 2
```

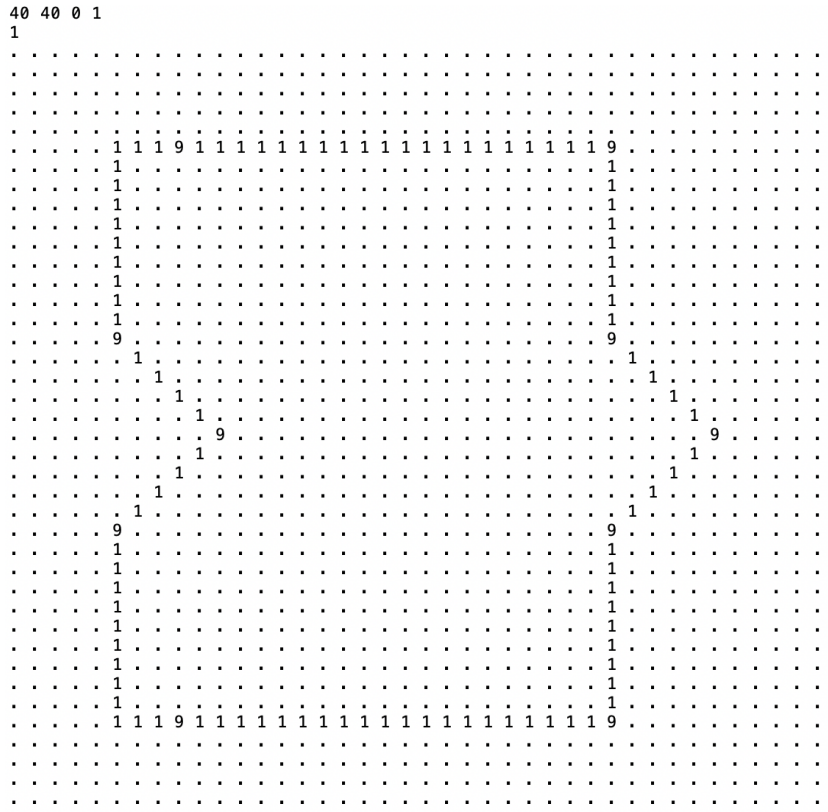
- outFile1 for dataC
When K = 1

```
40 40 0 1
1
108
5 5 9
6 5 1
7 5 1
8 5 1
9 5 1
10 5 1
11 5 1
12 5 1
13 5 1
14 5 1
15 5 9
16 6 1
17 7 1
18 8 1
19 9 1
20 10 9
21 9 1
22 8 1
23 7 1
24 6 1
25 5 9
26 5 1
27 5 1
28 5 1
29 5 1
30 5 1
31 5 1
32 5 1
33 5 1
34 5 1
35 5 9
35 6 1
35 7 1
35 8 1
35 9 1
35 10 1
35 11 1
35 12 1
35 13 1
35 14 1
35 15 1
```

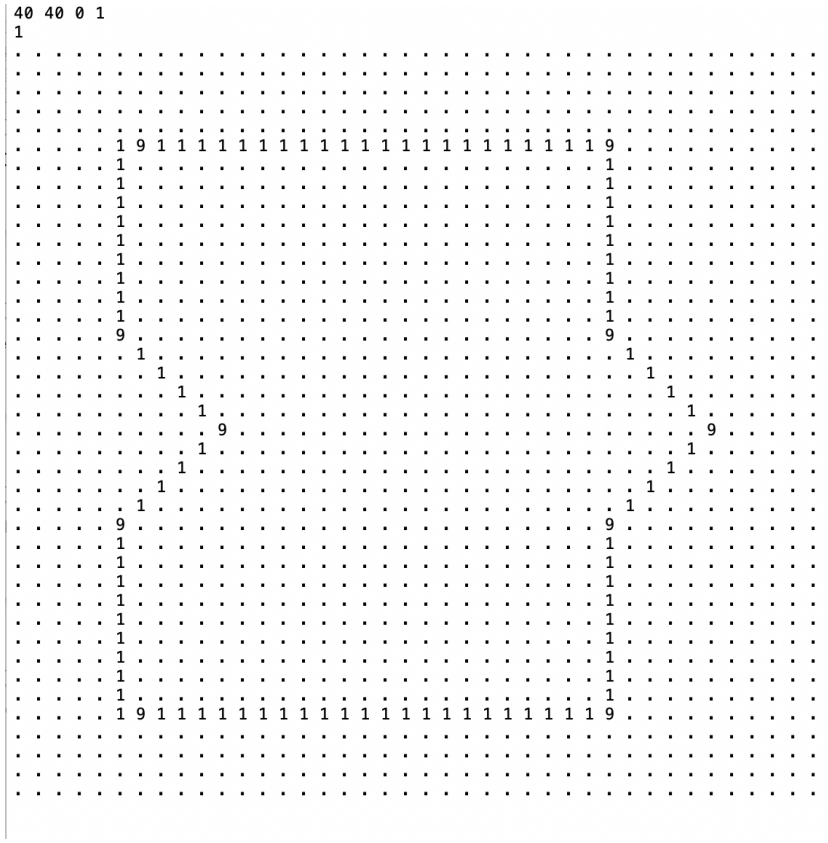
```
35 16 1
35 17 1
35 18 1
35 19 1
35 20 1
35 21 1
35 22 1
35 23 1
35 24 1
35 25 1
35 26 1
35 27 1
35 28 1
35 29 9
34 29 1
33 29 1
32 29 1
31 29 1
30 29 1
29 29 1
28 29 1
27 29 1
26 29 1
25 29 9
24 30 1
23 31 1
22 32 1
21 33 1
20 34 9
19 33 1
18 32 1
17 31 1
16 30 1
15 29 9
14 29 1
13 29 1
12 29 1
11 29 1
10 29 1
9 29 1
8 29 1
7 29 1
6 29 1
5 29 9
```

```
5 28 1
5 27 1
5 26 1
5 25 1
5 24 1
5 23 1
5 22 1
5 21 1
5 20 1
5 19 1
5 18 1
5 17 1
5 16 1
5 15 1
5 14 1
5 13 1
5 12 1
5 11 1
5 10 1
5 9 1
5 8 1
5 7 1
5 6 1
```

- outFile2 for dataC
When K = 4



When K = 2



When K = 1

```
40 40 0 1
1
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 9 . . . . . . . . . . . . . . . . . . . . . . . 9 . . . . . . . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . .
. . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . .
. . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . .
. . . . . . . . . . 9 . . . . . . . . . . . . . . . . . . . . . . . 9 . . . . .
. . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . .
. . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . .
. . . . . 9 . . . . . . . . . . . . . . . . . . . . . . . 9 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 9 . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

- outFile3 for dataC

When K = 1

Print Boundary Points Array:
5 5
6 5
7 5
8 5
9 5
10 5
11 5
12 5
13 5
14 5
15 5
16 6
17 7
18 8
19 9
20 10
21 9
22 8
23 7
24 6
25 5
26 5
27 5
28 5
29 5
30 5
31 5
32 5
33 5
34 5
35 5
35 6
35 7
35 8
35 9
35 10
35 11
35 12
35 13
35 14
35 15
35 16
35 17
35 18
35 19
35 20
35 21
35 22
35 23
35 24
35 25
35 26
35 27
35 28
35 29
34 29
33 29
32 29
31 29
30 29
29 29
28 29
27 29
26 29
25 29
24 30
23 31
22 32
21 33
20 34
19 33
18 32
17 31
16 30
15 29
14 29
13 29
12 29
11 29
10 29
9 29
8 29
7 29
6 29
5 29
5 28
5 27
5 26
5 25
5 24
5 23
5 22
5 21
5 20
5 19
5 18
5 17
5 16
5 15
5 14
5 13
5 12
5 11
5 10
5 9
5 8
5 7
5 6

row col curvature
6 5 0
7 5 0
8 5 0
9 5 0
10 5 0
11 5 0
12 5 0
13 5 0
14 5 0
15 5 1
16 6 0
17 7 0
18 8 0
19 9 0
20 10 2
21 9 0
22 8 0
23 7 0
24 6 0
25 5 1
26 5 0
27 5 0
28 5 0
29 5 0
30 5 0
31 5 0
32 5 0
33 5 0
34 5 0
35 5 2
35 6 0
35 7 0
35 8 0
35 9 0
35 10 0
35 11 0
35 12 0
35 13 0
35 14 0
35 15 0
35 16 0
35 17 0
35 18 0
35 19 0
35 20 0
35 21 0
35 22 0
35 23 0

```
35 24 0
35 25 0
35 26 0
35 27 0
35 28 0
35 29 2
34 29 0
33 29 0
32 29 0
31 29 0
30 29 0
29 29 0
28 29 0
27 29 0
26 29 0
25 29 1
24 30 0
23 31 0
22 32 0
21 33 0
20 34 2
19 33 0
18 32 0
17 31 0
16 30 0
15 29 1
14 29 0
13 29 0
12 29 0
11 29 0
10 29 0
```

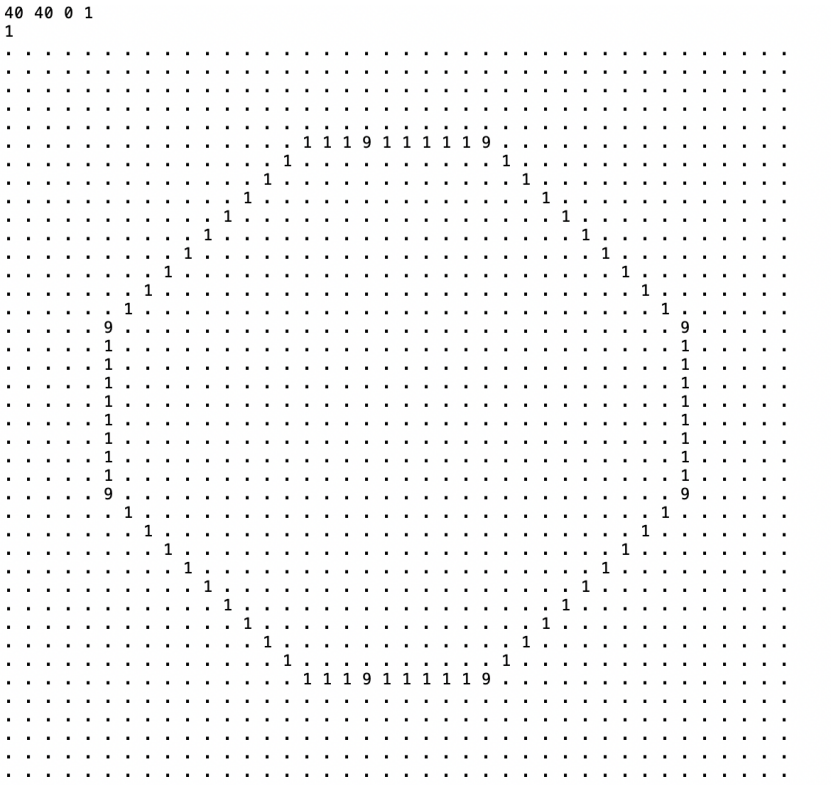
```
9 29 0
8 29 0
7 29 0
6 29 0
5 29 2
5 28 0
5 27 0
5 26 0
5 25 0
5 24 0
5 23 0
5 22 0
5 21 0
5 20 0
5 19 0
5 18 0
5 17 0
5 16 0
5 15 0
5 14 0
5 13 0
5 12 0
5 11 0
5 10 0
5 9 0
5 8 0
5 7 0
5 6 0
5 5 2
```

- outFile1 for dataD
When K = 1

```
40 40 0 1
1
76
5 15 9
6 14 1
7 13 1
8 12 1
9 11 1
10 10 1
11 9 1
12 8 1
13 7 1
14 6 1
15 5 9
16 5 1
17 5 1
18 5 1
19 5 1
20 5 1
21 5 1
22 5 1
23 5 1
24 5 9
25 6 1
26 7 1
27 8 1
28 9 1
29 10 1
30 11 1
31 12 1
32 13 1
33 14 1
34 15 9
34 16 1
34 17 1
34 18 1
34 19 1
34 20 1
34 21 1
34 22 1
34 23 1
34 24 9
33 25 1
32 26 1
31 27 1
30 28 1
29 29 1
28 30 1
27 31 1
26 32 1
25 33 1
24 34 9
23 34 1
22 34 1
21 34 1
20 34 1
19 34 1
18 34 1
17 34 1
16 34 1
15 34 9
14 33 1
13 32 1
12 31 1
11 30 1
10 29 1
9 28 1
8 27 1
7 26 1
6 25 1
5 24 9
5 23 1
5 22 1
5 21 1
5 20 1
5 19 1
5 18 1
5 17 1
5 16 1
```

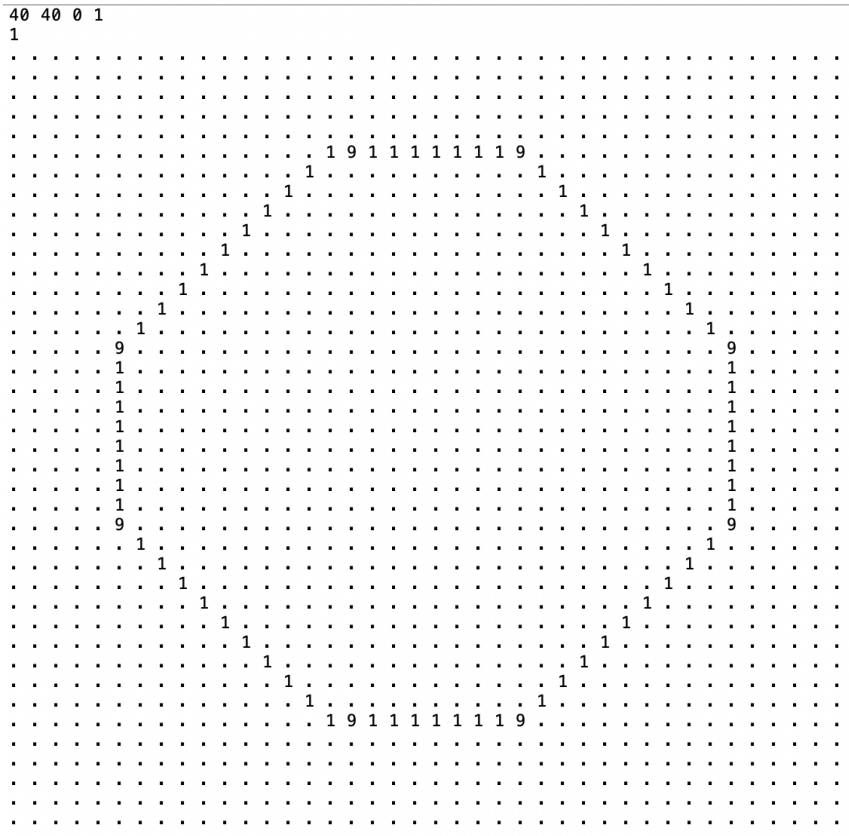
- outFile2 for dataD
When K = 4

```
40 40 0 1
1
```



When K = 2

```
40 40 0 1
1
```



When K = 1

```
40 40 0 1
1
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . 9 1 1 1 1 1 1 1 1 9 . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . 1 . . . . . . . . . . 1 . . . . . . . . . . . . . .
. . . . . . . . . . . . . 1 . . . . . . . . . . . . 1 . . . . . . . . . . . . .
. . . . . . . . . . . . 1 . . . . . . . . . . . . . . 1 . . . . . . . . . . . .
. . . . . . . . . . . 1 . . . . . . . . . . . . . . . . 1 . . . . . . . . . . .
. . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . .
. . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . .
. . . . . 9 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . .
. . . . . 9 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9 . . . . .
. . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . .
. . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . .
. . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . .
. . . . . . . . . 1 . . . . . . . . . . . . . . . . . . . . 1 . . . . . . . . .
. . . . . . . . . . 1 . . . . . . . . . . . . . . . . . . 1 . . . . . . . . . .
. . . . . . . . . . . 1 . . . . . . . . . . . . . . . . 1 . . . . . . . . . . .
. . . . . . . . . . . . 1 . . . . . . . . . . . . . . 1 . . . . . . . . . . . .
. . . . . . . . . . . . . 1 . . . . . . . . . . . . 1 . . . . . . . . . . . . .
. . . . . . . . . . . . . . 1 . . . . . . . . . . 1 . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . 9 1 1 1 1 1 1 1 1 9 . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

- outFile3 for dataD

Print Boundary Points Array:
5 15
6 14
7 13
8 12
9 11
10 10
11 9
12 8
13 7
14 6
15 5
16 5
17 5
18 5
19 5
20 5
21 5
22 5
23 5
24 5
25 6
26 7
27 8
28 9
29 10
30 11
31 12
32 13
33 14
34 15
34 16
34 17
34 18
34 19
34 20
34 21
34 22
34 23
34 24
33 25
32 26
31 27
30 28
29 29
28 30
27 31
26 32
25 33
24 34
23 34
22 34
21 34
20 34
19 34
18 34
17 34
16 34
15 34
14 33
13 32
12 31
11 30
10 29
9 28
8 27
7 26
6 25
5 24
5 23
5 22
5 21
5 20
5 19
5 18
5 17
5 16

row col curvature
6 14 0
7 13 0
8 12 0
9 11 0
10 10 0
11 9 0
12 8 0
13 7 0
14 6 0
15 5 1
16 5 0
17 5 0
18 5 0
19 5 0
20 5 0
21 5 0
22 5 0
23 5 0
24 5 1
25 6 0
26 7 0
27 8 0
28 9 0
29 10 0
30 11 0
31 12 0
32 13 0
33 14 0
34 15 3
34 16 0
34 17 0
34 18 0
34 19 0
34 20 0
34 21 0
34 22 0
34 23 0
34 24 1
33 25 0
32 26 0
31 27 0
30 28 0
29 29 0
28 30 0
27 31 0
26 32 0
25 33 0
24 34 1
23 34 0
22 34 0
21 34 0
20 34 0
19 34 0
18 34 0
17 34 0
16 34 0
15 34 1
14 33 0
13 32 0
12 31 0
11 30 0
10 29 0
9 28 0
8 27 0
7 26 0
6 25 0
5 24 1
5 23 0
5 22 0
5 21 0
5 20 0
5 19 0
5 18 0
5 17 0
5 16 0
5 15 3