CV Programming Language: CPP

Mock Final 2 Mock Final 2

Andres Quintero

(1) Open the three files: the binary image, SumRows and SumCols using argv or args

(2) Pretty Print the binary image and output the pretty print to a txt file (called it imgPrettyPrint)

(3) Use an array, sumRowsAry to store SumRows, and another array, sumColsAry, to store SumCols

(4) Output sumRowsAry (without indexes) to a text file (called it imgHPP) with the image header.

(5) Output sumColsAry (without indexes) to a text file (called it imgVPP), as that of imgHPP.

(6) Turn sumRowsAry to a binary 1-D array, using threshould value = 4 (i.e., a value >= 4 turns to 1 and less turns to 0), thenOutput the binary array to a text file (called it imgHPPbin.)

(7) Turn sumColsAry to a binary 1D array, using threshould value = 4 (i.e., a value >= 4 turns to 1 and less turns to 0), then Output the binary array to a text file (called it imgVPPbin.)

**Data Structures**

Class Image

numRow -int

numCol -int

minVal -int

maxVal -int

imageAry -int\*\*

loadImage(imageFile)-

1. numRows, numCol <- from imageFile
2. allocate imageAry numRow by numCol
3. i <- 0
4. j <- 0
5. input <- read from imageFile
6. imageAry[i][j] <- input
7. j++
8. repeat 4-7 while j < numCols
9. i++
10. repeat 3-9 while I < numRows

prettyPrint(imgPrettyPrint)-

1. imgPrettyPrint <- given
2. read imageAry from L to R, T to B

2- value <- from imageAry read

1. if(value > 0) print value ELSE print space
2. repeat 1 -4 until done with imgAry read

Class RCAry

numRow -int

numCol -int

minVal -int

maxVal -int

thresholdVal -int (**4)**

sumRowsAry -int\*

sumColsAry -int\*

loadRowAry(rowFile)

1. rowFile <- given

numRow, numCol, minVal, maxVal <- read from rowFile header

1. index <- read from rowFile
2. sum <- read from rowFile
3. sumRowAry[index] <- sum
4. repeat 1-4 until rowFile.eof()

printRowAry(imgHPP, imgHPPbin)

0– imgHPP, imgHPPbin <- given

1- output image header to imgHPP

2- i <- 0

1. sumRowsAry[i] -> imgHPP
2. i++
3. repeat 3-5 while i < numRows
4. i <- 0
5. if (sumRowsAry[i] >= threshold) print 1 else print 0 to imgHPPbin
6. i++
7. repeat7-9 while i < numRows

loadColAry(colFile)

1. colFile <- given

numRow, numCol, minVal, maxVal <- read from colFile header

1. index <- read from colFile
2. sum <- read from colFile
3. sumColsAry[index] <- sum
4. repeat 1-4 until colFile.eof()

printRowAry(imgVPP, imgVPPbin)

0– imgVPP, imgVPPbin <- given

1- output image header to imgVPP

2- i <- 0

1. sumColsAry[i] -> imgVPP
2. i++
3. repeat 3-5 while i < numCols
4. i <- 0
5. if (sumColsAry[i] >= threshold) print 1 else print 0 to imgVPPbin
6. i++
7. repeat7-9 while i < numCols

**SOURCE CODE**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

class Image{

public:

int numRows;

int numCols;

int minVal;

int maxVal;

int\*\* imageAry;

void loadImageAry(ifstream& inFile){

imageAry = new int\*[numRows];

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

imageAry[i] = new int[numCols];

}

int input;

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

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

inFile >> input;

imageAry[i][j] = input;

}

}

}

void prettyPrint(ofstream& outFile){

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

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

if(imageAry[i][j] > 0){

outFile << imageAry[i][j] << " ";

} else {

outFile << " "; // two spaces

}

}

outFile << endl;

}

}

};

class RCAry{

public:

int numRows;

int numCols;

int minVal;

int maxVal;

int thresholdVal;

int\* sumRowsAry;

int\* sumColAry;

void loadRowAry(ifstream& rowFile){

sumRowsAry = new int[numRows];

// image header has been read by now so now

// pairs (index, sum)

int index, sum;

while(!rowFile.eof()){

rowFile >> index;

rowFile >> sum;

sumRowsAry[index] = sum;

}

}

void loadColAry(ifstream& colFile){

sumColAry = new int[numCols];

// pairs (index, sum)

int index, sum;

while(!colFile.eof()){

colFile >> index;

colFile >> sum;

sumColAry[index] = sum;

}

}

void printRowAry(ofstream& imgHPP, ofstream& imgHPPbin){

// Header first

imgHPP << numRows << " " << numCols << " " << minVal << " " << maxVal << endl;

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

imgHPP << sumRowsAry[i] << endl;

}

// 1-D array with threshold 4

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

if(sumRowsAry[i] >= thresholdVal){

imgHPPbin << "1 "; // per-line not specified

} else {

imgHPPbin << "0 ";

}

}

}

void printColAry(ofstream& imgVPP, ofstream& imgVPPbin){

// Header

imgVPP << numRows << " " << numCols << " " << minVal << " " << maxVal << endl;

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

imgVPP << sumColAry[i] << endl;

}

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

if(sumColAry[i] >= thresholdVal){

imgVPPbin << "1 ";

} else {

imgVPPbin << "0 ";

}

}

}

};

int main(int argc, char\* argv[]){

ifstream imageFile(argv[1]);

ifstream rowFile(argv[2]);

ifstream colFile(argv[3]);

ofstream imgPrettyPrint(argv[4]);

ofstream imgHPP(argv[5]);

ofstream imgVPP(argv[6]);

ofstream imgHPPbin(argv[7]);

ofstream imgVPPbin(argv[8]);

Image image;

// Need to read first

imageFile >> image.numRows;

imageFile >> image.numCols;

imageFile >> image.minVal;

imageFile >> image.maxVal;

image.loadImageAry(imageFile);

image.prettyPrint(imgPrettyPrint);

RCAry Arys;

Arys.thresholdVal = 4;

//Row stuff

rowFile >> Arys.numRows;

rowFile >> Arys.numCols;

rowFile >> Arys.minVal;

rowFile >> Arys.maxVal;

Arys.loadRowAry(rowFile);

Arys.printRowAry(imgHPP, imgHPPbin);

//Col stuff

colFile >> Arys.numRows; // Still need to read these

colFile >> Arys.numCols;

colFile >> Arys.minVal;

colFile >> Arys.maxVal;

Arys.loadColAry(colFile);

Arys.printColAry(imgVPP, imgVPPbin);

// Closing files

imageFile.close();

rowFile.close();

colFile.close();

imgPrettyPrint.close();

imgHPP.close();

imgVPP.close();

imgHPPbin.close();

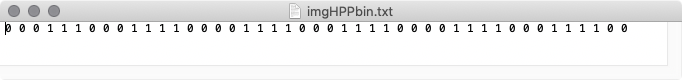
imgVPPbin.close();

}

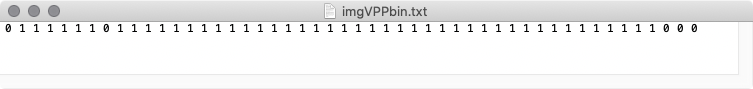
**OUTPUTSA screenshot of a cell phone

Description automatically generated**

**A close up of a logo

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**