CS 323\_33 Programming Language: C++

Project #6 RunningLength\_Methods1&4

Andres Quintero

Due Date:

Soft copy: 3/24/2020

Hard copy: 2/24/2020

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Main():

step 0: inFile 🡨 open argv[1]

outFile1 🡨 open argv[3]

step 1: numRows, numCols, minVal, maxVal 🡨 Read from inFile

step 2: whichMethod 🡨 from argv[2]

step 3: nameEncodeFile 🡨 argv[1] + “\_EncodeMethod” + “whichMethod”

step 4: encodeFile 🡨open (nameEncodeFile)

step 5: output numRows, numCols, minVal, maxVal to encodeFile

output whichMethod to encodeFile

step 6: case of whichMethod

case 1: encodeMethod1 (inFile, encodeFile)

case 4: encodeMethod4 (inFile, encodeFile)

default: error message

Step 7: close all files

**Source code:**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

// Prototypes

void encodeMethod1(fstream &inFile, fstream& encodeFile, int numRows, int numCols);

void enocdeMethod4(fstream& inFile, fstream& encodeFile, int numRows, int numCols);

int skipZeros(fstream& inFile, int& row, int& col, int& zeroCount, int numCols);

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

fstream inFile(argv[1]);

//Varibles

int numRows, numCols, minVal, maxVal;

int whichMethod;

// Reading image header from inFile

inFile >> numRows;

inFile >> numCols;

inFile >> minVal;

inFile >> maxVal;

whichMethod = stoi(argv[2]);

string fileName = argv[1];

string methodNumber = argv[2];

string nameEncodeFile = fileName + "\_EncodeMethod" + methodNumber + ".txt";

fstream encodeFile(nameEncodeFile, fstream::out);

//Writing image header to encodeFile

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

encodeFile << whichMethod << endl;

if (whichMethod == 1){

encodeMethod1(inFile, encodeFile, numRows, numCols);

} else if (whichMethod == 4) {

enocdeMethod4(inFile, encodeFile, numRows, numCols);

} else {

encodeFile << "Error in endcoding" << endl;

}

//closing files

inFile.close();

encodeFile.close();

}

// Functions

int skipZeros(fstream& inFile, int& row, int& col, int& zeroCount, int numCols){

int pixelVal;

zeroCount = 0;

inFile >> pixelVal;

while(pixelVal == 0){

inFile >> pixelVal;

col++;

if(col == numCols){

col = 0;

row++;

}

if(pixelVal == 0){

zeroCount++;

}

}

return pixelVal;

}

void enocdeMethod4(fstream& inFile, fstream& encodeFile, int numRows, int numCols){

int nextVal, zeroCount;

int row = 0;

int col = 0;

int length = 1;

int lastVal = skipZeros(inFile, row, col, zeroCount, numCols);

encodeFile << row << " " << col << " " << lastVal << " ";

while(inFile >> nextVal){

col++;

if(col == numCols){

col = 0;

row++;

}

if(nextVal == 0){

inFile >> nextVal;

col++;

if(col == numCols){

col = 0;

row++;

}

lastVal = 0;

} else if(nextVal == lastVal){

length++;

} else {

encodeFile << length << endl;

encodeFile << row << " " << col << " " << nextVal << " ";

length = 1;

lastVal = nextVal;

}

}

encodeFile << length << endl;

}

void encodeMethod1(fstream& inFile, fstream& encodeFile, int numRows, int numCols){

int row, col, length, currVal, nextVal;

row = 0;

while (row < numRows){

//1

col = 0;

length = 0;

inFile >> currVal;

encodeFile << row << " " << col << " " << currVal << " "; // first output before an checks

length++;

while(col < numCols-1){ //negative one to count for the offset of outputing once already

//2

col++;

//3

inFile >> nextVal;

//4

if(nextVal == currVal){

length++;

} else {

encodeFile << length << endl;

currVal = nextVal;

encodeFile << row << " " << col << " " << currVal << " ";

length = 1;

}

} // 5 (loop)

//6

encodeFile << length << endl;

row++;

}

}

**END OF SOURCE CODE**

Input images: **image1.txt** and **image2.txt**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**End input images**

image1.txt via **Method 1 and Method 4**

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

**END OF IMAGE1.TXT OUTPUT**

Image2.txt via **Method 1 and Method 4**

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

**END OF IMAGE2.TXT OUTPUT**