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**

image1.txt

```
10 22 0 9
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 4 4 4 4 4 4
4 0 4 4 4 4 4 4 4 4 4 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 3 0 0 3 3 3 3 3 3 7 7 7 7 7 7 7 7 7 7 7
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
7 7 0 0 0 0 0 2 3 4 2 2 3 3 4 4 4 4 4 4 0 0
0 0 0 0 0 0 1 1 1 1 1 9 9 9 9 9 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 6 6 6 6 6 6 6 6 6 6 6 6
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

image2.txt

```
20 22 0 9
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 4 4 4 4 4 4
4 0 4 4 4 4 4 4 4 4 4 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 3 0 0 3 3 3 3 3 3 7 7 7 7 7 7 7 7 7 7 7
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
7 7 0 0 0 0 0 2 3 4 2 2 3 3 4 4 4 4 4 4 0 0
0 0 0 0 0 0 1 1 1 1 1 9 9 9 9 9 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 6 6 6 6 6 6 6 6 6 6 6 6
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
7 7 0 0 0 0 0 2 3 4 2 2 3 3 4 4 4 4 4 4 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

End input images

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

image1.txt_EncodeMethod1.txt

```
10 22 0 9
1
0 0 0 15
0 15 4 7
1 0 4 1
1 1 0 1
1 2 4 9
1 11 0 11
2 0 0 5
2 5 3 17
3 0 3 3
3 3 0 2
3 5 3 6
3 11 7 11
4 0 7 22
5 0 7 2
5 2 0 5
5 7 2 1
5 8 3 1
5 9 4 1
5 10 2 2
5 12 3 2
5 14 4 6
5 20 0 2
6 0 0 6
6 6 1 5
6 11 9 5
6 16 1 6
7 0 1 10
7 10 6 12
```

image1.txt_EncodeMethod4.txt

```
10 22 0 9
4
0 15 4 8
1 3 4 8
2 5 3 20
3 5 3 6
3 11 7 35
5 8 3 1
5 9 4 1
5 10 2 2
5 12 3 2
5 14 4 6
6 6 1 5
6 11 9 5
6 16 1 16
7 10 6 12
```

END OF IMAGE1.TXT OUTPUT

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

image2.txt_EncodeMethod1.txt

```
20 22 0 9
1
0 0 0 15
0 15 4 7
1 0 4 1
1 1 0 1
1 2 4 9
1 11 0 11
2 0 0 5
2 5 3 17
3 0 3 3
3 3 0 2
3 5 3 6
3 11 7 11
4 0 7 22
5 0 7 2
5 2 0 5
5 7 2 1
5 8 3 1
5 9 4 1
5 10 2 2
5 12 3 2
5 14 4 6
5 20 0 2
6 0 0 6
6 6 1 5
6 11 9 5
6 16 1 6
7 0 1 10
7 10 6 12
8 0 0 22
9 0 0 22
10 0 0 22
11 0 0 22
12 0 0 22
13 0 0 22
14 0 7 22
15 0 7 2
15 2 0 5
15 7 2 1
15 8 3 1
15 9 4 1
15 10 2 2
15 12 3 2
15 14 4 6
15 20 0 2
16 0 0 22
17 0 0 22
18 0 0 22
19 0 0 22
```

image2.txt_EncodeMethod4.txt

```
20 22 0 9
4
0 15 4 8
1 3 4 8
2 5 3 20
3 5 3 6
3 11 7 35
5 8 3 1
5 9 4 1
5 10 2 2
5 12 3 2
5 14 4 6
6 6 1 5
6 11 9 5
6 16 1 16
7 10 6 12
14 0 7 24
15 8 3 1
15 9 4 1
15 10 2 2
15 12 3 2
15 14 4 6
```

END OF IMAGE2.TXT OUTPUT