CS 700-34 **RunLengthEncode**

Student: Shuhua Song

Due Date:	**Submission Data:**
Soft Copy: 03/29/2020	Soft Copy: 03/26/2020
Hard Copy: 03/29/2020	Hard Copy: 03/26/2020

Algorithm Steps:

I. decodeMethod1(encodeFile, decodeFile)

// encoded with zero and not wrap-around.

Step 0: RowCnt ← 0

Step 1: colCnt ← 0

Step 2: startRow, startCol, color, length ← read from encodeFile // get a run

Step 3: decodeFile ← output length times of color to decodeFile

Step 4: colCnt ++

Step 5: if colCnt >= numCols (or colCnt % numCols == 0)
deCodeFile ← print end of text line. (cout << endl)
colCnt ← 0
rowCnt ++

Step 6: repeat step 2 to step 5 while rowCnt < numRows and not EOF (encodeFile)

II. decodeMethod4(encodeFile, decodeFile)

// encoded without zero and wrap-around.

Step 0: RowCnt ← 0

Step 1: colCnt ← 0
newLength ← 0

Step 2: startRow, startCol, color, length ← read from encodeFile // get a run

Step 3: if (startRow !=0 or startCol != 0)
Calculate
newLength = startRow * 22 + startCol
decodeFile←output length time of "0" to decodeFile in the begining

Step 4: decodeFile ← output length times of color to decodeFile

Step 5: startRow, startCol, color, length ← read from encodeFile // get a run

Step 6: if (rowCnt == startRow and colCnt != startCol)

Calculate
newLength = startCol – colCnt
decodeFile←output length time of "0" to decodeFile

Step 6: if (rowCnt != startRow or colCnt != startCol)
Calculate
newLength = (startRow-rowCnt-1)*22 + (22-colCnt) + startCol;
decodeFile←output length time of "0" to decodeFile

Step 7: repeat step 4- step 6 while rowCnt < numRows and not EOF (encodeFile)

Step 8: check if (rowCnt < numRows and colCnt != 21)
Calculate
newLength = (startRow-rowCnt-1)*22 + (22-colCnt) + startCol;
decodeFile←output length time of "0" to decodeFile

****

III. addZero (length, decodeFile)

Step 0: decodeFile ← output "0" to decodeFile

Step 1: colCnt++

Step 2: if colCnt >= numCols (or colCnt % numCols == 0)

deCodeFile ← print end of text line. (cout << endl)

colCnt ← 0

rowCnt ++

Step 3: repeat Step 0 – Step 2 until i < length;

IV. addNonZero (length, decodeFile, color)

Step 0: decodeFile ← output color to decodeFile

Step 1: colCnt++

Step 2: if colCnt >= numCols (or colCnt % numCols == 0)

deCodeFile ← print end of text line. (cout << endl)

colCnt ← 0

rowCnt ++

Step 3: repeat Step 0 – Step 2 until i < length;

Code:

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

int rowCnt;
int colCnt;
//int leftEndPrint;
//int rightEndPrint;
class RunLength{
public:
    int numRows;
    int numCols;
    int minVal;
    int maxVal;

    int whichMethod;
    string nameEncodeFile;
    string nameDecodeFile;

    RunLength(){
        numRows = 0; numCols = 0; minVal = 0; maxVal = 0;
    }

    RunLength(int numRows, int numCols,int minVal, int maxVal){
        this->numRows = numRows;
        this->numCols = numCols;
        this->minVal = minVal;
        this->maxVal = maxVal;
    }

    void deCodeMethod1(ifstream& inFile, ofstream& outFile){ //with zero, no wrap-
around
         rowCnt = 0;
         colCnt = 0;
        while(!inFile.eof() && rowCnt < numRows){
            int startRow, startCol, color, length;
            inFile >> startRow;
            inFile >> startCol;
            inFile >> color;
            inFile >> length;
            for(int i=0; i<length; i++){
                outFile << color << " ";
                colCnt++;
                if(colCnt%numCols==0){
                    outFile << endl;
                    colCnt = 0;
                    rowCnt++;
                }
            }
        }
    }

    void deCodeMethod4(ifstream& inFile, ofstream& outFile){  // no zero and wrap-
around
      rowCnt = 0;
      colCnt = 0;
      int newLength = 0;
      int startRow, startCol, color, length;
```

```
        inFile >> startRow;
        inFile >> startCol;
        inFile >> color;
        inFile >> length;
      if(startRow != 0 || startCol != 0){
            newLength = startRow*22 + startCol;
            addZero(newLength, outFile);
       }

        while(!inFile.eof() && rowCnt < numRows){
            addNonZero(length, outFile, color);
//              for(int i=0; i<length; i++){
//                  outFile << color << " ";
//                  colCnt++;
//                 // cout << " colCnt:" << colCnt << endl;//
//                  if(colCnt%numCols==0){
//                      outFile << endl;
//                      colCnt = 0;
//                      rowCnt++;
//                  }
//              }
          inFile >> startRow;
          inFile >> startCol;
          inFile >> color;
          inFile >> length;

          if(rowCnt==startRow && colCnt != startCol){
              newLength = (startCol-colCnt);
              addZero(newLength, outFile);
          }
          if(rowCnt!= startRow || colCnt != startCol){
              newLength = (startRow-rowCnt-1)*22 + (22-colCnt) + startCol;
              addZero(newLength, outFile);
          }
       }
         if(rowCnt<numRows && colCnt!=21){
            newLength = (numRows-rowCnt-1)*22 + (22-colCnt);
            addZero(newLength, outFile);
         }
    }

    void addNonZero(int length, ofstream& outFile, int color){
        for(int i = 0; i<length; i++){
            outFile << color << " ";
            colCnt++;
            if(colCnt%numCols==0){
                outFile << endl;
                colCnt = 0;
                rowCnt++;
            }
        }
    }


    void addZero(int length, ofstream& outFile){
        for(int i = 0; i<length; i++){
            outFile << "0" << " ";
            colCnt++;
            if(colCnt%numCols==0){
                outFile << endl;
                colCnt = 0;
```

```
                rowCnt++;
            }
        }
    }

};


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

    ifstream inFile1(argv[1]);//input File:  image1_EncodeMethod1
    string nameEncodeFile = argv[1]; //image1_EncodeMethod1_Decoded
    int numRows, numCols, minVal, maxVal, whichMethod;
    inFile1 >> numRows;
    inFile1 >> numCols;
    inFile1 >> minVal;
    inFile1 >> maxVal;
    inFile1 >> whichMethod;
    string substring = nameEncodeFile.substr(0, 0+nameEncodeFile.size()-4);
    string nameDecodeFile = substring + "_Decoded" + ".txt";
    cout << nameDecodeFile;
    ofstream decodeFile(nameDecodeFile);

    RunLength *run = new RunLength(numRows, numCols, minVal,maxVal );
    decodeFile << numRows << " " << numCols << " " << minVal << " " << maxVal << endl;

    if(whichMethod==1){
        run->deCodeMethod1(inFile1, decodeFile);
    }else if(whichMethod==4){
        run->deCodeMethod4(inFile1, decodeFile);
    }else{
       cout << "Error input!!!!" << endl;
       exit(1);
    }

   inFile1.close();
   decodeFile.close();

    return 0;
}
```

a. image1_EncodeMethod1

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
8 0 0 22
9 0 0 22

b. image1_EncodeMethod1_Decoded

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

c. image1_EncodeMethod4

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

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

d. image1_EncodeMethod4_Decoded

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

e. image2_EncodeMethod1

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

f. image2_EncodeMethod1_Decoded

20 22 0 9
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 4 4 4 4 4 4 4
4 0 4 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 0 0 3 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 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 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 1 1 6 6 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 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

g. image2_EncodeMethod4

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

h. image2_EncodeMethod4_Decoded.

20 22 0 9
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 4 4 4 4 4 4 4
4 0 4 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 0 3 3 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 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 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 1 1 6 6 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 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