```
********************************************************************************
CSCI 381-26                  Project 10: ChainCode                  Language: C++

Name: Akshar Patel

Due date: Soft copy: 4/23/2020
          Hard copy: 4/23/2020

Submission date: Soft copy: 4/23/2020
                 Hard copy: 4/23/2020

********************************************************************************
```

Part 1: Algorithm

```
*******************************
I. main (…)
*******************************
Step 1: labelFile <-- open label file from argv[1]
         propFile <-- open property file from argv[2]
         output image header to ChainCodeFile
         output image header to deBugFile // per text line
         imageAry <-- dynamically allocated
         loadImage (imageAry )
         CCAry <-- dynamically allocated

Step 2: CC <-- get the next connected component from the property file

Step 3: CClabel <-- get the label of CC

Step 4: clearCC () // zero out the old CClabel for next cc

Step 5: loadCC (CClabel, CCAry)
           // Extract the pixels with CClabel from imageAry to CCAry.

Step 6: getChainCode (CC, CCAry) // see algorithm below

Step 7: repeat step 2 to step 5 until all connected components are processed.

Step 8: close all files
```

Part 2: Source code

```
#include <iostream>
#include<fstream>

using namespace std;

class image{
public:
    int numRows, numCols, minVal, maxVal;
    int** imageAry;
    int** CCAry;
    image(int row,int col,int min,int max){
        numRows = row;
        numCols = col;
        minVal = min;
        maxVal = max;
    }
    void loadImage(int** Ary, ifstream& file){
        for(int i = 1 ; i < numRows + 1 ; i++){
            for(int j = 1 ; j < numCols + 1 ; j++){
                file>>Ary[i][j];
```

```
            }
        }
    }
    void set2DZero(int** Ary){
        for(int i = 0 ; i < numRows + 2 ; i++){
            for(int j = 0 ; j < numCols + 2 ; j++){
                Ary[i][j] = 0;
            }
        }
    }
};
class connectCC{
public:
    int numRows, numCols, minVal, maxVal;
    int label, numbpixels;
    connectCC(int row,int col,int min,int max){
        numRows = row;
        numCols = col;
        minVal = min;
        maxVal = max;
    }
    void clearCC(int** Ary){
        for(int i = 1 ; i < numRows + 1 ; i++){
            for(int j = 1 ; j < numCols + 1 ; j++){
                Ary[i][j] = 0;
            }
        }
    }
    void loadCC(int CClabel, int** Ary, int** imgAry){
        for(int i = 1 ; i < numRows + 1 ; i++){
            for(int j = 1 ; j < numCols + 1 ; j++){
                if(imgAry[i][j] == CClabel){
                    Ary[i][j] = CClabel;
                }
            }
        }
    }
    void printImg(ofstream& file, int** Ary){
        for(int r = 1; r < numRows + 1; r++){
            for(int c =1; c < numCols + 1; c++){
                file<<Ary[r][c]<<" ";
            }
            file<<endl;
        }
    }
};
class chainCode{
public:
    class point{
    public:
        int row;
        int col;
    };
    point startP;
    point currentP;
    point nextP;
    point neighborCoord[8];
    int lastQ, nextDir, nextQ, pchaindir;
    int zeroTable[8] = {6, 0, 0, 2, 2, 4, 4, 6};
```

```
    void getChainCode(int CCL, int** & Ary,int  minR,int minC,int maxR,int maxC,
ofstream& defile, ofstream& chainfile){
        int label = CCL,stop;
        int minRow = minR, minCol = minC, maxRow = maxR, maxCol = maxC;
        for(int i = minRow ; i <= maxRow ; i++){
            for(int j = minCol ; j <= maxCol ; j++){
                if(Ary[i][j] == label){
                    defile<<label<<" "<<i<<" "<<j<<endl;
                    chainfile<<label<<" "<<i<<" "<<j<<" ";
                    startP.row = i;
                    startP.col = j;
                    currentP.row = startP.row;
                    currentP.col = startP.col;
                    lastQ = 4;
                    stop = 1;
                    break;
                }
            }
            if(stop == 1)break;
        }
        do{
            nextQ = (lastQ + 1) % 8;
            pchaindir = findNextP(nextQ, Ary, nextP,chainfile, defile);
            chainfile<<pchaindir;
            defile<<pchaindir<<" "<<endl;
            lastQ = zeroTable[pchaindir - 1];
            currentP.row = nextP.row;
            currentP.col = nextP.col;
        }while(currentP.row == startP.row || currentP.col == startP.col);
    }
    int findNextP(int nextQ, int**& Ary, point& nextP,ofstream& cha, ofstream& def){
        int chDir, loop = 0;
        loadNeighborCoord(currentP);
        while(loop < 8){
            switch(nextQ){
                case 0: if(Ary[neighborCoord[0].row][neighborCoord[0].col] > 0)
                            chDir = 0; break;
                case 1: if(Ary[neighborCoord[1].row][neighborCoord[1].col] > 0)
                            chDir = 1; break;
                case 2: if(Ary[neighborCoord[2].row][neighborCoord[2].col] > 0)
                            chDir = 2; break;
                case 3: if(Ary[neighborCoord[3].row][neighborCoord[3].col] > 0)
                            chDir = 3; break;
                case 4: if(Ary[neighborCoord[4].row][neighborCoord[4].col] > 0)
                            chDir = 4; break;
                case 5: if(Ary[neighborCoord[5].row][neighborCoord[5].col] > 0)
                            chDir = 5; break;
                case 6: if(Ary[neighborCoord[6].row][neighborCoord[6].col] > 0)
                            chDir = 6; break;
                case 7: if(Ary[neighborCoord[7].row][neighborCoord[7].col] > 0)
                            chDir = 7; break;
                default:
                    break;
            }
            loop++;
        }
        nextP.row = neighborCoord[chDir].row;
        nextP.col = neighborCoord[chDir].col;
        return chDir;
    }
    void loadNeighborCoord(point& currentP){
```

```
        neighborCoord[0].row = currentP.row;
        neighborCoord[0].col = currentP.col + 1;
        neighborCoord[1].row = currentP.row - 1;
        neighborCoord[1].col = currentP.col + 1;
        neighborCoord[2].row = currentP.row - 1;
        neighborCoord[2].col = currentP.col;
        neighborCoord[3].row = currentP.row - 1;
        neighborCoord[3].col = currentP.col - 1;
        neighborCoord[4].row = currentP.row;
        neighborCoord[4].col = currentP.col - 1;
        neighborCoord[5].row = currentP.row + 1;
        neighborCoord[5].col = currentP.col - 1;
        neighborCoord[6].row = currentP.row + 1;
        neighborCoord[6].col = currentP.col;
        neighborCoord[7].row = currentP.row + 1;
        neighborCoord[7].col = currentP.col + 1;
    }
};
int main(int argc, char** argv){
    string inputName = argv[1];
    ifstream labelFile;
    labelFile.open( inputName );

    string inputName2 = argv[2];
    ifstream propFile;
    propFile.open( inputName2 );

    string outputName1 = argv[3];
    ofstream ChainCodeFile;
    ChainCodeFile.open( outputName1 );

    string outputName2 = argv[4];
    ofstream deBugFile;
    deBugFile.open( outputName2 );

    if(labelFile.is_open() && propFile.is_open()){
        if(ChainCodeFile.is_open() && deBugFile.is_open()){
            int row, col, min, max;
            labelFile>>row>>col>>min>>max;
            image i(row,col,min,max);
            connectCC c(row,col,min,max);
            chainCode ch;
            ChainCodeFile<<row<<" "<<col<<" "<<min<<" "<<max<<endl;
            deBugFile<<row<<" "<<col<<" "<<min<<" "<<max<<endl;
            i.imageAry = new int* [row + 2];
            for( int k = 0; k < row + 2; k++ ){
                i.imageAry[k] = new int[col + 2];
            }
            i.set2DZero(i.imageAry);
            i.loadImage(i.imageAry, labelFile);
            i.CCAry = new int* [row + 2];
            for( int k = 0; k < row + 2; k++ ){
                i.CCAry[k] = new int[col + 2];
            }
            i.set2DZero(i.CCAry);
            propFile.ignore(256,'\n');
            int cc;
            propFile>>cc;
            for(int k = 1 ; k <= cc; k++){
                int CClabel, minR, minC, maxR, maxC;
                propFile>>CClabel;
```

```
                propFile.ignore(256,'\n');
                propFile.ignore(256,'\n');
                propFile>>minR>>minC>>maxR>>maxC;
                c.clearCC(i.CCAry);
                c.loadCC(CClabel, i.CCAry, i.imageAry);
                ChainCodeFile<<"chain code for cc: "<<k<<endl;
                ch.getChainCode(CClabel, i.CCAry, minR, minC, maxR, maxC, deBugFile,
ChainCodeFile);
                ChainCodeFile<<endl;
            }
            labelFile.close();
            propFile.close();
            ChainCodeFile.close();
            deBugFile.close();
        }else{cout<<"Error!! Could NOT create output file"<<endl;}
    }else{cout<<"Error!! Could NOT open input file"<<endl;}
}
```

Part 3: Output

```
- ChainCodeFile for chainCodeImg1
     20 31 0 1
     chain code for cc: 1
     1 3 15 5

- deBugFile for chainCodeImg1
     20 31 0 1
     1 3 15
     5

- ChainCodeFile for chainCodeImg2
     20 40 0 3
     chain code for cc: 1
     1 3 8 5
     chain code for cc: 2
     2 3 30 5
     chain code for cc: 3
     3 13 24 12

- deBugFile for chainCodeImg2
     20 40 0 3
     1 3 8
     5
     2 3 30
     5
     3 13 24
     12
```

This is not correct I tried my best to solve but I guess my while loop is in correct when I do currentP != startP loop goes to infinite. So it shows only first next point.