Project 6 (C++): You are to implement both 4-connected and 8-connected component algorithms in this project.

*** You will be given two data files: data1 and data2 .
What do you need to do as follows:
a) Implement your program based on the specs below.
b) Run your program twice; first using 4 and then using 8.

Your hard copies include:

- Cover page
- Source code
- RFprettyPrintFile for 4-connectness for data1
- labelFile for 4-connectness for data1
- propertyFile for 4-connectness for data1
- RFprettyPrintFile for 4-connectness for data2
- labelFile for 4-connectness for data2
- propertyFile for 4-connectness for data2
- RFprettyPrintFile for 8-connectness for data1
- labelFile for 8-connectness for data1
- propertyFile for 8-connectness for data1
- RFprettyPrintFile for 8-connectness for data2
- labelFile for 8-connectness for data2
- propertyFile for 8-connectness for data2

****
Language: C++
Project points: 10pts
Due Date: Soft copy (*.zip) and hard copies (*.pdf):

+1 (11/10 pts): early submission, 4/5/2022 Tuesday before midnight.
-0 (10/10 pts): on time, 4/8/2022 Friday before midnight.
-1 (9/10 pts): 1 day late, 4/9/2022 Saturday before midnight.
-2 (8/10 pts): 2 days late, 4/10/2022 Sunday before midnight.
(-10/10 pts): none submission, 4/10/2022 Sunday after midnight

*** Name your soft copy and hard copy files using the naming convention as given in the project submission requirement.
*** All on-line submission MUST include Soft copy (*.zip) and hard copy (*.pdf) in **the same email attachments** with correct email subject as stated in the email requirement; otherwise, your submission will be rejected.
****
I. Inputs (argv[1]):
a) A binary image.
b) Connectness: from argv[2]
II. Outputs:
a) RFprettyPrintFile (argv[3]): (include in your hard copy) for the followings:
** a proper caption means the caption should say what the printing is.

- reformatPrettyPrint of the result of the Pass-1 with proper captions
- print newLabel and the EQAry after Pass-1, with proper captions
- reformatPrettyPrint of the result of the Pass-2 with proper captions
- print newLabel and the EQAry after Pass-2, with proper captions
- Print the EQAry after manage the EQAry, with proper caption
- reformatPrettyPrint of the result of the Pass-3 with proper captions
- reformatPrettyPrint of the result bounding boxes drawing.

b) labelFile (argv[4]): to store the result of Pass-3 -- the labelled image file
with image header, numRows numCols newMin NewMax. ** This file to be used in future processing.

c) propertyFile (argv[5]): ** (include in your hard copy)
To store the connected component properties.
The format is to be as below:

- 1st text-line, the header of the input image,
- 2nd text-line is the total number of connected components.
- from 3rd text, use four (4) text-lines per each connected component:
- label
- number of pixels
- upperLftR upperLftC //the r c coordinated of the upper left corner
- lowerRgtR lowerRgtC //the r c coordinated of lower right corner

For an example:

```
45  40  0   9  // image header
9                    // there are a total of 9 CCs in the image
1                    // CC label 1
187         // 187 pixels in CC label 1
4    9   // upper left corner of the bounding box at row 4 column 9
35  39  // lower right corner of the bounding box at row 35 column 39
:                    :
** This file to be used in future processing.
```

****
III. Data structure:
****

- A CClabel class
    - (int) numRows
    - (int) numCols
    - (int) minVal
    - (int) maxVal
    - (int) newMin
    - (int) newMax
    - (int) newLabel // initialize to 0
    - (int) trueNumCC // the true number of connected components in the image
      // It will be determined in manageEQAry method.
    - (int) zeroFramedAry[][] // a 2D array, need to dynamically allocate
      //at run time of size numRows + 2 by numCols + 2.
    - (int) NonZeroNeighborAry [5] // 5 is the max number of neighbors you have to check.
      // For easy programming, you may consider using this 1-D array
      // to store pixel(i, j)'s non-zero neighbors during pass 1 and pass2.
    - (int) EQAry [] // an 1-D array, of size (numRows * numCols) / 4
      // dynamically allocate at run time, and initialize to its index, i.e., EQAry[i] = i.
    - Property (1D struct or class)
        - (int) label   // The component label
        - (int) numpixels // total number of pixels in the cc.
        - (int) minR // with respect to the input image.
        - (int) minC // with respect to the input image.
        - (int) maxR // with respect to the input image.
        - (int) maxC // with respect to the input image.

        // In the Cartesian coordinate system, any rectangular box can be represented by two points: upper-left corner and the lower-right of the box. Here, the two points:(minR minC) and(maxR maxC) represents the smallest rectangular box that the cc can fit in the box; object pixels can be on the border of the box.

    - (Property) CCproperty []

        // A struct 1D array for storing all components' properties.

        // The size of array is the actual number of cc after manageEQAry

- methods:
    - constructor(...) // need to dynamically allocate all arrays; and assign values to numRows,, etc.
    - zero2D (...) // ** Initialized a 2-D array to zero. You must implement this method, don't count on Java.

```
- minus1D (...) // ** Initialized a 1-D array to -1.
- loadImage (...)
        // read from input file and write to zeroFramedAry begin at(1,1)
- imgReformat (zeroFramedAry, RFprettyPrintFile) // Print zeroFramedAry to RFprettyPrintFile
- connect8Pass1 (...) // On your own, as taught in class and algorithm is in lecture note
- connect8Pass2 (...) // On your own, as taught in class and algorithm is in lecture note
- connect4Pass1 (...) // On your own, as taught in class and in lecture note
- connect4Pass2 (...) // On your own, as taught in class and in lecture note
- connectPass3 (...) // On your own. There is no differences between 4-connectness and 8-connectness.
- drawBoxes (...) // Draw the bounding boxes on all connected components in zeroFramedAry.
                  // See algorithm below
- updateEQ (...) // Update EQAry for all non-zero neighbors to minLabel, it will be easier to use
                  //NonZeroNeighborAry to store all non-zero neighbors.
- (int) manageEQAry (...) // The algorithm was taught in class and in lecture note.
                          // The method returns the true number of CCs in the labelled image.
- printCCproperty (...) // Prints the component properties to propertyFile using the format given in the above.
                  // On your own.
- printEQAry (...) // Print EQAry with index up to newLabel, not beyond. On your own
- printImg (...) // Output image header and zeroFramedAry (inside of framing) to labelFile
                  // on your own.
```

****

IV. main(...)

****

```
step 0: inFile ← open the input file
        RFprettyPrintFile , labelFile, propertyFile ← open from args[]
         numRows, numCols, minVal, maxVal ← read from inFile
        dynamically allocate zeroFramedAry.
        newLabel ← 0
step 1: zero2D (zeroFramedAry)
step 2: loadImage (inFile, zeroFramedAry)
step 3: Connectness ← argv[2]
step 4: if connectness == 4
            connect4Pass1 (… )
            imgReformat (zeroFramedAry, RFprettyPrintFile)
            printEQAry (newLabel, RFprettyPrintFile)
                // print the EQAry up to newLable with proper caption
            Connect4Pass2 (…)
            imgReformat (zeroFramedAry, RFprettyPrintFile)
            printEQAry (newLabel, RFprettyPrintFile)
                // print the EQAry up to newLabel with proper caption

step 5: if connectness == 8
            connect8Pass1 (… )
            imgReformat (zeroFramedAry, RFprettyPrintFile)
            printEQAry (newLabel, RFprettyPrintFile)
                // print the EQAry up to newLabel with proper caption
            Connect8Pass2 (…)
            imgReformat (zeroFramedAry, RFprettyPrintFile)
            printEQAry (newLabel, RFprettyPrintFile)
                // print the EQAry up to newLabel with proper caption
step 6: trueNumCC ← manageEQAry (EQAry, newLabel)
            printEQAry (newLabel, RFprettyPrintFile)
            // print the EQAry up to newLabel with proper caption
step 7: connectPass3 (...)
step 8: imgReformat (zeroFramedAry, RFprettyPrintFile)
step 9: printEQAry (newLable, RFprettyPrintFile)
```

```
                    // print the EQAry up to newLabel with proper caption
step 10: output numRows, numCols, newMin, newMax to labelFile
step 11: printImg (labelFile) // Output the result of pass3 inside of zeroFramedAry
step 12: printCCproperty (propertyFile) // print cc properties to propertyFile
step 13: drawBoxes(zeroFramedAry, CCproperty) // draw on zeroFramed image.
step 14: imgReformat (zeroFramedAry, RFprettyPrintFile)
step 15: print trueNumCC to RFprettyPrintFile with proper caption
step 16: close all files
```

****

VI. drawBoxes (zeroFramedAry, CCproperty)

****

```
// This method may contain bugs, report bugs to Dr. Phillips

step 1: index ← 1

step 2:   minRow ← CCproperty[index]'s minR  + 1
          minCol ← CCproperty[index]'s minC + 1
          maxRow ← CCproperty[index]'s maxR + 1
          maxCol ← CCproperty[index]'s maxC + 1
          label ← CCproperty[index]'s label

step 3: Assign all pixels on minRow from minCol to maxCol ← label
          Assign all pixels on maxRow from minCol to maxCol ← label
          Assign all pixels on minCol from minRow to maxRow ← label
          Assign all pixels on maxCol from minRow to maxRow ← label

step 4: index++
step 5: repeat step 2 to step 4 while index <= the actual number of cc
```