School of Computing
National University of Singapore
CS2010 Data Structures and Algorithms 2
Semester 2, AY 2015/16

Assignment 1 – Connected Component Analysis
Assigned – Jan 19th, 2016
Due – Jan 29th (Friday by 11.59pm), 2016 (in lab assessment Feb 1st, 2016)

Objective:

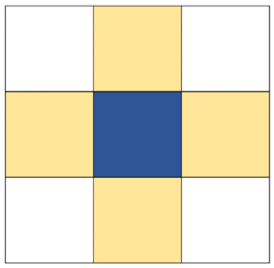
- To "re-familiarize" yourself with Java programming
- To convert a high-level algorithmic description to working Java code.
- To use basic Java ADTs, in this case a Queue

Preparation:

Download the file assignment1.zip from IVLE into your working directory. Uncompress the file and you should find the following document, java source code (A1ConnectedComponent.java, ImageUtil.java, ConnectedComponentAlgorithm.java) and 10 image files: 1.png, 2.png, 3.png, 4.png, 5.png, 6.png, 7.png, 8.png, 9.png, and 10.png.

Connected Components Algorithm:

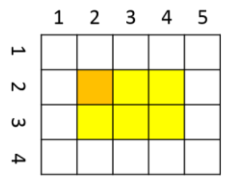
This algorithm finds all connected components in a binary image. Any binary pixels that are connected make up the same component. We will considered pixels connected their horizontal and vertical neighbors are touching. This is called "4-connected" pixels. For example, the 4-connected pixels of the blue below are the four yellow points.



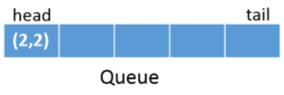
This algorithm to the find the N4-connected regions in a binary image. The description of the algorithm is as follows:

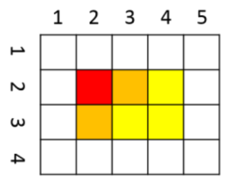
```
    Input: Image A (binary – 0 = background, 255 = foreground)
    Output: Image B (each component colored), # of components
1.  Initialize an "Visited Image" V as all zeros, Initialize components ID=0
2.  Initialize an empty Q
3.  Scan the image from top to bottom, row by row
4.  When a foreground pixel in A that has not been already been visited in V
5.  Increment component ID
6.  Insert the pixel coordinate, u = (x, y) ,  into an empty Q
7.  Mark image V(u) as being visited (i.e. mark it as component ID)
8.  while the Q is not empty
9.          Remove u out of the queue and visit u.
10.         for all pixels, v, connected to this pixel, u
11.         if v has not been visited in V
12.                 Mark V(v) using the component ID
13.                 Insert pixel v into the queue.
14.         end for
15. end while  // now all connected pixels will have the same component ID in V
16. end Scan Image
```

Here is a simple example how the algorithm is working. In this example, yellow pixels are foreground pixels; orange pixels are those have been visited and are in the queue; red pixels are those have been visited and were in the queue but now removed; white pixels are background:

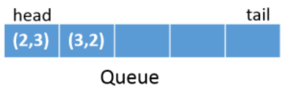


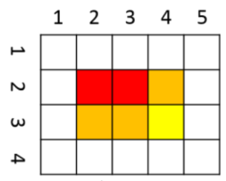
Step5-6: place (2,2) in the queue



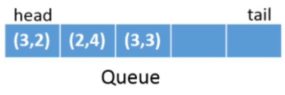


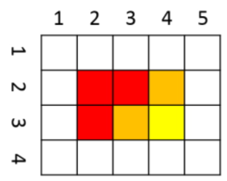
Step8-13 1st iteration: take (2,2) out of the queue and insert (2,3), (3,2) in the queue



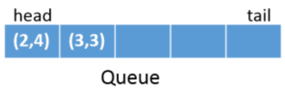


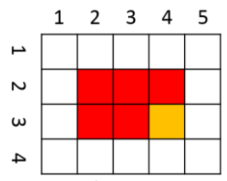
Step8-13 2nd iteration: take (2,3) out of queue and place (2,4), (3,3) in the queue



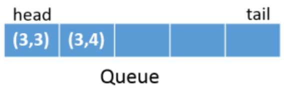


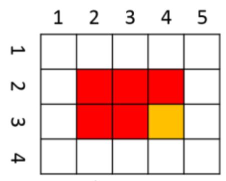
Step8-13 3rd iteration: take (3,2) out of the queue



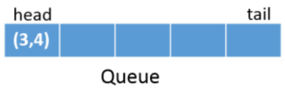


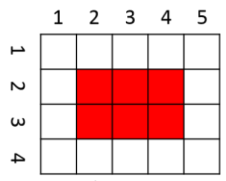
Step8-13 4th iteration: take (2,4) out of the queue and insert (3,4) in the queue



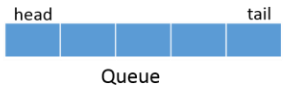


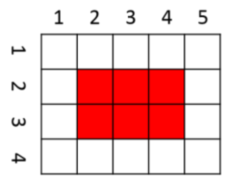
Step8-13 5th iteration: take (3,3) out of queue



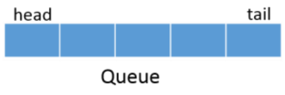


Step8-13 6th iteration: take (3,4) out of queue





Step14: the queue is empty, finish visiting a connected component



Your Task:

You need to implement the connected components algorithm described on the previous page. To visualize the result, each component should be assigned different colors, Fig 1 and Fig 2, show examples in the input and output. You are recommended to follow the procedure below to finish this assignment:

- Step1: to read picture from local file, *ImageUtil.readBinaryImage* should be used.
- Step2: find all connected components by the algorithm described above.
- Step3: map different components to different colors, *ImageUtil.mapColor* should be used.
- Step4: save the color picture you got from step3 to a file.

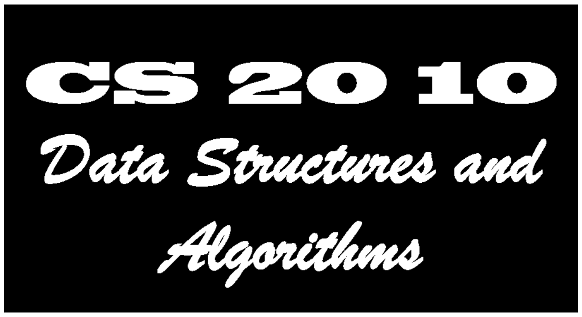


Fig1. Input image.

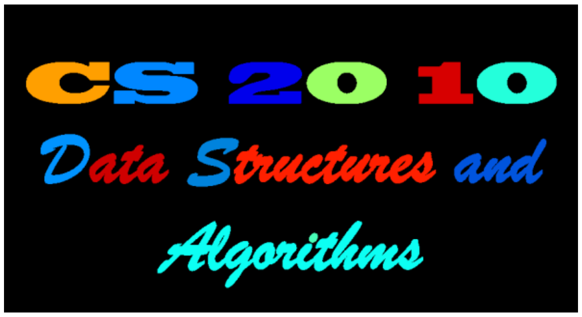


Fig2. Algorithms result.

Code for step1, step3 and step4 are given to you and you only need to finish step2. Please read *Assignment1ConnectedComponent.java* for details.

You need to run your codes to do connected components analysis on all the 10 test pictures (1.png, 2.png, 3.png, 4.png, 5.png, 6.png, 7.png, 8.png, 9.png, 10.png). These are placed in a directory called “images”.

【Input】A black(0) and white(255) image in png format. Black pixels are background and white pixels are foreground. White pixels are regions that you need to search connected component. Fig1 is an example. There are 10 images in total. Try our code on all ten images.

【Output】The number of components and an RGB image with assigned colors indicating each component. The name of the result image is <test_image_id>_<number_of_components>.png (e.g. there are 13 components in 1.png. Then your result image name should be *1_13.png*). Fig2 is the example of result of Fig1.

【Implementing】Please work from the source code provided.

【Hint】Java provides both Queue and 2D Point classes.

Submission Instruction

Submit the following:

1. Submit your code on IVLE before 11.59pm Jan 29 to be considered for full marks.
2. Please put your Java codes in a folder with the image files and submit the entire folder zipped. Use the following convention to name your zipped folder: MatriculationNumber_yourName_Assignment1. For example, if your matriculation number is A1234567B, and your name is Chow Yuen Fatt, for this assignment, your file name should be A1234567B_ChowYuenFatt_Assignment1.zip.
3. It is up to you to test to make sure that your Java code in the zipped file is complete. It is recommended that you unzip your Java code in a different location and test it to make sure it runs.

Assessment (Feb 1st, 2016 – In Lab)

- You will be assessed individually in the lab session (you must attend your assigned lab!)
- You'll have approximately 4-5 minute Q&A session with Dr. Brown or one of the TAs (Hakki or Sixing).
- We will run the code directly from your submitted IVLE submission, you won't be able to bring an updated version to the lab.
- We will run your code and then ask you anything about your code we want. Be prepared to answer questions. (I forget is not a good answer!).