Project Module 2

Due date: 07 December 2022 2359 Hrs

Instructions

- Any attempt to cheat in the project will lead to ZERO marks in all the project of the course.
- **Start your work early!** You need time to understand the project and to answer the many questions that will arise as you read the provided description.

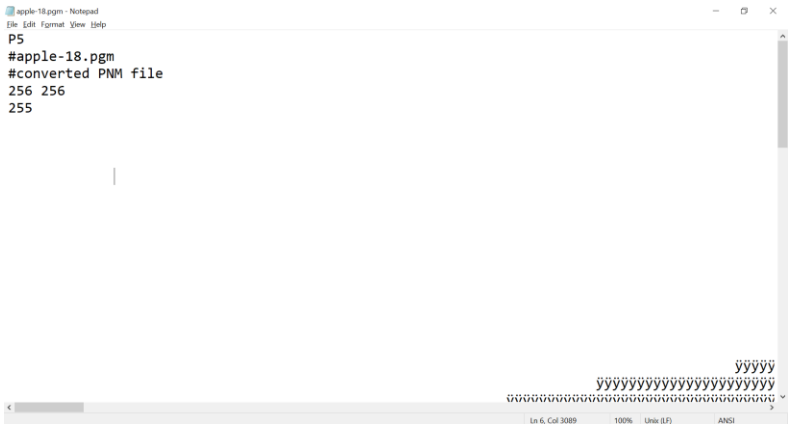
Submission

- Write a report detailing your work of 5-7 pages explaining your results and input-output relationship. Provide a brief introduction to the problem in your own words, then describe your approach (solution) to the problem.
- **Mention in your report some of the challenges and programming errors and mistakes you encountered while making your solution.**
- You can include figures and images in your report. Show the output of your code for multiple inputs in your report.
- What to submit:
    - Submit the report
    - Code
    - supported images used in the project
    - a readme file, detailing how to run the submitted code
    - and the log files. Description of log files is at the end of this project.
    - Zip all these into a single file and name it DSA_Section_Name_Reg.No_Module_1
    - Replace section, name and reg. no with your own info.
- Do not wait for the deadline, start your work immediately, the project may take longer than you initially think.
- Try to cover all the aspects mentioned in this document.

Coding guidelines

- The images provided to you are binary images (0, 255) shade values only.
- The goal of this project is to familiarize you with the arrays and queue, stack, link list, trees, and hashing data structures. You will write code to apply basic image processing operations on the images.
- Display a menu (console menu) to show all the possible options present in the app which the user can select.
- Your code must be implemented using templates. Do not write type specific codes, create all the data structures thru templates.

Data Preparation

- Download and install the filestar software thru the link: https://filestar.com/skills/gif/convert-gif-to-pgm
- Now convert your gif files (provided with the project) to the pgm format using the Filestar/other software. You can open pgm files in the notepad to see the contents.
- The pgm file may look like,
- 

- Other than the comments with # the file shows 256 256 in the above screenshot of the pgm file denotes that the file size is 256x256
- The binary images have only two possible pixel values, 0 or 255. The pixel is either On/Off. The presence of the pixel value can be assumed to be a 255, and the absence of the pixel value indicates a 0 pixel value.

Image Class formation

- First, create an ADT class to store the image. Inside the class the image is represented as an array of pixel values.
- Implement a 'Read' method which takes as a parameter the image file path and read the image and store it in the array. Moreover, it will also update the image width and height property recorded in the relevant fields of the class.
- Implement a 'Save' method which writes the image in the PGM format.
- 'GetPixel' takes the row and column no. of the image and returns its pixel value. It must handle the relevant exception if anyone accesses a pixel outside of the image size.
- 'SetPixel' updates the value of a specific pixel in the image, it must also handle the out of bound exception
- 'GetSize' method will return the size of the image as an object.
- 'ConvertToNegative': this method will invert the values of the binary image, 0 to 255 and vice versa.

Q1:

Write C++ code to compute basic statistics about the image, i.e.

a) Mean pixel value by adding all the pixel values and dividing it by the no. of pixels.
b) The number of black and no. of white pixels.

c) The average no. of black pixels in each row.
d) Convert image to its negative thru the image class method.

Q2)

Connected component extraction from an image region. To know more about connected component, read the details at:

- https://homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm
- https://en.wikipedia.org/wiki/Connected-component_labeling
- https://www.mathworks.com/help/images/label-and-measure-objects-in-a-binary-image.html

The user will provide the pixel coordinates, your job is to extract the connected component and write it on the disk as an independent binary image. The file name must be appended with a timestamp to avoid overwriting the file. Use time library to get system time .

In order to extract the connected component you will need stack and queue.

Each pixel has 8-neighbors, the neighbor of a pixel that has the same value that of the pixel is part of the connected component. Given a pixel as a starting point, extract all the pixels of the connected component.

To start the connected component extraction Enqueue the starting pixel provided by the user.

Make an empty output image by setting all the values to 255.

Then unless the Queue is empty repeat:

Step 1: Dequeue a pixel location and set this location in the output image to 0.

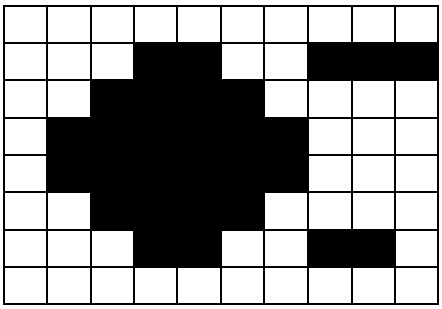
Step 2: Then check for the neighboring pixels in the input image an enqueue all the connected pixels.

Q3:

Repeat the process implemented in the Q2, but use the Stack data structure in place of the Queue.

Q4:

In this part of the project you will use link list to apply run-length coding to the image. The run-length coding (RLC) enables compact encoding of the image and reduce the image's transmission bandwidth requirement.



The above image when represented using the RLC scheme looks like the below given data.

8 10
-1
4 5 8 10 -1
3 6 -1
2 7 -1
2 7 -1
3 6 -1
4 5 8 9 -1
-1

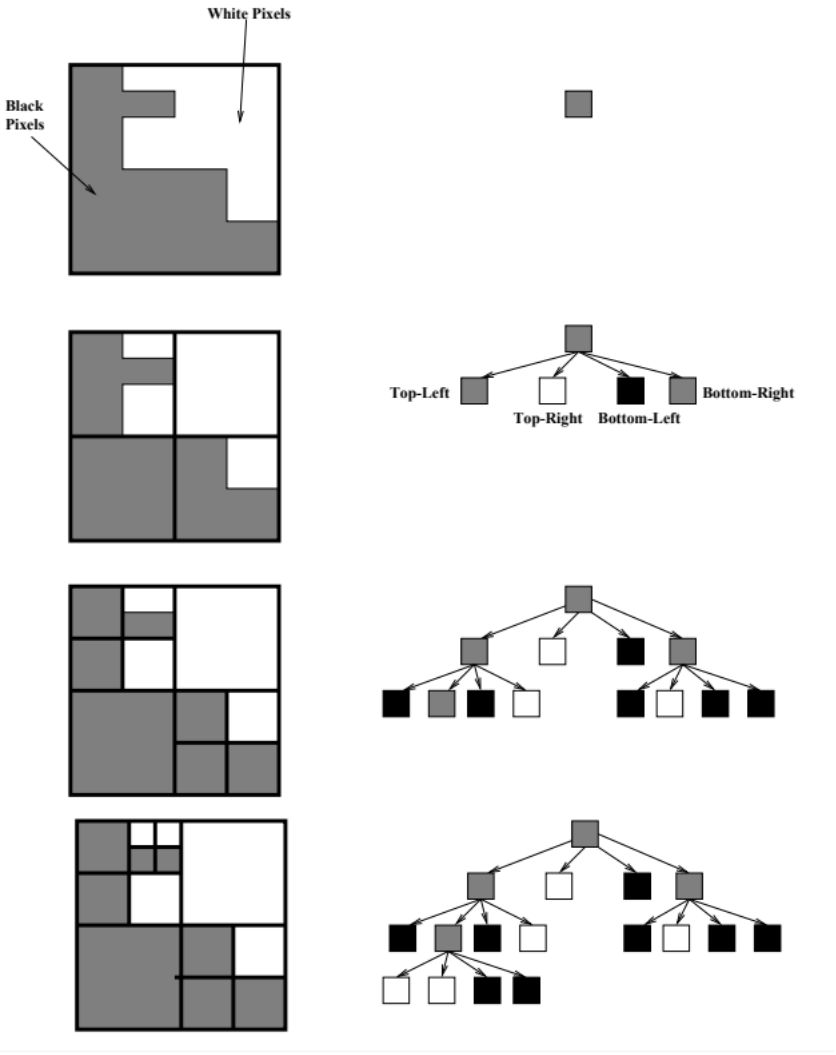
Here the first line shows image size, i.e. 8x10, 8 rows and 10 columns then every row is represented using the numbers. In each row the set of black pixels are marked with cell numbers, like for the 2nd row the black pixels are from column 4 to 5 and 8 to 10 and for the 3rd row, the black pixels span from 3 to 6 column. Similarly, in the 7th row the pixels 4 to 5 and 8 to 9 are black. The -1 is used to show the end of line. If a line contain more than 1 segment of black pixels it is represented accordingly as shown in the row no. 2 and 7.

Write code to perform the following operations:

a. Read a pgm file and convert it to the RLC format and write RLC file to the disk
b. Read an RLC file and convert it back to image, which the user can write back to the disk.
c. From the RLC format compute the total number of black pixels in the image.
d. Perform the image negative operation (white pixels to black and black to white)

You will use link list data structure for this purpose.

Q5: Image files can be easily represented using the quad trees data structure. Each node of the trees has 3 color values, i.e. white, black or gray. In the first row, the root node represent the whole image, as it has a mix of black and white pixels. The root node has a gray color. In the next level we divide the image into 4 regions and image using the notation as presented in the figure below. Every gray node has 4 children representing its 4 sub-regions. The left nodes can have either black or white color.

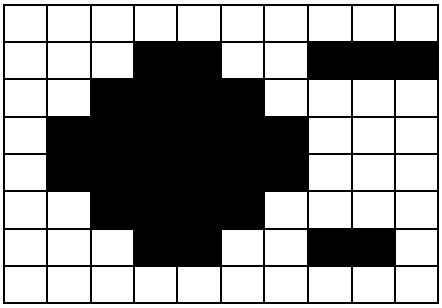


A. Write code to convert a PGM image into a Quad tree structure
B. Convert the image from quad tree to a PGM file.
C. Convert the image into negative using quad tree and write back the pgm to the disk.
D. Print the pre-order, and post-order traversal of the tree and write it to the disk.

Q6:

In this part of the project you will design a simple shape recognition procedure. The for the problem are black and white images. First we will store the images in the database, the database is a hash table.

The procedure will be to first compute horizontal and vertical projection profiles of the binary images. The horizontal projection is computed by computing the sum of black pixels in each row and noting the count in a 1D array. Similarly, count the no. of black pixels in each column and store it in a 1D array we call it the vertical projection profile. An example computation of horizontal and vertical projections is given below:



Vertical projection:

0	2	4	6	6	4	2	2	2	1

Horizontal projection:

0
5
4
6
6
4
4
0

The next step is to combine both the horizontal and vertical projection and with the four digit folding method compute the number for hash processing.

For example the above computed projections are combined together:

0	2	4	6	6	4	2	2	2	1	0	5	4	6	6	4	4	0

Then create groups of 4 digits and take the sum:

0246 + 6422 + 2105 +4664 + 4000

Sum is: 17437

Now take the mod with the hash table size , for example, I am going to take 401 as the hash table size. Then compute the mod, 17437%401 = 194

Place this image name at 194 in the hash table.

You have to read a database of images and using the above-mentioned method and store image names in at the hash at the key locations computed using the above method. The above's image's name will be stored at the index 194 in the hash table.

You may use any appropriate method for collision resolution. Keep in mind to use the same method while searching back the image in the hash table.

While searching for the image you may use the same procedure of computing the horizontal and vertical projections, then four digit folding method followed by mod with the has table size. Then match the file name with the query image.

The database of images will be provided to you later.

Make 1 single project and display the menu options

Q1: press 1 to run solution of Q1

Q2: press 2 to run the solution of Q2

…

Q6: press 6 to run the solution of Q6, then display appropriate sub-menu to choose options between A to D.

Then proceed with appropriate interaction with the user. Your program must never crash because of the incorrect input from the user. Handle checks on all the inputs and manage exceptions accordingly. Before closing the program write a log file which summarize actions performed in this run of the program. Then name it like log_202210292200. The number at the end of the file name is date and 24 Hr time stamp.

- Please note that these log files will be generated by your program. So write code to generate these files.
- Submit these log files with your project.
- You can convert the pgm files generated thru your code back to image format using the Filestar/other software.
- If you have any confusion in understanding the project, you can ask question at the google classroom thread.

Good Luck 😊