Gebze Technical University
Computer Engineering

CSE 222 - 2019 Spring

HOMEWORK 5 REPORT

Sıla Bengisu Yüksekli
1801042877

Course Assistant: Doç.Dr. Erchan Aptoula

1 INTRODUCTION

1.1 Problem Definition

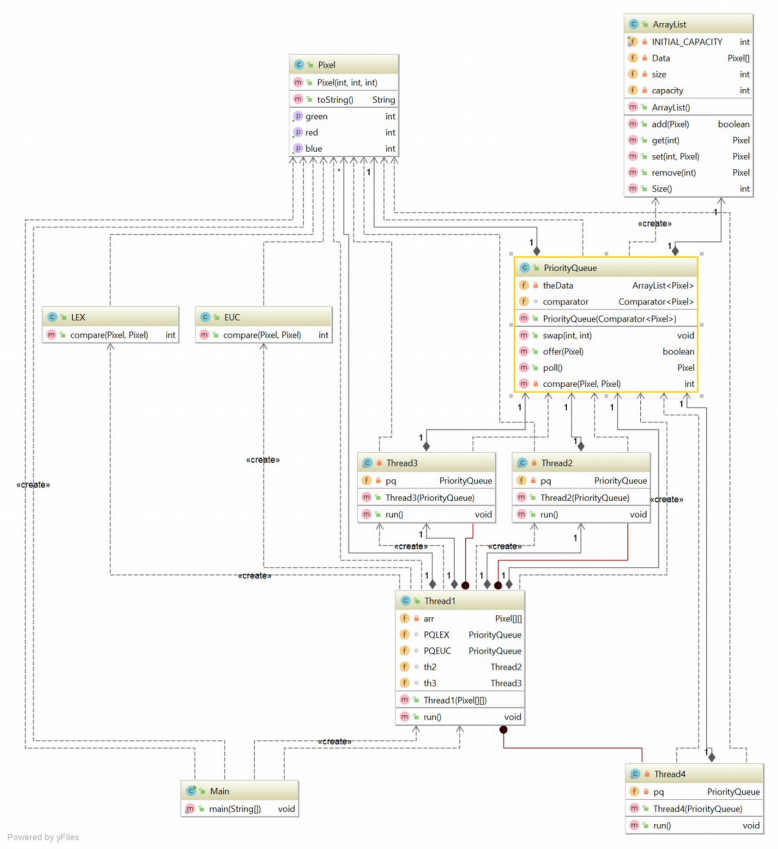
In this problem, some image is given and this image has unknown number of pixels. Pixels have three integer values that represents the colors. First integer represents red, the second one is green and the third one blue.
We have to read this pixels one by one by the help of thread according to priority of pixels and then you have to print pixel values. In this problem, there are three types of priority: Lex , Euc , Bmx.. For example according to the Euc priority we have to look the root of the sum of the square of colors and According to the Lex piority, it compares the the pixels according to alphabetical order of their component letters.

1.2 System Requirements

You can run this program everywhere which has java virtual machine. But if you want to use IntelliJ like me, maybe you need some requirements like minimum 1 GB RAM, 300 MB hard disk space, minimum 1 GB for caches, 1024x768 minimum screen resolutionSystem requirements.

2.METHOD

2.1 Class Diagrams



2.2 Use Case Diagrams.

The user can open the file in intellij and run from the green button which is located at right-up edge, but if user want to run in some other place, if there is java virtual machine, he/she can do.

2.3 Problem Solution Approach

Firstly I read the problem definition repeatedly. Because it was a complicated problem. In this problem the user gaves an image to program and program reads the pixels of this image and print them on the screen one by one. After I had understood the problem, Firstly I decided to learn how to load an image in java, I made a little research about image loading and tried to load example.png.

After that I started to implement my own PriorityQueue as a data structure. I kept an ArrayList as a data field, because when you dont have enough capacity, reallocation is more easier than array. That's why insertion is more realible when you work on a project something big. So I wrote my own ArrayList to use in PriorityQueue.

Due to the pixels that I have to read from the given image, I wrote a Pixel class that represents the colors of one pixel.

In this problem we have to use PriorityQueus as a data structure, because I had to read this pixels and insert them in the queue according to their priorty and this priorities was partioned into three parts. First one is EUC, second one is LEX, and the last one is BMX. Because of that I started to researched about them and tried to implement them.

3 RESULT

3.1 Test Cases

I tested my program in main repeatedly by using some arbitary Pixel objects to understand the correctness of my PriorityQueue. After the compilition, I look the results and checked on the paper, then I compare with the results that was found by computer. Then I tested loading of image. I printed on the screen information about the loading. So when I compiled my program I could test whether the program is working. I made Thread in main and tested it. By this methods, I decided to my priority queue and loading image and thread1 is successful.

3.2 Running Results

```
10  public class Main {
11
12      public static void main(String[] args) {
13
14          Pixel p1 = new Pixel(1, 1, 1);
15          Pixel p2 = new Pixel(1, 1, 1);
16          Pixel p3 = new Pixel(1, 2, 3);
17          Pixel p4 = new Pixel(2, 3, 4);
18          Pixel p5 = new Pixel(2, 4, 5);
19          Pixel p6 = new Pixel(2, 4, 6);
20          Pixel p7 = new Pixel(2, 4, 7);
21          Pixel p8 = new Pixel(2, 4, 8);
22          Pixel p9 = new Pixel(2, 4, 7);
23
24          PriorityQueue pq = new PriorityQueue(new EUC());
25
26          pq.offer(p1);
27          pq.offer(p2);
28          pq.offer(p3);
29          pq.offer(p4);
30          pq.offer(p5);
31          pq.offer(p6);
32          pq.offer(p7);
33          pq.offer(p8);
34          pq.offer(p9);
35
36          for (int i = 0; i < pq.theData.Size(); ++i) {
37              System.out.println(pq.theData.get(i));
38
39          }
```

Test case1 – testing Euc comparator and priorityQueue

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=65150:C:\Program Files\JetBrains
2 4 8
2 4 7
2 4 6
2 4 7
1 2 3
1 1 1
2 4 5
1 1 1
2 3 4
```

Result

```java
public class Main {

    public static void main(String[] args) {

        Pixel p1 = new Pixel(1, 1, 1);
        Pixel p2 = new Pixel(1, 1, 1);
        Pixel p3 = new Pixel(4, 2, 3);
        Pixel p4 = new Pixel(2, 3, 4);
        Pixel p5 = new Pixel(2, 4, 5);
        Pixel p6 = new Pixel(2, 4, 6);
        Pixel p7 = new Pixel(2, 4, 7);
        Pixel p8 = new Pixel(2, 4, 8);
        Pixel p9 = new Pixel(2, 4, 7);

        PriorityQueue pq = new PriorityQueue(new LEX());

        pq.offer(p1);
        pq.offer(p2);
        pq.offer(p3);
        pq.offer(p4);
        pq.offer(p5);
        pq.offer(p6);
        pq.offer(p7);
        pq.offer(p8);
        pq.offer(p9);

        for (int k = 0; k < pq.theData.Size(); ++k) {
            System.out.println(pq.theData.get(k));
```

Test case2 – testing Lex comparator and priorityQueue

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=49265:C:\Program Files\JetBrains
4 2 3
2 4 8
2 4 7
2 4 7
2 3 4
1 1 1
2 4 6
1 1 1
2 4 5
```

Result

```java
BufferedImage image = null;

try {

    image = ImageIO.read(new File( pathname: "example.png"));
    System.out.println("Loading is successful");
}

catch (Exception e) {
    System.out.println(e.getMessage());
}
```

Test case3 – Testing image loading

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=49198:C:\Program Files\JetBrains\
Loading is successful

Process finished with exit code 0
```

Result

```
Thread1 th1= new Thread1(pixel);
th1.run();
```

Test case4 – Testing iworking of thread

```
Thread 1:255 255 105
Thread 1:255 255 105
Thread 1:255 255 105
Thread 1:255 255 105
Thread 1:255 255 105
Thread 1:255 255 105
Thread 1:255 255 105

Process finished with exit code 0
```

Result