Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 5 REPORT

NURULLAH GÜNDOĞDU
161044108

ÖZGÜ GÖKSU

1 INTRODUCTION

1.1 Problem Definition

A png or jpg file is imported as input. It separates the colors inside the file and recognizes a 2-dimensional arr. Then, the values in this figure are added to 3 different heap. These heaps are sorted by different comparison methods. Then the thread is processed using the given thread operations.

1.2 System Requirements

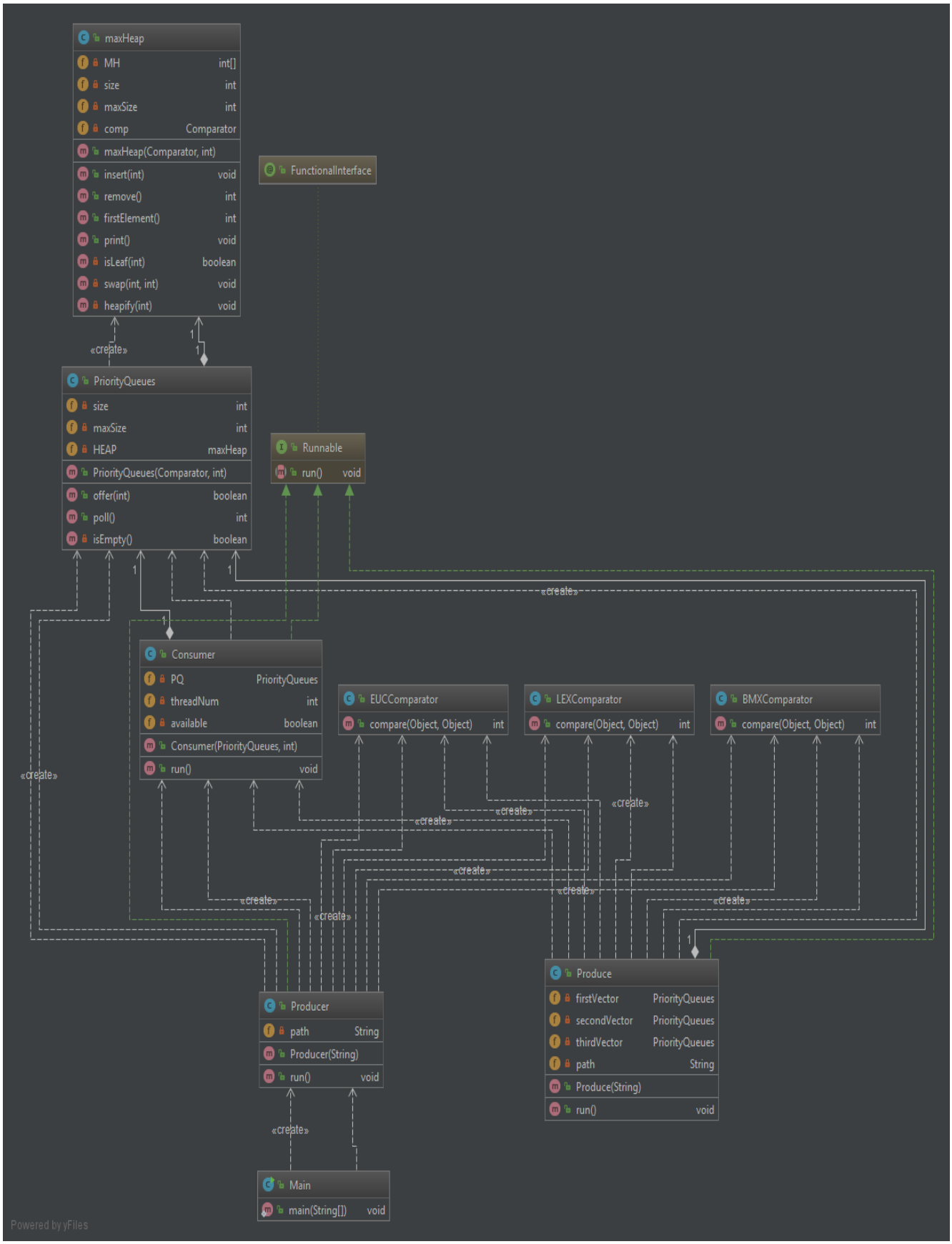
You have to download İntellij IDEA for run the programs. The program requirements are listed below.

Microsoft Windows 10/8/7/Vista/2003/XP (incl.64-bit)
2 GB RAM minimum, 4 GB RAM recommended
1.5 GB hard disk space + at least 1 GB for caches
1024x768 minimum screen resolution

You can download program here;
https://www.jetbrains.com/idea/download/#section=windows (Community Version)

2 METHOD

2.1 Class Diagrams



2.2 Problem Solution Approach

```
public int poll()
```

This function performs the operation of removing the element from heap.
T (logn) = O (logn(n)).

```
public boolean offer(int element)
```

This function makes the process of adding element to heap.
T (logn) = O (log(n)).

```
public int compare(Object o1, Object o2)
```

This function compares the incoming values with the BMX method. The BMX method converts the red, blue and green values of binary values into binary numbers. It then creates a 24-bit binary number by combining these binary numbers. Compares the 2 numbers resulting from this operation.
T (1) = O (1).

```
public int compare(Object o1, Object o2)
```

This function compares the incoming values with the EUC method. The EUC method calculates the square root of the sum of the squares of red, blue And green values of the incoming values. and compares the resulting values.
T (1) = O (1).

```
public int compare(Object o1, Object o2)
```

This function compares the incoming values with the LEX method. The LEX method compares the red, green, and blue tones in the corresponding values.

T (1) = O (1).

```
public void run()
```

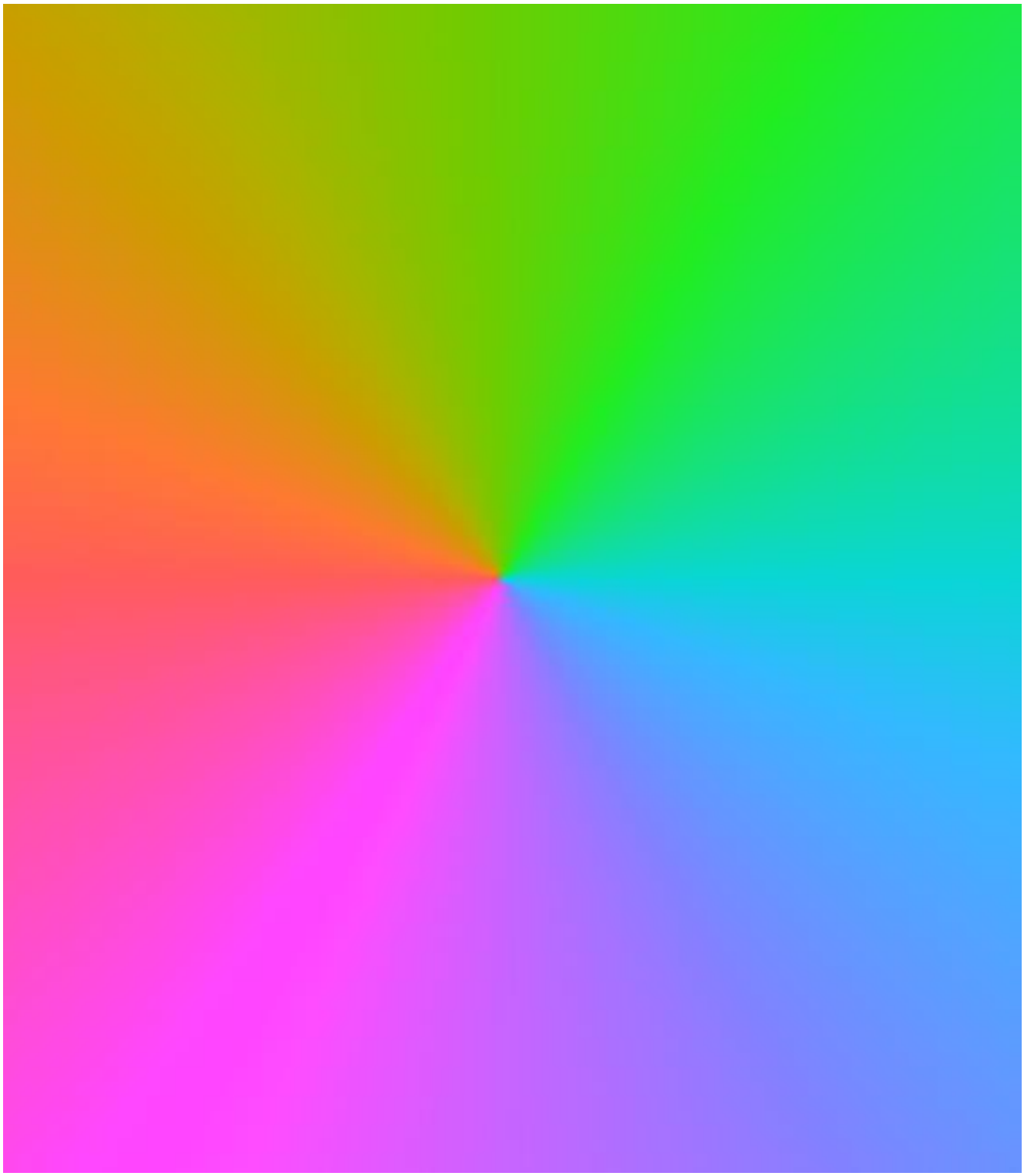
This function is the function of the producer class. This function adds elements to heap using thread. And prevents the thread operations from interfering by waiting.
T (n * log(n)) = O (n * log(n)).

```
public void run()
```

This function is the function of the consumer class. This function extracts the element from heap using the thread. And prevents the thread operations from interfering by waiting.
T (n) = O (n).

3 RESULT

3.1 Test Cases



3.2

```
C:\Program Files\Java\jdk-10.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\Int
 -classpath C:\Users\nurullah\IdeaProjects\2019_HW5\out\production\2019_HW5 HW5.Main C:\\U
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [200, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [201, 159, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
Thread 1: [202, 158, 0]
```

```
Thread 3-PQBMX: [250, 124, 46]
Thread 3-PQBMX: [249, 125, 45]
Thread 3-PQBMX: [249, 125, 45]
Thread 3-PQBMX: [249, 125, 45]
Thread 3-PQBMX: [248, 126, 44]
Thread 3-PQBMX: [248, 126, 44]
Thread 3-PQBMX: [248, 126, 44]
Thread 3-PQBMX: [248, 126, 44]
Thread 3-PQBMX: [247, 126, 43]
Thread 3-PQBMX: [247, 126, 43]
Thread 3-PQBMX: [247, 126, 43]
Thread 3-PQBMX: [246, 127, 42]
Thread 3-PQBMX: [246, 127, 42]
Thread 3-PQBMX: [246, 127, 42]
Thread 3-PQBMX: [245, 128, 41]
Thread 3-PQBMX: [245, 128, 41]
Thread 3-PQBMX: [245, 128, 41]
Thread 3-PQBMX: [245, 128, 41]
Thread 3-PQBMX: [244, 129, 40]
Thread 3-PQBMX: [244, 129, 40]
Thread 3-PQBMX: [244, 129, 40]
Thread 3-PQBMX: [244, 129, 40]
Thread 3-PQBMX: [243, 129, 39]
Thread 3-PQBMX: [243, 129, 39]
Thread 3-PQBMX: [243, 129, 39]
Thread 3-PQBMX: [242, 131, 38]
Thread 3-PQBMX: [242, 130, 38]
Thread 3-PQBMX: [242, 130, 38]
Thread 3-PQBMX: [242, 130, 38]
Thread 3-PQBMX: [241, 131, 37]
Thread 3-PQBMX: [241, 131, 37]
Thread 3-PQBMX: [241, 131, 37]
Thread 3-PQBMX: [240, 132, 36]
Thread 3-PQBMX: [240, 132, 36]
Thread 3-PQBMX: [240, 132, 36]
Thread 3-PQBMX: [240, 132, 36]
Thread 3-PQBMX: [239, 132, 35]
Thread 3-PQBMX: [239, 132, 35]
Thread 3-PQBMX: [239, 132, 35]
Thread 3-PQBMX: [238, 134, 34]
Thread 3-PQBMX: [238, 133, 34]
Thread 3-PQBMX: [238, 133, 34]
Thread 3-PQBMX: [238, 133, 34]
```

```
Thread 1: [255, 115, 59]
Thread 1: [255, 115, 59]
Thread 1: [255, 115, 59]
Thread 1: [255, 115, 59]
Thread 1: [255, 115, 59]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 114, 60]
Thread 1: [255, 113, 60]
Thread 2-PQLEX: [255, 121, 51]
Thread 2-PQLEX: [255, 121, 51]
Thread 2-PQLEX: [255, 121, 50]
Thread 2-PQLEX: [255, 120, 54]
Thread 2-PQLEX: [255, 120, 53]
Thread 2-PQLEX: [255, 120, 53]
Thread 2-PQLEX: [255, 120, 52]
Thread 2-PQLEX: [255, 119, 54]
Thread 2-PQLEX: [255, 119, 54]
Thread 2-PQLEX: [255, 119, 54]
Thread 2-PQLEX: [255, 118, 55]
Thread 2-PQLEX: [255, 118, 55]
Thread 2-PQLEX: [255, 118, 55]
Thread 2-PQLEX: [255, 118, 55]
Thread 2-PQLEX: [255, 117, 57]
Thread 2-PQLEX: [255, 117, 56]
Thread 2-PQLEX: [255, 117, 56]
Thread 2-PQLEX: [255, 116, 58]
Thread 2-PQLEX: [255, 116, 58]
Thread 2-PQLEX: [255, 116, 57]
Thread 2-PQLEX: [255, 115, 59]
Thread 2-PQLEX: [255, 115, 59]
Thread 2-PQLEX: [255, 115, 59]
Thread 2-PQLEX: [255, 115, 59]
Thread 2-PQLEX: [255, 114, 60]
Thread 2-PQLEX: [255, 114, 60]
Thread 2-PQLEX: [255, 114, 60]
Thread 2-PQLEX: [255, 113, 60]
```

```
Thread 2-PQLEX: [230, 138, 26]
Thread 2-PQLEX: [229, 139, 25]
Thread 2-PQLEX: [229, 139, 25]
Thread 2-PQLEX: [229, 139, 25]
Thread 3-PQBMX: [255, 120, 54]
Thread 3-PQBMX: [255, 120, 53]
Thread 3-PQBMX: [255, 120, 53]
Thread 3-PQBMX: [255, 119, 54]
Thread 4-PQEUC: [245, 128, 41]
Thread 3-PQBMX: [255, 119, 54]
Thread 1: [255, 113, 60]
Thread 2-PQLEX: [228, 140, 24]
Thread 1: [255, 113, 61]
Thread 1: [255, 113, 61]
Thread 3-PQBMX: [255, 119, 54]
Thread 3-PQBMX: [255, 118, 55]
Thread 3-PQBMX: [255, 118, 55]
Thread 3-PQBMX: [255, 118, 55]
Thread 3-PQBMX: [255, 118, 55]
Thread 3-PQBMX: [255, 117, 57]
Thread 2-PQLEX: [255, 113, 61]
Thread 2-PQLEX: [228, 140, 24]
Thread 2-PQLEX: [228, 140, 24]
Thread 2-PQLEX: [228, 140, 24]
Thread 4-PQEUC: [245, 128, 41]
Thread 2-PQLEX: [227, 140, 23]
Thread 1: [255, 113, 61]
Thread 3-PQBMX: [255, 117, 56]
Thread 3-PQBMX: [255, 117, 56]
Thread 3-PQBMX: [255, 116, 58]
Thread 3-PQBMX: [255, 116, 58]
Thread 3-PQBMX: [255, 115, 59]
Thread 3-PQBMX: [255, 115, 59]
Thread 3-PQBMX: [255, 115, 59]
Thread 3-PQBMX: [255, 115, 59]
Thread 3-PQBMX: [255, 116, 57]
Thread 3-PQBMX: [255, 114, 60]
Thread 3-PQBMX: [255, 114, 60]
Thread 3-PQBMX: [255, 114, 60]
Thread 3-PQBMX: [255, 113, 61]
Thread 3-PQBMX: [255, 113, 60]
Thread 3-PQBMX: [209, 153, 5]
Thread 3-PQBMX: [209, 153, 5]
```