**King Fahd University of Petroleum & Minerals**

**College of Computer Science and Engineering**

**Information and Computer Science Department**

**ICS 202 – Data Structures**

# Binary Heaps

**Objectives**

The objective of this lab is to design, implement and use Binary Heaps.

**Outcomes**

After completing this Lab, students are expected to:

• Design classes for Binary Heaps.

• Use Binary Heaps for a real life application.

**Notes**

For the purpose of this lab, you may download the attached programs.

**Lab Exercises**

1. Complete the class **BinaryHeap.java** by providing the code for **percolateUp, percolateDown, buildHeapBottomUp**  and **buildHeapTopDown** methods. Test your methods by modifying the provided test class (main method in the **BinaryHeap.java** file). Are you getting the same result using **topDown** and **bottomUp** methods?

Your output should be as follows: For bottom-up heap construction:

Text, letter

Description automatically generated

For top-down heap construction (comment the bottom-up statement in constructor)

Text, letter

Description automatically generated

(Observe the difference in the order of the elements 6, 8, 4, 9 and 10 in the constructed heap).

1. Patients arrive at a hospital with varying priorities. Each patient has the following attributes: Name (String), Emergency level (integer: 5 is the most urgent and 1 is the least urgent).

|  |  |  |
| --- | --- | --- |
| Level V | Resuscitation | See patient immediately |
| Level IV | Emergency | Within 15 minutes |
| Level III | Urgency | Within 20 minutes |
| Level II | Less Urgency | Within 30 minutes |
| Level I | Non-Urgency | Within 60 minutes |

Write a class in Java modeling a **Patient** which extends **Comparable**. Two patients can be compared based on their emergency level. If the emergency level is the same, then their serial is opposite of the alphabetical order of their names.

Now create an array of 10 patients at random in a main class (**Hospital**). [Provide their names but generate the emergency level randomly].

(a) Print the array,

(b) Create a binary heap of these patients using **enqueue**,

(c) Now heapsort these and print them.

This application illustrates use of a heap as a priority queue. Sample output is as follows:

Text, table

Description automatically generated