
CS202, Fall 2023

Homework 3 - Heaps and Priority Queues

Due: 30/11/2023

Before you start your homework please read the following instructions carefully:

FAILURE TO FULFIL ANY OF THE FOLLOWING REQUIREMENTS WILL RESULT IN A GRADE SCORE OF 0 (zero) WITHOUT ANY CHANCE OF REDEMPTION.

- See the course page for any late submission policies and Honor Code for Assignments.
- Upload your solutions in a single ZIP archive using the Moodle submission form. Name the file as `studentID_name_surname_hw3.zip`.
- Your ZIP archive should contain **only** the following files:
  - Your `.cpp` and `.h` files, as well as a file named `HospitalSimulationMgr.cpp`, which includes the main function and Makefile.
  - Do not forget to put your name, student id, and section number in all of these files. Add a header (see below) to the beginning of each file:

    ```
    /**
     * Author : Name & Surname
     * ID: 12345678
     * Section : 1
     * Homework : 3
     * Description : description of your code
     */
    ```
  - Do not put any unnecessary files such as the auxiliary files generated from your preferred IDE.
- Your code must compile.
- Your code must be complete.
- Your code must run on the **dijkstra.ug.bcc.bilkent.edu.tr** server.
- For any question related to the homework contact your TA: *ziya.ozgul@bilkent.edu.tr*

Hospital Simulation

In this programming assignment, you are asked to find a software solution to the scheduling problem of Bilkent Hospital. The hospital's manager is trying to figure out how many doctors should work in the patient service of the hospital. For each doctor in the hospital, the expense of the hospital increases; however according to the standards of the hospital, the average waiting time for all patients should not exceed a given amount of time. So, he needs to optimize this number and asks for your help in this task. The hospital has the data of predict examination time of patient. Your program should use these data to calculate average waiting times and find out the minimum number of doctors needed to meet the average waiting time requirement.

The data are stored in a plain text file[1]. The first line of the file contains the number of patients. The subsequent lines contain four integers, each separated by one or more whitespace characters (space or tab). These denote, respectively, the patient id, the priority of patient, arrival time of patient (in minutes from a given point [e.g. 12:00 am]) and service time (in minutes).

For example, from the file content given below, we understand there are 3 patients. The first patient with id 1 arrives at the hospital at minute 1, and his examination lasts for 5 minutes. His priority level is 3. The second patient with id 14 arrives at the bank at minute 70, and his examination lasts 10 minutes. The third patient with id 5 arrives at the bank at minute 82, and his examination lasts 70 minutes. The second and third patients have priority level of 2.

Sample input file:

```
3
1       3    1     5
14      2    70    10
5       2    82    70
```

In this assignment, you are asked to write a simulation program that reads patient data from an input file and calculates the minimum number of doctors required for a given maximum average waiting time.

In your implementation, you may make the following assumptions:

- The data file will always be valid. All data are composed of integers.
- In the data file, the patients are sorted according to their arrival times.

Your implementation must obey the following requirements:

- The patient with the highest priority should be examined first.
- In case of having two patients with the same highest priority, the patient who has waited longer should be selected first.

[1] The file is a UNIX-style text file with the end-of-line denoted by a single \n (ASCII 0x0A)

- If more than one doctor is available at a given time; the patient is assigned to the doctor with a lower id.
- When a doctor starts giving a service to a patient, the doctor should finish his service with this patient even though another patient with a higher priority comes to the bank.
- Once a patient is assigned to a doctor, the doctor immediately starts carrying out the examination of that patient and is not available during the examination time given for that patient. After the examination of that patient carries out, the doctor becomes available immediately.
- The waiting time of a patient is the duration (difference) between the arrival time of the patient and the time he is assigned to a doctor.

In your implementation, you MUST use a heap-based priority queue to store patients who are waiting for a doctor (i.e., to store patients who have arrived at the hospital but their examinations have not been conducted yet). If you do not use such a heap-based priority queue to store these patients, then you will get no points from this homework.

The name of the input file and the maximum average waiting time will be provided as command line arguments to your program. Thus, your program should run using two command line arguments. Thus, the application interface is simple and given as follows:

```
username@dijkstra:~>./simulator <filename> <avgwaitingtime>
```

Assuming that you have an executable called "simulator", this command calls the executable with two command line arguments. The first one is the name of the file from which your program reads the customer data. The second one is the maximum average waiting time; your program should calculate the minimum number of cashiers required for meeting this **`avgwaitingtime`**. You may assume that the maximum average waiting time is given as an integer.

SAMPLE OUTPUT:

Suppose that you have the following input file consisting of the patient data. Also suppose that the name of the file is `patients.txt`.

```
12
1       4     1       10
2       5     1       14
3       3     1       6
4       3     1       5
5       4     4       10
6       5     7       14
7       4     9       10
8       5     11      14
9       3     13      6
10      3     14      5
11      4     15      10
12      5     17      14
```

The output for this input file is given as follows for different maximum average waiting times. Please check your program with this input file as well as the others that you will create. Please note that we will use other input files when grading your assignments.

```
username@dijkstra:~>./simulator   patients.txt    5

Minimum number of doctors required: 4

Simulation with 4 doctors:

 Doctor  0 takes  patient  2 at minute 1 (wait: 0 mins)
 Doctor  1 takes  patient  1 at minute 1 (wait: 0 mins)
 Doctor  2 takes  patient  3 at minute 1 (wait: 0 mins)
 Doctor  3 takes  patient  4 at minute 1 (wait: 0 mins)
 Doctor  3 takes  patient  5 at minute 6 (wait: 2 mins)
 Doctor  2 takes  patient  6 at minute 7 (wait: 0 mins)
Doctor 1 takes patient 8 at minute 11 (wait: 0 mins)
Doctor 0 takes patient 7 at minute 15 (wait: 6 mins)
Doctor 3 takes patient 11 at minute 16 (wait: 1 mins)
Doctor 2 takes patient 12 at minute 21 (wait: 4 mins)
Doctor 0 takes patient 9 at minute 25 (wait: 12 mins)
Doctor 1 takes patient 10 at minute 25 (wait: 11 mins)

Average waiting time: 3 minutes

username@dijkstra:~>./simulator   patients.txt    10

Minimum number of doctors required: 3

Simulation with 3 doctors:

 Doctor  0 takes  patient  2 at minute 1 (wait: 0 mins)
 Doctor  1 takes  patient  1 at minute 1 (wait: 0 mins)
 Doctor  2 takes  patient  3 at minute 1 (wait: 0 mins)
 Doctor  2 takes  patient  6 at minute 7 (wait: 0 mins)
 Doctor  1 takes  patient  8 at minute 11 (wait: 0 mins)
 Doctor  0 takes  patient  5 at minute 15 (wait: 11 mins)
Doctor 2 takes patient 12 at minute 21 (wait: 4 mins)
Doctor 0 takes patient 7 at minute 25 (wait: 16 mins)
Doctor 1 takes patient 11 at minute 25 (wait: 10 mins)
Doctor 0 takes patient 4 at minute 35 (wait: 34 mins)
Doctor 1 takes patient 9 at minute 35 (wait: 22 mins)
Doctor 2 takes patient 10 at minute 35 (wait: 21 mins)

Average waiting time: 9.83333 minutes
```

Grading

If your code runs in $O(n^2 * \log n)$ time, you will get up to 75 points. If your code runs in $O(n \log^2 n)$, you may get full points.