CS3353

Programming assignment

1. Implement insertion and deletion for a ***max-heap*** as defined in class.

Input: list of pairs of an integer and an operation, e.g., pre (or post or in) 1.in 3.in 5.in del 2.in ….. Note that the “pre”, “post” or “in” is the method you use to traverse the heap for output display. Also, note that “del” will delete the max key by default.

/\* Notice that pairs are delimited by a single space in between. Note that input is to be read in from a file in a batch. \*/

Output: the list before operation and the list after operation displayed on the screen standard output (not into a file) by using the traversal method as specified in your input file.

**Input examples:**

pre 1.in 34.in del 24.in

post 1.in 3.in 5.in del 2.in

in 1.in del 12.in 5.in

pre 1.in 3.in 25.in del 4.in 231.in

**Example input file:**

pre 1.in 3.in 5.in del 2.in

**Example output:** For the same input example file as above, the output should will be as follows:

Operation: 1.in

Heap before: NULL

Heap after: 1

Operation: 3.in

Heap before: 1

Heap after: 3 1

Operation: 5.in

Heap before: 3 1

Heap after: 5 1 3

Operation: del

Heap before: 5 1 3

Heap after: 3 1

Operation: 2.in

Heap before: 3 1

Heap after: 3 1 2

Note: The output will be different if the traversal mentioned is different in the input file i.e. for input “ post 1.in 3.in 5.in del 2.in” .The “post” traversal would produce a different output even though the max heap is the same tree. The output should be printed as per the traversal mentioned in the input.

2. This problem is to simulate ***scheduling CPU jobs*** by using a **PRIORITY QUEUE** to be built. Your program to be written for the simulation should run in a loop, each iteration of which corresponds to a ***time slice*** for the CPU. Each job is assigned a priority, which is an integer between -10 (highest priority) and +10 (lowest priority), inclusive. Each job is also assigned an arrival time – the time at which the job shows up at the CPU. From among all jobs waiting to be processed in a time slice, the CPU, the CPU must work on a job with highest priority.

In this simulation, each job will come with arrival time, which is the time at which the job has arrived (integer >= 0). Each job will also include a ***length*** value, which is an integer between 1 and 100, inclusive, indicating the number of time slices that are needed to process this job. Jobs **CAN** be interrupted in the middle if another job with higher priority arrives. Your simulator must output the name of the job running on the CPU in each time slice as shown below.

**Note**: If there are two or more jobs with same priority, the jobs that arrived earlier are executed first (LIFO – last come first out) basis.

**Input File format:** Each line in the input file contains the details one single job delimited by “space”. Read the entire file first and then process the priority queue algorithm on all the jobs based on the arrival time. As some jobs will have a different arrival time, the priority queue will have to be re-done after every time slice. The order of the jobs in the input file are random and not sorted in any way.

JobName priority arrivalTime lengthOfJob

**Example input file:**

Job114 -9 0 25

Job345 2 0 66

Job234 -10 10 5

Job999 10 27 56

**Output format:** Output on the command line should display the job arrivals on that time slice (if any) and job that is being executed for that timeslice along with the priority and the remaining time of that job.

Time Slice #n – Job arrivals at this time slice (if any)

Executing JobName priority remainingLengthOfJob

**Example output File:** For the same input example file as above, the output should will be as follows:

Time Slice #1 - Job114 arrived

Job345 arrived

Executing Job114 -9 25

Time Slice #2 - Executing Job114 -9 24

Time Slice #3 - Executing Job114 -9 23

Time Slice #4 - Executing Job114 -9 22

Time Slice #5 - Executing Job114 -9 21

.

.

Time Slice #10 - Job234 arrived

Executing Job234 -10 5

Time Slice #11 - Executing Job234 -10 4

Time Slice #12 - Executing Job234 -10 3

Time Slice #13 - Executing Job234 -10 2

Time Slice #14 - Executing Job234 -10 1

Time Slice #15 - Executing Job114 -9 16

Time Slice #16 - Executing Job114 -9 15

.

.

Time Slice #27 - Job999 arrived

Executing Job114 -9 4

Time Slice #28 - Executing Job114 -9 3

Time Slice #29 - Executing Job114 -9 2

Time Slice #30 - Executing Job114 -9 1

Time Slice #31 - Executing Job345 2 66

Time Slice #32 - Executing Job345 2 65

Time Slice #33 - Executing Job345 2 64

Time Slice #34 - Executing Job345 2 63

Time Slice #35 - Executing Job345 2 62

.

.

Time Slice #96 - Executing Job345 2 1

Time Slice #97 - Executing Job999 10 56

Time Slice #98 - Executing Job999 10 55

Time Slice #99 - Executing Job999 10 54

.

.

**Example input file 2:**

Job114 -9 0 25

Job345 8 0 66

Job234 -10 2 5

Job999 6 99 7

Job456 9 6 10

Job121 5 8 10

**Example output File:** For the same input example file as above, the output should will be as follows:

Time Slice #1 - Job114 arrived

Job345 arrived

Executing Job114 -9 25

Time Slice #2 - Job234 arrived

Executing Job234 -10 5 24

Time Slice #3 - Executing Job234 -10 4

Time Slice #4 - Executing Job234 -10 3

Time Slice #5 - Executing Job234 -10 2

Time Slice #6 - Job456 arrived

Executing Job234 -10 1

Time Slice #7 - Executing Job114 -9 24

Time Slice #8 - Job121 arrived

Executing Job114 -9 24

.

.

Time Slice #25 - Executing Job114 -9 1

Time Slice #26 - Executing Job114 -9 5

Time Slice #27 - Executing Job114 -9 4

Time Slice #28 - Executing Job114 -9 3

Time Slice #29 - Executing Job114 -9 2

Time Slice #30 - Executing Job114 -9 1

Time Slice #31 - Executing Job121 5 10

Time Slice #32 - Executing Job121 5 9

.

.

Time Slice #40 - Executing Job121 5 1

Time Slice #41 - Executing Job345 8 66

.

.

Time Slice#99 - Job999 arrived

Executing Job999 6 7

Time Slice#100 - Executing Job999 6 6

Time Slice#102 - Executing Job999 6 5

Time Slice#103 - Executing Job999 6 4

Time Slice#104 - Executing Job999 6 3

Time Slice#105 - Executing Job999 6 2

Time Slice#106 - Executing Job999 6 1

Time Slice#107 - Executing Job345 8 8

Time Slice#108 - Executing Job345 8 7

.

.

Time Slice#114 - Executing Job345 8 1

The output of **all** the time slices **MUST** be printed to the console.

A command line user interface to read the input must be provided as follows:

Enter your input file name:

The program is to be submitted to the directory in “handin”.

\*Provide a README file with specific instructions for compilation and execution for the grader to follow, and any notes you want to add.