Lab09

100 Points

Due date:
11:59pm, Monday 04/15/2019 for Tuesday labs.
11:59pm, Wednesday 04/17/2019 for Thursday labs.

Purpose:
For this lab, you will implement a fair scheduler in C++ using a Minmax heap.

General Requirements:
In this assignment, **you will develop an array-based implementation** of a Minmax heap. You will design the scheduler in such a way that every process gets CPU time without waiting for a long time, even if the process has low priority. Each process will be given a pair containing the priority of the element and its expected runtime. In particular, the first element is the priority of that process, i.e., priority(x), where 1 is the highest priority, and 10 is the lowest priority. The second element is the estimated time (y) in milliseconds required for that process to run.

The scheduler divides the CPU time into chunks. The first chunk will run processes for 20 milliseconds, where it will start scheduling processes starting from the highest priority process at the root (min) level. The next chunk will run for 10 milliseconds, where it will start scheduling processes starting with the lowest priority process at the max level. This cycle is repeated until all the processes are finished processing. The time series for the cycle starts with T1 and ends at Tn, which is the last chunk when all of the processes have finished executing. All odd time slots (i.e., T1, T3, T5, etc.) will have a duration of 20ms, and the even time slots (i.e., T2, T4, T6, etc.) will have a duration of 10ms. Each time a process is run, the time in the pair representing the process must be updated with the remaining time for the process to complete. The processes which are completed should be removed from the heap**. The initial build of the Minmax heap should use the bottom-up approach**. Each time you remove a process, you should heapify the heap.

In the Minmax heap:

The root of T is at A[1].

Here is where the min and max nodes are located:
Min node: floor(lg(i)) = even
Max node: floor(lg(i)) = odd

The grandparent of A[i] is at A[floor(i/4)], if it exists.

The Minmax heap methods should be implemented as follows:

- buildheap() - should build the Minmax heap using the **bottom up approach.**

- insert(x) - should insert x into the Minmax heap. This means you should add a new process to all the existing processes in the heap and see how your scheduling order changes with the new process added.
- deletemin() - should delete the process with highest priority from the Minmax heap.
- deletemax() - should delete the lowest priority process from the Minmax heap.
- findmin() - should print the highest priority process from the Minmax heap.
- findmax() - should print the lowest priority process from the Minmax heap.
- runSchedule() - should print out the scheduling sequence in which each of the processes is executed starting from time chunk T1 and ending with the time chunk Tn. Finally, it should give the scheduling order in which the processes are executed.

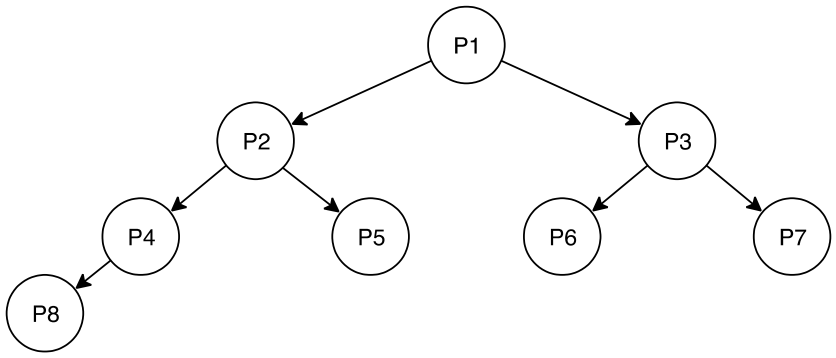
The file you will read the processes from will be data.txt. You may hard code the file name in your program if you wish.

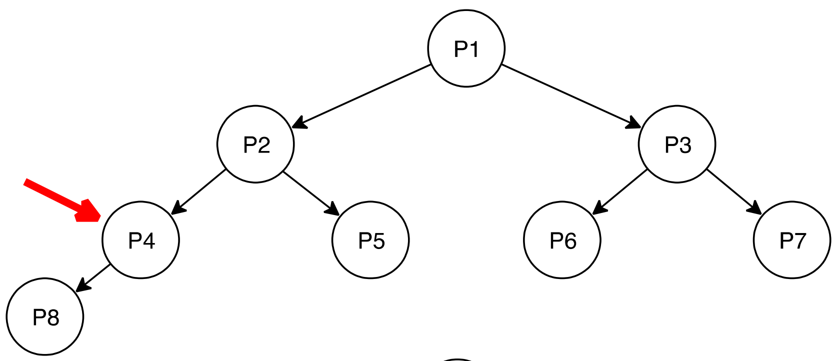
In this lab, you should build the heap using the samples which are in the data.txt. After that, your program should have a simple menu like this:

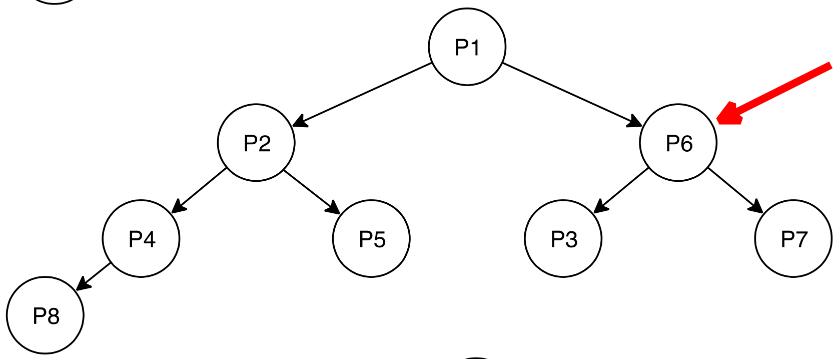
```
-----------------------------------------------------------
Please choose one of the following commands:
1-  Insert
2-  DeleteMin
3-  DeleteMax
4-  FindMin
5-  FindMax
6-  RunSchedule
7-  Exit
```

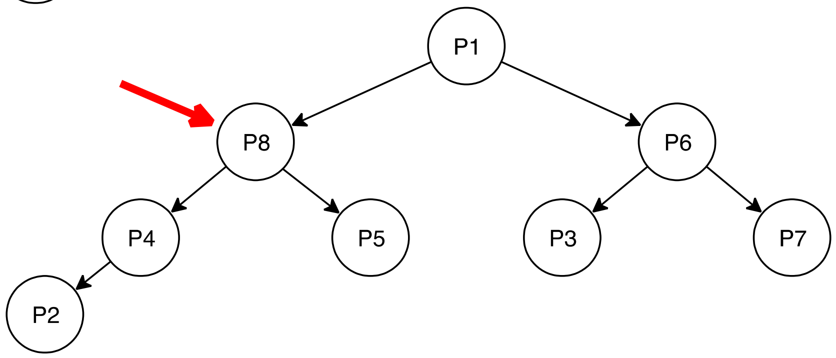
Expected output:
data.txt**:**
(1,10), (3,12), (1,15), (2,8), (5,20), (7,23), (4,5), (10,10)

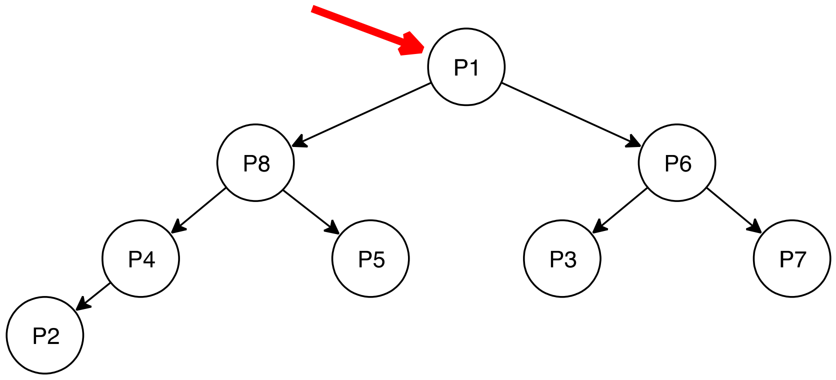
Below we demonstrate the process of building the heap on the above dataset:











Please choose one of the following commands:

1- Insert
2- DeleteMin
3- DeleteMax
4- FindMin

5- FindMax
6- RunSchedule
7- Exit
>6

Initial Minmax heap:
P1
P8 P6
P4 P5 - P3 P7
P2

Scheduling output:

T1: 20 ms
P1(1,0), P3(1,5)
P1 and P3 are executed. P1 is completed and removed from heap.

Updated Minmax heap:
P3
P8 P6
P4 P5 - P2 P7

T2: 10 ms
P8(10,0)
P8 is completed and removed from heap.

Updated Minmax heap:
P3
P5 P6
P4 P7 – P2

T3: 20 ms
P3(1,0), P4(2,0), P2(3,5)
P3, P4 and P2 are executed. P3 and P4 are completed and removed from heap.

Updated Minmax heap:
P2
P5 P6
P7

T4: 10 ms
P6(7,13)
P6 is executed

Updated Minmax heap:
P2
P5 P6
P7

T5: 20ms
P2(3,0), P7(4,0), P5(5,10)
P2, P7 and P5 are executed. P2 and P7 are completed and removed from heap.

Updated Minmax heap:
P5
P6

T6:10ms
P6(7,3)
P6 is executed

Updated Minmax heap:
P5
P6

T7:20ms
P5(5,0), P6(7,0)
P5 and P6 executed. P5 and P6 are completed and removed from heap.

Updated Minmax heap:
Heap is empty

Scheduling order:
P1 P8 P3 P4 P2 P7 P5 P6

```
----------------------------------------------------------
Please choose one of the following commands:
1-   Insert
2-   DeleteMin
3-   DeleteMax
4-   FindMin
5-   FindMax
6-   RunSchedule
7-   Exit
>1
Add a process:
>(4,10)
```

```
-----------------------------------------------------------
Please choose one of the following commands:
1-   Insert
2-   DeleteMin
3-   DeleteMax
4-   FindMin
5-   FindMax
6-   RunSchedule
7-   Exit
>6
```

Initial Minmax heap:
P1
P8 P6
P4 P5 - P3 P7
P2 P9

Scheduling output:

T1: 20 ms
P1(1,0), P3(1,5)
P1 and P3 are executed. P1 is completed and removed from heap.

Updated Minmax heap:
P3
P8 P6
P4 P5 – P9 P7
P2

T2: 10 ms
P8(10,0)
P8 is completed and removed from heap.

Updated Minmax heap:
P3
P5 P6
P4 P2 – P9 P7

T3: 20 ms
P3(1,0), P4(2,0), P2(3,5)
P3, P4 and P2 are executed. P3 and P4 are completed and removed from heap.

Updated Minmax heap:
P2

P5 P6
P7 P9

T4: 10 ms
P6(7,13)
P6 is executed

Updated Minmax heap:
P2
P5 P6
P7 P9

T5: 20ms
P2(3,0), P9(4,0), P7(4,0)
P2, P9 and P7 are executed. P2, P9 and P7 are completed and removed from heap.

Updated Minmax heap:
P5
P6

T6:10ms
P6(7,3)
P6 is executed

Updated Minmax heap:
P5
P6

T7:20ms
P5(5,0), P6(7,0)
P5 and P6 executed. P5 and P6 are completed and removed from heap.

Updated Minmax heap:
Heap is empty

Scheduling order:
P1 P8 P3 P4 P2 P7 P5 P6

Please choose one of the following commands:
1- Insert
2- DeleteMin
3- DeleteMax
4- FindMin

```
5-   FindMax
6-   RunSchedule
7-   Exit
>4
P1(1,10)

------------------------------------------------------------
Please choose one of the following commands:
1-   Insert
2-   DeleteMin
3-   DeleteMax
4-   FindMin
5-   FindMax
6-   RunSchedule
7-   Exit
>5
P8(10,10)

------------------------------------------------------------
Please choose one of the following commands:
1-   Insert
2-   DeleteMin
3-   DeleteMax
4-   FindMin
5-   FindMax
6-   RunSchedule
7-   Exit
>2

------------------------------------------------------------
Please choose one of the following commands:
1-   Insert
2-   DeleteMin
3-   DeleteMax
4-   FindMin
5-   FindMax
6-   RunSchedule
7-   Exit
>3

------------------------------------------------------------
Please choose one of the following commands:
1-   Insert
2-   DeleteMin
```

```
3-   DeleteMax
4-   FindMin
5-   FindMax
6-   RunSchedule
7-   Exit
>6

Initial Minmax heap:
P3
P5 P6
P4 P2 – P9 P7

Scheduling output:

T1: 20 ms
P3(1,0), P4(2,3)
P3 and P4 are executed. P3 is completed and removed from heap.

Updated Minmax heap:
P4
P5 P6
P7 P2 – P9

T2: 10 ms
P6(7,13)
P6 is executed.

Updated Minmax heap:
P4
P5 P6
P7 P2 – P9

T3: 20 ms
P4(2,0), P2(3,0), P7(4,0)
P4, P2 and P7 are executed. P4, P2 and P7 are completed and removed from heap.

Updated Minmax heap:
P9
P5 P6

T4: 10 ms
P6(7,3)
P6 is executed
```

Updated Minmax heap:
P9
P5 P6

T5: 20ms
P9(4,0), P5(5,10)
P9 and P5 are executed. P9 is completed and removed from heap.

Updated Minmax heap:
P5
P6

T6:10ms
P6(7,0), P5(5,3)
P6 and P5 are executed. P6 is completed and removed from heap.

Updated Minmax heap:
P5

T7:20ms
P5(5,0)
P5 is executed. P5 is completed and removed from heap.

Updated Minmax heap:
Heap is empty

Scheduling order:
P1 P8 P3 P4 P2 P7 P5 P6

```
-----------------------------------------------------------
Please choose one of the following commands:
1-  Insert
2-  DeleteMin
3-  DeleteMax
4-  FindMin
5-  FindMax
6-  RunSchedule
7-  Exit
>7
Byebye!
```

Questions:

Please answer the following questions in not more than 5 lines each and submit it with your implemented code in PDF format.

1. What is the worst time complexity of each of the operations that you have implemented.
   - **a.** Add process into the scheduler.
   - **b.** Delete minimum priority process from the scheduler.
   - **c.** Delete maximum priority process from the scheduler.
2. Is the average case complexity equal to the worst case complexity in each of the 3 cases above? If not, mention the average case complexity for each of the above operations in a., b., and c., and also explain the reason in short for the difference.

Submission:

Follow the conventions below to facilitate grading:

Report

- **Please include your answers (answers.pdf) in your folder before compressing it.**

Source Code

Place all your source files (*.cpp, *.hpp) and input files in a single folder with no subfolders.

- Name your folder using the convention Lastname_LabX (e.g., Smith_Lab09).
- Include a functioning Makefile inside the folder. (The makefile should also include the clean command.)
- **Verify that your code runs on the lab Linux machines before submission.**

Compressed File

- Compress using .zip, .rar, or .tar.gz.
- Name your file using the convention Lastname_LabX (e.g., Smith_Lab09.zip).

Email

- Use the following subject for your email: Lastname_LabX (e.g., Smith_Lab09).
- Send your code to l290w868@ku.edu if you are in one of Lei's sections or to dhwanipandya1401@ku.edu if you are in one of Dhwani's sections.
- Anytime you have a question about the lab, put the word question in the subject of the email.