Lab Assignment 3: Trees

Task 3: Printer Simulation 2

Introduction

In this lab task you will simulate the scenario of a laboratory printer (as shown in Fig 1). On average there will be one print task in N seconds. The size of the print tasks ranges from 1 to M pages. The printer in the lab can process P pages per minute (i.e. 60 seconds) at good quality. The scenaria is similar to the one presented in the lab 2, but applies a priority queueing mechanism. The printer always processes the printing task with the smallest number of pages.

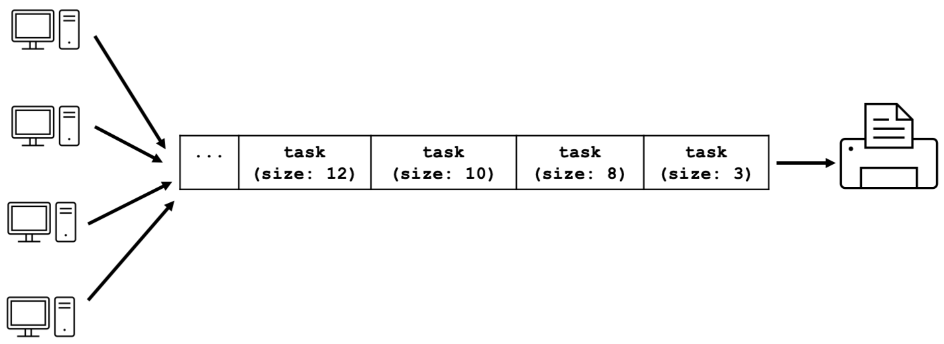


Figure 1: laboratory printer scenario

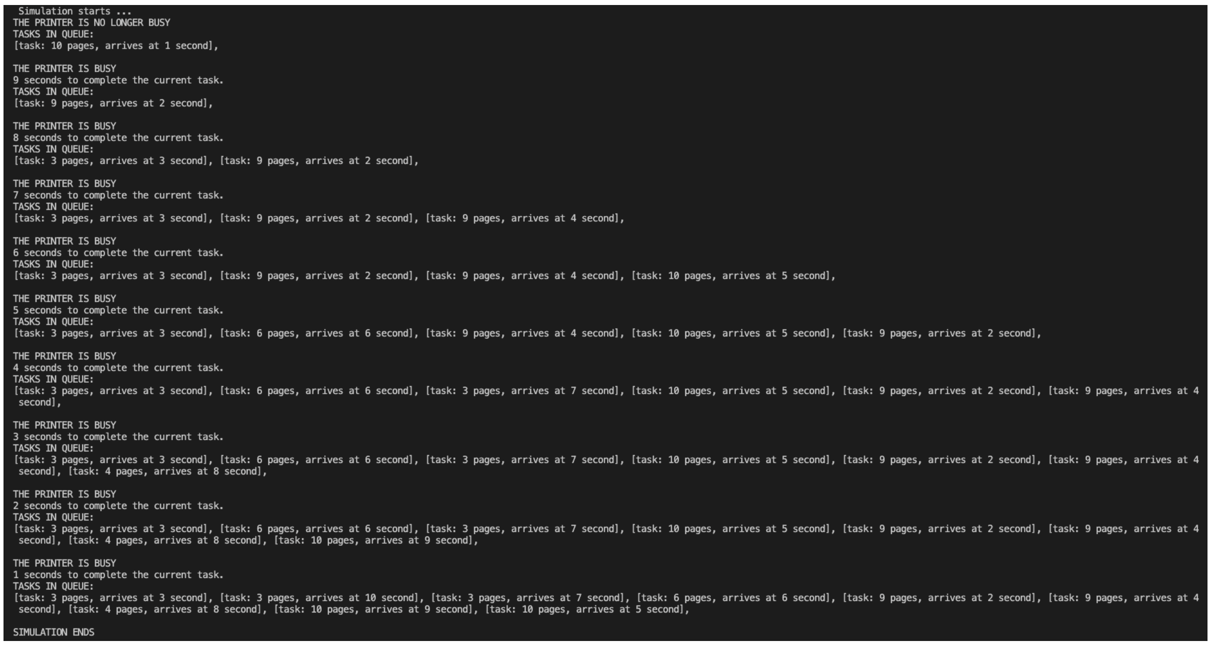
The simulation runs roughly as the steps as in the simulation presented in the lab 2:

- Set the printer to empty.
- Create a queue of print tasks.
- While (haven't reached clock limit)
  - Does a new print task get created? If so, add it to the queue with the current second as the timestamp upon its arrival.
  - If the printer is not busy and if a task is waiting, remove the next task from the print queue and assign it to the printer.
  - The printer now does one second of printing if necessary, i.e., subtracts one second from the time required for the task. If the task has been completed, in other words the time required has reached zero, the printer is no longer busy.
- Simulation ends.

Requirements

You should implement the interfaces for printer ADT, task ADT and priority queue ADT as defined in the header files (in printer_simulation_headers_priority_queue.zip), and a main function for the simulation process. As defined in the priority queue ADT interface, the queue should be implemented using a min binary heap. The binary heap should be represented as an array. In the exercise you have learned how to implement a max binary heap using array. You can also reuse your implementation for printer and task and the main function for the lab 2.

Once you've got your program implemented correctly, you should be able to produce output like the example shown in Fig 2, where $N = 1$, $M = 10$, $P = 60$, the simulation runs for 10 seconds and the array size is 20. You can choose different values for the simulation parameters in your program.

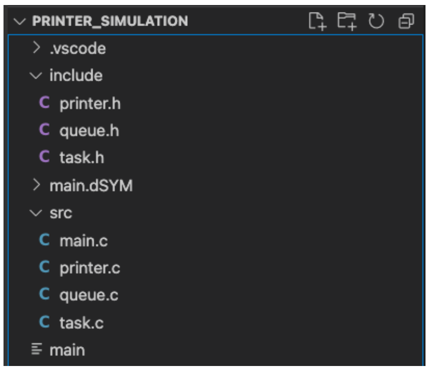


```
 Simulation starts ...
THE PRINTER IS NO LONGER BUSY
TASKS IN QUEUE:
[task: 10 pages, arrives at 1 second],

THE PRINTER IS BUSY
9 seconds to complete the current task.
TASKS IN QUEUE:
[task: 9 pages, arrives at 2 second],

THE PRINTER IS BUSY
8 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 9 pages, arrives at 2 second],

THE PRINTER IS BUSY
7 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 9 pages, arrives at 2 second], [task: 9 pages, arrives at 4 second],

THE PRINTER IS BUSY
6 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 9 pages, arrives at 2 second], [task: 9 pages, arrives at 4 second], [task: 10 pages, arrives at 5 second],

THE PRINTER IS BUSY
5 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 6 pages, arrives at 6 second], [task: 9 pages, arrives at 4 second], [task: 10 pages, arrives at 5 second], [task: 9 pages, arrives at 2 second],

THE PRINTER IS BUSY
4 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 6 pages, arrives at 6 second], [task: 3 pages, arrives at 7 second], [task: 10 pages, arrives at 5 second], [task: 9 pages, arrives at 2 second], [task: 9 pages, arrives at 4
 second],

THE PRINTER IS BUSY
3 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 6 pages, arrives at 6 second], [task: 3 pages, arrives at 7 second], [task: 10 pages, arrives at 5 second], [task: 9 pages, arrives at 2 second], [task: 9 pages, arrives at 4
 second], [task: 4 pages, arrives at 8 second],

THE PRINTER IS BUSY
2 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 6 pages, arrives at 6 second], [task: 3 pages, arrives at 7 second], [task: 10 pages, arrives at 5 second], [task: 9 pages, arrives at 2 second], [task: 9 pages, arrives at 4
 second], [task: 4 pages, arrives at 8 second], [task: 10 pages, arrives at 9 second],

THE PRINTER IS BUSY
1 seconds to complete the current task.
TASKS IN QUEUE:
[task: 3 pages, arrives at 3 second], [task: 3 pages, arrives at 10 second], [task: 3 pages, arrives at 7 second], [task: 6 pages, arrives at 6 second], [task: 9 pages, arrives at 2 second], [task: 9 pages, arrives at 4
 second], [task: 4 pages, arrives at 8 second], [task: 10 pages, arrives at 9 second], [task: 10 pages, arrives at 5 second],

SIMULATION ENDS
```

Figure 2: example program output

Deliverables

The deliverable is a zip file in which the files are organized as in Fig 3. Variables in your code files should have proper names, and the code has to include sufficient comments for readability.



```
PRINTER_SIMULATION
  > .vscode
  v include
      printer.h
      queue.h
      task.h
  > main.dSYM
  v src
      main.c
      printer.c
      queue.c
      task.c
    main
```

Figure 3: file organization