Jordan University of Science and Technology

Computer Engineering Department

CPE 473: Operating Systems

Programming Assignment #1: OS Scheduler

Notes:

1- **The due date is 18/11/2023 at 11:55 PM.**
2- **Late submissions (i.e. after the due date) will not be accepted.**
3- **Submit the source code files and write your name and ID as a comment on the top.**
4- **Groups are allowed (up to 2 students). Can be from different sections.**

In this assignment, you will implement a simple OS scheduler using C/C++. The scheduler's task is to receive a set of processes and their details and then decide the order of executing these processes based on the chosen algorithm. Finally, the scheduler will output the order of process execution, in addition to some stats about each of the processes.

The scheduling algorithm chosen for this assignment will be **Priority Scheduling**. The input will start with an integer N, representing the number of processes, followed by N lines (one for each process). For each line i, the line will start with a string s, representing the process name, followed by three numbers representing the arrival time, processing time, and priority for the ith process, respectively. These values will be separated by tabs (i.e. '\t'). The values for input numbers can be up to 100,000,000.

As you learned in class, the main aspect of Priority Scheduling is the priority value for each process (3rd column). However, If multiple processes have the same priority value, we decide based on the arrival time. In the case where processes have the same priority and arrival time, we choose the process that had its name listed first in the input file.

Your program should print a line indicating the order of executing the processes. Moreover, for each process, your program should print a line showing the process's name, response time, turnaround time, and delay. The order of these lines should match the execution order. See the sample output below for details.

The input will be read from a file (in.txt), and the output should be written to a file (out.txt). The output format must strictly match the formatting shown in the sample output.

Grading Rules:

- Each student is expected to fully understand all the aspects and details of the entire code.
- The grading will take place in a Linux environment (Ubuntu, using the g++ compiler).
- You MUST implement the code using this same environment to get a full mark.
- Your grade will depend on multiple aspects, such as:
    1. The correctness of your code's output across multiple secret test cases
    2. Your code's ability to handle corner test cases like having many processes or large processing times.
    3. Your answers to the questions during the discussion

Sample Input (file *in.txt*):

5

A 0 3 3

C 4 4 4

B 2 6 6

D 6 5 5

E 8 2 2

Sample output (file *out.txt*):

ABDCE

A: (response=0, turnaround=3, delay=0)

B: (response=1, turnaround=7, delay=1)

D: (response=3, turnaround=8, delay=3)

C: (response=10, turnaround=14, delay=10)

E: (response=10, turnaround=12, delay=10)