CS 4323 Design and Implementation of Operating Systems I

Group Project: Full Marks 100
(Due Date and End Date: 10/25/2023, 11:59 PM CT)

This is a group project assignment that must be done in the allocated group and using either C, C++ or Java programming language. If you do not know the group members, then please refer the module **GroupProject** under **Home** page in Canvas.

Students are free to make assumptions and have their own creativity on the project if they do not conflict with the requirements specified in the assignment.

The project will familiarize students with the scheduling and the concept of queue. Please make sure to refer to the lecture slides and the textbook for theoretical concept behind scheduling and queue.

Problem description:

In this project, you need to implement the pre-emptive version of the shortest-job first (SJF) algorithm. Your program needs to read the data from the given input file. The structure of the input file is shown below:

```
4

P0      3       6
P1      5       5
P2      2       7
P3      0       1
P4      2       4
P5      4       2
P6      8       2
```

Fig. Input file structure

Description of the input file structure is as follows:

- The first line represents the queue size.
- The second line is empty.
- From the third line onwards, the data for the process starts, where:
    - the first column represents the process ID.
    - the second column represents the process arrival time.
    - the third column represents each process burst time.

Your program should run for any input file with similar file structure. The TA will use similar input file but with different values.

Project requirements:

The project needs to account for the queue size. The size of the queue specifies how many processes can be accommodated in the system at any time. For example, if the system has a queue of size 5, then at any time, the system can only accommodate five processes at the most. If the queue is full, any new process arriving in the system will be dropped. Once the process is dropped, it will not be considered for the scheduling. Only the processes in the queue will be served after scheduling. This implies that the scheduling algorithm will be applied to those processes P_i that are present in the queues.

Your program should display the following output:

Output 1: Show the scheduling order of the processes using a Gantt chart, where the process ID and the time information should be clearly specified.

Output 2: For each process, you need to compute the response time, waiting time and the turnaround time. This information needs to be displayed in the tabular form along with the process ID, process arrival time and the process burst time, which are obtained from the input file.

Output 3: For the algorithm, you need to compute the average response time, the average waiting time and the average turnaround time.

Output 4: System statistics showing how many processes were served and how many processes were dropped. For this part, your program should implement queue.

Grading Rubrics:

The grading will be as follows:

• Submission of progress report meeting all the requirements	[10 Points]
• Correctly reading the input file	[5 Points]
• Correct implementation of the Output 1	[20 Points]
• Correct implementation of the Output 2	[25 Points]
• Correct implementation of the Output 3	[20 Points]
• Correct implementation of the Output 4	[15 Points]
• Report	[5 Points]

Note:

- Failure to follow standard programming practices will lead to points deduction, even if the program is running correctly. Some of the common places where you could lose points are:
  - Program not compiling successfully: -20 points
    - The TA will run the program in CSX machine.
  - No comments on code (excluding function): -5 points
  - No comments on function: -5 points
  - Not writing meaningful identifiers: -5 points
  - Failure to submit files as specified: -10 points
  - Not appropriate message displayed for user interaction: -10 points

Grading Criteria:

This is a group project. So, grading will focus on students' ability to work in the group and successfully completing the work. Each member in the group should coordinate as a team rather than individual and that is the key to scoring maximum points.

Points to remember:

- Failure to complete the project as a group will deduct 20 points, even though every individual has completed their part. As this is a group project and students are expected to learn to work in the group and meet the team's objective, and not just individual objective(s).

- It is not necessary that all group members get equal points. It depends on what responsibility each student has taken and whether the student has delivered the task.

- Work need to be distributed uniformly among all group member and should be clearly specified in the report.
    - Be sure to submit a progress report as specified in the progress report file.

- It is the group's responsibility to alert the instructor with any issue that is happening in the group.
    - Students are supposed to use the group created in the Canvas for all correspondence. That will serve as the proof, in case there is any dispute among the group members.
    - Time is very important. Group members are expected to respond quickly.
    - You need to have sufficient days to combine individual work. You are going to make one submission as a group and not individually.

- If any of the group member have plagiarized the code, then the complete project will be considered plagiarized, and all members will be awarded zero. So, each group member needs to be very alert of plagiarism issue, not only of their code but fellow member's code. Any plagiarism issue should be brought to the instructor's attention as soon as possible. University and department have strict policy on plagiarism and will be strictly followed.

- Final project will be evaluated together with the progress report. So, they should have a correlation. Commitments on the progress report should be reflected on the final submissions.
    - Grades of the progress report will be included in the final grading. Please refer to the grading rubric.

- A group page has been created in the Canvas, which can be accessed through the **People** from the left of the Canvas page. It is highly encouraged that every group member uses this platform for communication and to share files. In case of any dispute, only the information shared on this page will be used to resolve dispute.

Submission Guidelines:

- Only one group member needs to submit the project in the Canvas. Once the project is submitted in the Canvas, the member who has submitted the project needs to email the TA, cc'ing the instructor and all the group members, about the submission. If any group member has dispute about the submission, it should be brought into the instructor's attention within 24 hrs.
    - Each group is responsible to select the member to submit the file.
    - Failure to submit the files on time will result in the late penalty, as specified in the syllabus, for the entire group.

- Both the progress report and the final project must be submitted by their due date. Please remember that the due date and the end date are same for the progress report as well as the final project. Once the due/end date is passed, the link will be disabled and no submission will be allowed.
    - Please refer to the progress report file for further instruction.

- Final project submission instruction:
    - Each group (not each group member) needs to submit:
        - A zipped file that includes:
            - Project report
            - Source code with appropriate extension.
            - readMe.txt file
        - All source code should be submitted in the .pdf format. Please copy paste in textual form. Do not include any screenshot of the code in this pdf. This pdf is used for plagiarism purpose. It should not be inside the zipped file.

- The project report needs to include:
    - 1st page needs to include Group number and team members, both of which are available in the Canvas.
    - A final UML diagram, which should clearly specify the work done by each member in the group.
        - This could be same as that submitted in the progress report, if nothing has changed from the progress report. Otherwise, you need to make a new UML diagram showing the changes.
    - Screenshots of the outputs tested. This part should include the validation of work done by the entire group.

- The source code needs to fulfill the following requirement:
    - Each student's work needs to be in a separate file.
        - Each student needs to write the code in their own file. This will let the TA see each individual group member's work.
        - Each file should include the header information, which should include:
            - Group Number
            - Author of the source file (or the code)

        - Email
        - Date
    - There should be sufficient comments in the program.
        - Each function should be clearly described what it does along with the input arguments and return values.
  - Note that the TA will not run each individual student's work separately. The code needs to work for the entire project.

- The readMe.txt file needs to include:
  - how to run your program on CSX server,
    - Please include all the details to run your program, without assuming TA knows it.
  - on which CSX server the program is tested.

- Remember, since the TA and the instructor will have no way of knowing the project activities, we will consider what is presented in the submitted file as the final contribution. Based on that submission, we will be grading. So, if you fail to provide sufficient information, it will be difficult for us to grade fairly. It is the group's responsibility to provide as much information as possible in the final submission.