CSC227 Operating System
Course Project
(100 points)
Due Date: 27th Nov 2023 Via LMS/Blackboard

Objective:
The purpose of this project is to write a java program that simulates the CPU Scheduler (also known as Short-Term Scheduler) of an operating system. This is a group project. Each group can have up to 3 students, and all of them must be from sections the instructor teaches.

The program implements the following CPU scheduling algorithms.
(1) First-Come-First-Serve (FCFS)
(2) Shortest-Job-First (SJF)
(3) Round-Robin with time slice = 3 (RR-3)
(4) Round-Robin with time slice = 5 (RR-5)

When a process is created, a Process Control Block (PCB) is expected to be created with the following information:

- Process ID: Contains the process ID.
- Process state: Contains the state of the Process (New, Ready, Running, Waiting, Terminated)
- Burst Time (in ms)
- Memory Required in MB

The program will read process information from a file (job.txt) - this file will be provided by the instructor. File reading should be performed in an independent thread that creates the PCBs and put them in the job queue. Loading the jobs to ready queue should be performed in an independent thread that continuously checks the available space in memory to load the next job from the job queue to the ready queue. You must make sure that jobs can be loaded to ready queue only if enough space for the job is available in memory.

The main thread will perform the scheduling algorithms. *Your program should let the user to choose the scheduling algorithm to be applied.*

A sample input file of four jobs is given as follows **(**Process ID, burst time in ms, memory required in MB):
[Begin of job.txt]
Job1
5, 8, 800
Job2
3, 10, 2000
Job3
8, 3, 4000
Job4
6, 7, 1000
[End of job.txt]

Note: You can assume that
(1) There are no more than 30 jobs in the input file (job.txt).

(2) Processes arrive in the order they are read from the file, and you can assume all of them arrived at time 0.
(3) Main memory is limited to 8192 MB (Assume that single process cannot require more than the memory size).
(4) If two or more processes have the same burst time (in SJF) then the tie is broken using FCFS.

Compare the *average waiting times* and the *average turnaround times* of all jobs for each scheduling algorithm.
Output the details of each algorithm's execution. You need to show which jobs are selected at what times as well as their starting and stopping burst values. You can choose your display format, but it is recommended that you display the results in terms of a Gantt chart. You have to make sure that processes can only loaded to ready queue only if enough space for the process is available in memory.

Turn in:
(1) Soft copy of the program with proper documentation.
(2) Soft copy of the test runs using the given input files from the instructor. (You can download test cases from the Blackboard)
(3) Soft copy of the report (at least two pages) of this programming assignment including:
   a) Software and hardware tools you used
   b) How did you divide your project to subtasks and how they interact with each other?
   c) Strength and weakness of your program
   d) If you need to simulate the whole operating system (instead of just the CPU Scheduler here), explain how you can modify your program to do that.

Presentation Day:
(1) Print and bring the project cover sheet on the day of presentation (You can download the project cover sheet from the Blackboard).
(2) Your program will be evaluated based on a new input file on the day of presentation.
(3) You will be asked some questions regarding your experience on this project.
(4) Your grade will be given mainly on how you answer the instructor's questions regarding your code. In other words, group's members could receive different grades based on their contribution to the project and individual evaluation.