CS419 Project 1: Implementing CPU Scheduling Algorithms

You may work alone or with one partner (maximum group size = 2). We recommend working in pairs so you can exchange ideas and review each other's code. However, if you prefer working independently, you are welcome to do so.

Objectives:

1. Explain the basic concepts of CPU scheduling.
2. Implement scheduling algorithms including First-Come First-Served (FCFS), Shortest Job First (SJF, non-preemptive), Round Robin (RR), and Shortest Remaining Time First (SRTF).
3. Simulate workloads to measure per-process waiting times and compute average waiting time for each algorithm.
4. Compare and contrast the performance of these algorithms via simulation.

What to submit:

1. The completed **SJF.java**, **RR.java, and SRTF.java** files.
   (Do not submit any other files, as those should not be modified.)
2. A **PDF** file containing the *per-process waiting times* as well as the *average waiting tim*e for each test case, preferably in table format, as reported by your simulation.
    a. For RR, be sure to use a time quantum of *5* for *schedule1.txt* and a time quantum of *10* for *schedule2.txt*.

If working in pairs, submit *one* copy with a note that includes the names of both members.

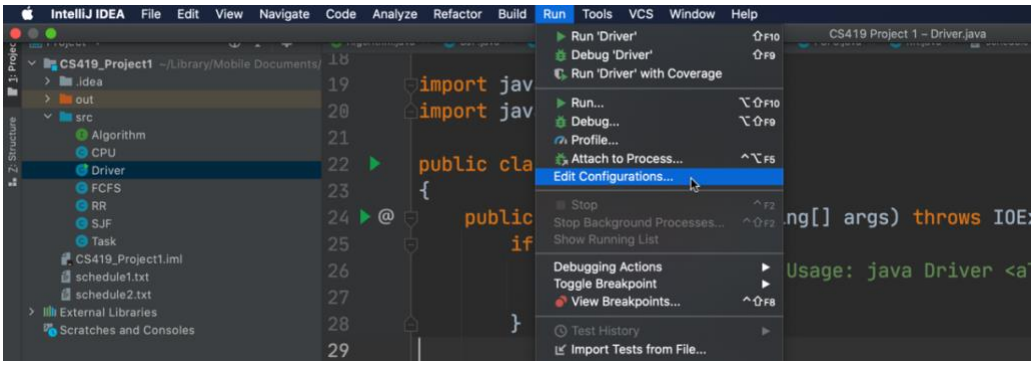
Instructions:

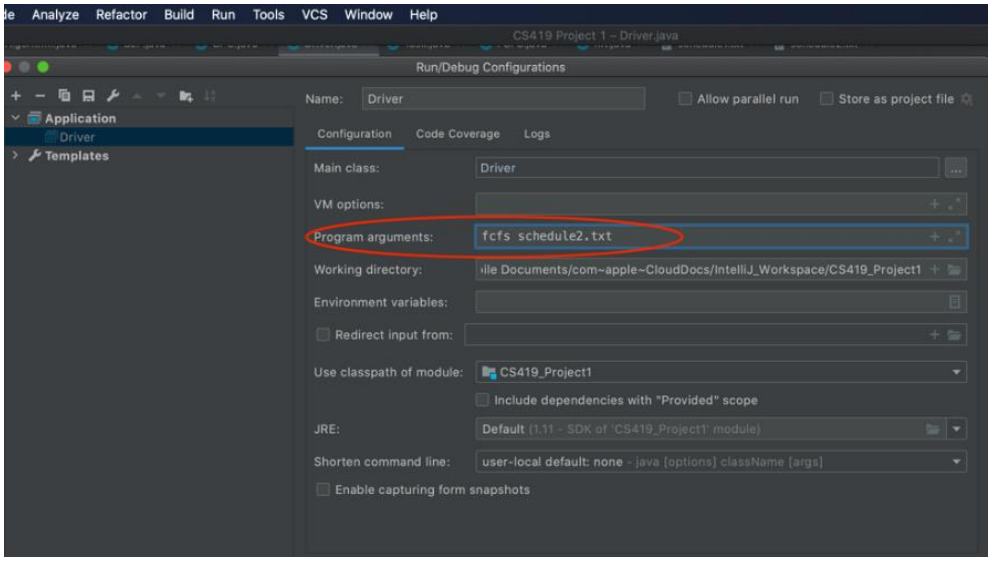
1. The implementation of FCFS is already complete and is provided to you as an example. Please first review this code and use it to guide your own implementation.
2. Your task is to implement three additional algorithms by filling in the corresponding Java files:
    a. **SJF.java**: the non-preemptive version of Shortest Job First
    b. **RR.java**: Round Robin
    c. **SRTF.java**: Shortest Remaining Time First (i.e., the preemptive version of SJF)
3. All other Java files are provided for you – they are already complete and should not be modified.
4. You are provided with two test cases, *schedule1.txt* and *schedule2.txt*, each containing a set of processes (first column), their arrival times (second column), and their CPU burst times (third column). Use both to test your implementation.
    a. When testing RR, remember to change the time quantum when you change test cases: use quantum=5 for *schedule1.txt*, and quantum=10 for *schedule2.txt*.
5. Several simplifying assumptions have been made:
    a. Processes are sorted by their arrival time in the test case file.

b. No two processes arrive at the same time.
c. Each process only has a single CPU burst and no I/O wait.
d. No context switch delay.
e. For SJF and SRTF, assume burst times are known exactly (i.e., no need to do any approximation).

6. The *main* method (in *Driver.java*) requires two arguments: the first one is the scheduling algorithm (select from: *fcfs*, *sjf*, *rr,* and *srtf,* not case-sensitive), and the second one is the name of the test case file.

   In IntelliJ, you can specify the arguments in the "Program arguments" box in the "Run/Debug Configurations" window, which can be accessed by clicking the "Edit Configurations…" option under the "Run" tab. Please refer to the screenshots below.





Grading:

This project is worth 100 points, distributed as follows:

- Implementation of scheduling algorithms (SJF, RR, SRTF): 30 points each (90 total)
- The PDF file reporting the simulation results: 10 points

Partial credit will be awarded for partially correct implementations.