Homework 5

WORKING WITH CPU SCHEDULING ALGORITHMS

Version 1.0

The objective of this assignment is to get you comfortable with different CPU scheduling algorithms. You will be implementing multiple scheduling algorithms some with support for preemption while tracking measures of scheduling effectiveness.

This assignment may be modified to clarify any questions (and the version number incremented), but the crux of the assignment and the distribution of points will not change.

Due: Wednesday, April 16th, @ 8:00 pm MT

Generative AI Use and Consequences
Use of AI tools such as ChatGPT, Claude, Github Co-Pilot, and/or their ilk to write or "improve" your code or written work at *any* stage is prohibited; this includes the ideation phase. It is your responsibility to ensure that you don't have the GitHub Co-Pilot extension installed in your IDE; assignment solutions generated by Co-Pilot aren't written by you. Turning in code or an essay written by generative AI tools will be treated as turning in work created by someone else, namely an act of plagiarism and/or cheating.

Ultimately, you will get out of the class what you put in. Simply copying and pasting code from generative AI tools is neither ethical nor does it contribute to your learning experience. There are multiple reasons why these generative AI tools are detrimental to your learning experience:

1. They rob you of the ability to think and learn the concepts for yourself. Solving problems is an essential step to gaining a solid understanding of the material.
2. You will struggle with the in-classroom quizzes and exams where you will not have access to these tools.
3. While we acknowledge that these tools are likely to become an important part of a software engineer's workflow in the future, you are much more likely to use these tools in an effective manner if you already have expertise in the relevant technical topics. Developing such expertise requires putting in the effort to learn these topics without the assistance of these tools.
4. These tools are prone to generating imperfect or even incorrect solutions, so trusting them blindly can lead to bad consequences.

1 Description of Task

In this assignment you will be implementing several basic CPU scheduling algorithms. Information about the processes that must be scheduled including the number of processes, their start times, and burst durations are provided in a separate file.

You will be implementing following CPU scheduling algorithms using C or C++ with the assumption that all jobs are CPU bound i.e. they do not block for I/O:

- First Come First Serve (FCFS) [Non-preemptive] **3 points**
- Shortest Job First (SJF) [Preemptive] **4 points**
- Priority Scheduling [Preemptive] **3 points**

You will profile the performance of these algorithms by tracking several measures of effectiveness such as average turnaround time, average waiting time, and throughput. The **turnaround time** for a process is the difference between a job's submission and completion times. The average turnaround time reports the average turnaround time for the set of processes that were scheduled. The **waiting time** for a process reflects the total amount of time spent by that process in the ready queue. The average waiting time reports the average waiting time for the set of processes that were scheduled. The **throughput** for a scheduling algorithm measures the total number of tasks processes that completed per unit of time.

2 Requirements of Task

1. You need to read the list of processes for your scheduling algorithms from a file. Every line in this file includes a record with comma separated fields. The format for this record is the following: <ProcessID>,<Arrival Time>, <Burst Duration>, <Priority>. Not all fields are used by all scheduling algorithms. For example, for FCFS you only need the process IDs, arrival times, and burst durations. All processes in your input files will be provided a unique process ID. The arrival times and burst durations are integers. Process priorities have a range of [1-50]; the lower this number, the higher the priority i.e. a process with priority=1 has a higher priority than a process with priority=2.

2. Implement FCFS (non-preemptive) and report average turnaround time, average waiting time, and throughput.

3. Implement SJF (preemptive) and report average turnaround time, average waiting time, and throughput.

4. Implement priority scheduling (preemptive) and report average turnaround time, average waiting time, and throughput.

3 Command line arguments for grading:

```
> ./Scheduler <input_filename>
```

The `input_filename` contains information about the processes that need to be scheduled.

4 What to Submit

Assignments should be submitted through Canvas. E-mailing the codes to the Professor, GTA, or the class accounts will result in an automatic 1 point deduction.

Use the CS370 *Canvas* to submit a single .zip file that contains:

- All .c and .h files related to the assignment (please document your code),
- Please use the Makefile provided on the assignments page for this assignment
- A README.txt file containing a description of each file and any information you feel the grader needs to grade your program.

Filename Convention: The main file must be named Scheduler.c; any additional .c and .h files can be named anything you want. The archive file should be named as <LastName>-<FirstName>-HW5.zip. E.g. if you are Cameron Doe and submitting for HW5, then the zip file should be named Doe-Cameron-HW5.zip.

5 Grading

This assignment would contribute a maximum of 10 points towards your final grade. The grading will also be done on a 10-point scale. The points are broken up as follows:

- 3 points for implementing FCFS
- 4 points for implementing SJF [preemptive]
- 3 points for implementing priority scheduling [preemptive]

You are required to work alone on this assignment.

6 Late Policy

All assignments are due at 8:00 PM on the due date. There is a late penalty of 10% per-day for up to a maximum of 2 days.

7 Version Change History

This section will reflect the change history for the assignment. It will list the version number, the date it was released, and the changes that were made to the preceding version. Changes to the first public release are made to clarify the assignment; the spirit or the crux of the assignment will not change.

Version	Date	Comments
1.0	3/24/2025	First public release of the assignment.