Programming Assignment 4
CPU Scheduling

Objective

This project involves implementing several different process scheduling algorithms. The scheduler will be assigned a predefined set of tasks and will schedule the tasks based on the selected scheduling algorithm. Each task is assigned a priority and CPU burst. The following scheduling algorithms will be implemented:

- First-come, first-served (FCFS), which schedules tasks in the order in which they request the CPU.
- Shortest-job-first (SJF), which schedules tasks in order of the length of the tasks' next CPU burst
- Priority scheduling, which schedules tasks based on priority.
- Round-robin (RR) scheduling, where each task is run for a time quantum (or for the remainder of its CPU burst).

Priorities range from 1 to 10, where a higher numeric value indicates a higher relative priority. For round-robin scheduling, the length of a time quantum is 10 milliseconds.

Assignment: CPU Scheduler

The implementation of this project may be completed in C/C++, and program files are provided in the source code download for the text. These supporting files read in the schedule of tasks, insert the tasks into a list, and invoke the scheduler. The schedule of tasks has the form ***[task name] [priority] [CPU burst],*** with the following example format:

```
T1, 4, 20
T2, 2, 25
T3, 3, 25
T4, 3, 15
T5, 10, 10
```

Thus, task T1 has priority 4 and a CPU burst of 20 milliseconds, and so forth. It is assumed that all tasks arrive at the same time, so your scheduler algorithms do not have to support higher-priority processes preempting processes with lower priorities. In addition, tasks do not have to be placed into a queue or list in any particular order. There are a few different strategies for organizing the list of tasks, as first presented in Section 5.1.2. One approach is to place all tasks in a single unordered list, where the strategy for task selection depends on the scheduling algorithm.

The file *driver.c* reads in the schedule of tasks, inserts each task into a linked list, and invokes the process scheduler by calling the *schedule()* function. The *schedule()* function executes each task according to the specified scheduling algorithm. Tasks selected for execution on the CPU are determined by the pick *NextTask()* function and are executed by invoking the *run()* function defined in the file. A *Makefile* is used to determine the specific scheduling algorithm that will be invoked by driver. For example, to build the FCFS scheduler, we would enter make rr and would execute the scheduler (using the schedule of tasks schedule.txt) as follows: *./fcfs* schedule.txt Refer to the README file in the source code download for further details. Before proceeding, be sure to familiarize yourself with the source code provided as well as the *Makefile*

Grading

The program will be graded on the basic functionality, error handling and how well the implementation description was followed. Be sure to name your programming project schedulers.zip. Note that documentation and style are worth 10% of the assignment's grade!