**CMP 426 (section A01) CMP 697 (section A01): Operating Systems Syllabus**

## Project 1: CPU Scheduling Simulator

**(Total Points: 120)**

In this project, you will write a program to simulate a few CPU scheduling policies we discussed in class. You will be using Java or C or C++ to implement the simulator. The simulator selects a task to run from ready queue based on the scheduling algorithm. Since the project intends to simulate a CPU scheduler, it does not require any actual process creation or execution. When a task is scheduled, the simulator will simply print out what task is selected to run at a time. It outputs the way similar to Gantt chart style.

### Project Descriptions

The selected scheduling algorithms to implement in this project are:

- First Come First Serve (FCFS)
- Round Robin (RR)
- Shortest Job First (SJF)

The above algorithms are already described in class slides, class video presentations and textbook Chapter 5.

### Process information

The process information will be read from an input file. The format is:

pid arrival_time burst_time

Example:

```
1 0 10
2 0 8
3 1 5
```

All of fields are integer type where:

- pid is a unique numeric process ID
- arrival_time is the time when the task arrives in the unit of milliseconds
- burst_time the is the CPU time requested by a task, in the unit of milliseconds

The program will be run from the command line where you provide the name of the file where the processes are stores.

The simulator first reads task information from input file and stores all data in a data structure. Then it starts simulating one scheduling algorithm in a time-driven manner. At each time unit (or slot), it adds any newly arrived task(s) into the ready queue and calls a specific scheduler algorithm in order to select appropriate task from ready queue. When a task is chosen to run, the simulator prints out a message indicating what process ID is chosen to execute for this time slot. If no task is running (i.e. empty ready queue), it prints out an “idle” message. Before advancing to the next time unit, the simulator should update all necessary changes in task and ready queue status

### Additional Outputs

- Your program will output the waiting times of all the processes and compute the average waiting time
- Your program will output the response times of all the processes and compute the average response time
- Your program will output the turn-around times of all the processes and compute the average-turnaround time

### Submissions

- All your source code must be submitted on Blackboard
- Due date; **Friday November 20th, 2020 by 11:59 pm**. **No late submission will be accepted.**