

CSSE 340 Operating Systems

Project 2: CPU Scheduling Simulator

1. Project Objectives:

This programming project is to simulate a few CPU scheduling policies discussed in the class. You will write a C/C++ program to implement a simulator with different scheduling algorithms. The simulator selects a task to run from ready queue based on the scheduling algorithm. Since the project intends to **simulate** a CPU scheduler, so 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.

2. Project Descriptions:

The selected scheduling algorithms to implement in this project are 1) First Come First Serve (**FCFS**), 2) Round Robin (**RR**), and 3) Shortest Remaining Time First (**SRTF**). The detailed algorithms are already described in class slides and textbook Chapter 5.

2.1 Task Information

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

```
pid arrival_time burst_time
```

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` is the CPU time requested by a task, in the unit of milliseconds

The time unit for `arrival_time`, `burst_time` and `interval` is millisecond.

2.2 Command-line Usage and Examples

Usage: `proj2 input_file [FCFS|RR|SRTF] [time_quantum]`

where `input_file` is the file name with task information described in section 2.1. FCFS, RR, and SRTF are names of scheduling algorithms. The `time_quantum` only applies to RR. FCFS is non-preemptive while RR and SRTF are all preemptive. **The last argument is needed only for RR.** (See following table for more examples)

| Examples | Description |
|---------------------------------|---|
| <code>proj2 input.1 FCFS</code> | FCFS scheduling with the data file "input.1" |
| <code>proj2 input.1 RR 2</code> | Simulate RR scheduling with time quantum 2 milliseconds (4th parameter is required even for quantum 1 millisecond) with the data file "input.1" |
| <code>proj2 input.1 SRTF</code> | Simulate SRTF scheduling with the data file "input.1" |

2.2 Design Hints

No sample code will be given for this project. However, here is a possible design logic you can refer. 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.

2.3 Sample Inputs and Outputs

Sample input files and expected outputs are shown in Appendix. You can use it to verify your results. Notice that these input files are not for testing and grading your program in.

3. Requirements:

The project requires to simulate FCFS, RR, and SRTF scheduling for given tasks and to compute the average waiting time, response time, turnaround time and overall CPU usage. You can find their definitions in textbook as well as in class slides.

[1] Implement scheduling algorithm for FCFS, RR, and SRTF. The program should schedule tasks and print progress of task every unit time (millisecond) as shown in sample outputs.

[2] Print statistical information. As soon as all tasks are completed, the program should compute and print 1) **average waiting time**, 2) **average response time**, 3) **average turnaround time** and 4) **overall CPU usage**.

Note: if you use static array to implement ready queue structure, you can assume the maximum queue length is 20.

4. Suggested Methodology:

- [1] Implement one scheduling algorithm at each step
- [2] You can use circular linked list as a queue structure
- [3] You can use a data structure similar to PCB for each task, though it will be much simpler

5. Submission:

- [1] All source code files
- [2] A readme file that briefly describe each file, how to compile the file(s), and how to run the file.
- [3] Due date: 10/17, Tuesday, 1:30PM
- [4] Makefile file is optional (but it is preferred).
- [5] Email your files to zhuy@seattleu.edu with subject “CSSE340 Project2”, but no zip files which will not be accepted by the SU email server.

6. Grading Policy:

- [1] If you do not have the files specified above, it will result in a score of zero.
- [2] If the program cannot be compiled and run, it will result in a score of zero.
- [3] Collaboration will result in a score of zero for all the students involved.

7. Appendix: about input file and output information by CPU scheduling simulator

for FCFS, RR, and SRTF

```
% more input.1
1  0  10
2  0  9
3  3  5
4  7  4
5  10 6
6  10 7

% proj2
Usage: proj2 input_file FCFS|RR|SRJF [quantum]

% proj2 input.1 FCFS
Scheduling algorithm: FCFS
Total 6 tasks are read from "input.1". press 'enter' to start...
=====
<system time  0> process  1 is running
<system time  1> process  1 is running
<system time  2> process  1 is running
<system time  3> process  1 is running
<system time  4> process  1 is running
<system time  5> process  1 is running
<system time  6> process  1 is running
<system time  7> process  1 is running
<system time  8> process  1 is running
<system time  9> process  1 is running
<system time 10> process  1 is finished.....
<system time 10> process  2 is running
<system time 11> process  2 is running
<system time 12> process  2 is running
<system time 13> process  2 is running
<system time 14> process  2 is running
<system time 15> process  2 is running
<system time 16> process  2 is running
<system time 17> process  2 is running
<system time 18> process  2 is running
<system time 19> process  2 is finished.....
<system time 19> process  3 is running
<system time 20> process  3 is running
<system time 21> process  3 is running
<system time 22> process  3 is running
<system time 23> process  3 is running
<system time 24> process  3 is finished.....
<system time 24> process  4 is running
<system time 25> process  4 is running
<system time 26> process  4 is running
<system time 27> process  4 is running
<system time 28> process  4 is finished.....
<system time 28> process  5 is running
<system time 29> process  5 is running
<system time 30> process  5 is running
<system time 31> process  5 is running
<system time 32> process  5 is running
<system time 33> process  5 is running
<system time 34> process  5 is finished.....
<system time 34> process  6 is running
<system time 35> process  6 is running
<system time 36> process  6 is running
<system time 37> process  6 is running
<system time 38> process  6 is running
<system time 39> process  6 is running
<system time 40> process  6 is running
<system time 41> process  6 is finished.....
<system time 41> All processes finish .....
=====
```

Avarage cpu usage : 100.00 %
Avarage waiting time : 14.17
Avarage response time : 14.17
Avarage turnaround time: 21.00
=====

% proj2 input.1 RR 2
Schdeuling algorithm: RR
Total 6 tasks are read from "input.1". press 'enter' to start...
=====

```
<system time 0> process 1 is running
<system time 1> process 1 is running
<system time 2> process 2 is running
<system time 3> process 2 is running
<system time 4> process 1 is running
<system time 5> process 1 is running
<system time 6> process 3 is running
<system time 7> process 3 is running
<system time 8> process 2 is running
<system time 9> process 2 is running
<system time 10> process 1 is running
<system time 11> process 1 is running
<system time 12> process 4 is running
<system time 13> process 4 is running
<system time 14> process 3 is running
<system time 15> process 3 is running
<system time 16> process 5 is running
<system time 17> process 5 is running
<system time 18> process 6 is running
<system time 19> process 6 is running
<system time 20> process 2 is running
<system time 21> process 2 is running
<system time 22> process 1 is running
<system time 23> process 1 is running
<system time 24> process 4 is running
<system time 25> process 4 is running
<system time 26> process 4 is finished.....
<system time 26> process 3 is running
<system time 27> process 3 is finished.....
<system time 27> process 5 is running
<system time 28> process 5 is running
<system time 29> process 6 is running
<system time 30> process 6 is running
<system time 31> process 2 is running
<system time 32> process 2 is running
<system time 33> process 1 is running
<system time 34> process 1 is running
<system time 35> process 1 is finished.....
<system time 35> process 5 is running
<system time 36> process 5 is running
<system time 37> process 5 is finished.....
<system time 37> process 6 is running
<system time 38> process 6 is running
<system time 39> process 2 is running
<system time 40> process 2 is finished.....
<system time 40> process 6 is running
<system time 41> process 6 is finished.....
<system time 41> All processes finish .....
```

=====

Avarage cpu usage : 100.00 %
Avarage waiting time : 22.50
Avarage response time : 4.00
Avarage turnaround time: 29.33
=====

% proj2 input.1 SRTF
Schdeuling algorithm: SRTF
Total 6 tasks are read from "input.1". press 'enter' to start...
=====

```

<system time 0> process 2 is running
<system time 1> process 2 is running
<system time 2> process 2 is running
<system time 3> process 3 is running
<system time 4> process 3 is running
<system time 5> process 3 is running
<system time 6> process 3 is running
<system time 7> process 3 is running
<system time 8> process 3 is finished.....
<system time 8> process 4 is running
<system time 9> process 4 is running
<system time 10> process 4 is running
<system time 11> process 4 is running
<system time 12> process 4 is finished.....
<system time 12> process 2 is running
<system time 13> process 2 is running
<system time 14> process 2 is running
<system time 15> process 2 is running
<system time 16> process 2 is running
<system time 17> process 2 is running
<system time 18> process 2 is finished.....
<system time 18> process 5 is running
<system time 19> process 5 is running
<system time 20> process 5 is running
<system time 21> process 5 is running
<system time 22> process 5 is running
<system time 23> process 5 is running
<system time 24> process 5 is finished.....
<system time 24> process 6 is running
<system time 25> process 6 is running
<system time 26> process 6 is running
<system time 27> process 6 is running
<system time 28> process 6 is running
<system time 29> process 6 is running
<system time 30> process 6 is running
<system time 31> process 6 is finished.....
<system time 31> process 1 is running
<system time 32> process 1 is running
<system time 33> process 1 is running
<system time 34> process 1 is running
<system time 35> process 1 is running
<system time 36> process 1 is running
<system time 37> process 1 is running
<system time 38> process 1 is running
<system time 39> process 1 is running
<system time 40> process 1 is running
<system time 41> process 1 is finished.....
<system time 41> All processes finish .....

```

```

=====
Avarage cpu usage      : 100.00 %
Avarage waiting time   : 10.50
Avarage response time  : 9.00
Avarage turnaround time: 17.33
=====

```