**Final Project**

For this assignment please submit a single **zip** file. The zip file will contain your code and a **README** **pdf** document with instructions on how to run your code. Include **screenshots** of your program running on five different time quantum in the zip file as well.

Your objective is to create a program that simulates **Round Robin CPU scheduling**. You could use the virtual machine from the first assignment to test your program to make sure it will run.

**Program Requirements:**

The program you will create is used to generate a schedule of processes. The processes are stored in a plaintext file. The program accepts two parameters. The first parameter is the path of the processes file. The second parameter is the time quantum. (More on that on the next page).

Your simulation should include the following:

|  |  |
| --- | --- |
| Clock | timestamps all events for processes, such as creation time, completion time, etc. |
| Process Creator | creates processes at arrival time |
| CPU | runs processes for a time slice (time quantum) |
| Queue | FIFO ready queue used by both the process creator and CPU |
| Process Arrival Time | arrival time of new processes into the ready queue |
| Process Service Time | amount of time required by the processes to complete execution |
| Time Quantum | time each process can spend in the CPU, before it is removed |
| Context Switch | number of times a process is switched |

Your program should also print out the following performance evaluation criteria:

|  |  |
| --- | --- |
| CPU Utilization |  |
| Throughput |  |
| Average Waiting Time |  |
| Average Turnaround Time |  |

Include a **README pdf** document as your project report. Your report should explain CPU scheduling and how Round Robin works. Your report should detail your project implementation and how to analyze the program output. You must include the resulting output of your program using ***five*** *different time quantum* on the same process list. You can either use screenshots or simply copy the output into your report. The analysis of the results of the five runtimes should exhibit the performance of the algorithm based on the size of the time quantum and how this size impacts the outcome of the scheduling.

**Sample Input:**

Your program will be tested with an input file that has a certain format such as a csv (comma space value) format (if applicable to your computing environment) or any other format that resembles a table. The three columns are Process ID, Arrival Time and Burst Time. Each row is a process. The following is an example using a CSV file:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***CSV File***   |  | | --- | | pid,arrive,burst  1,0,5  2,1,7  3,0,2  4,2,6 | |  | ***Meaning***   |  |  |  | | --- | --- | --- | | pid | arrive | burst | | 1 | 0 | 5 | | 2 | 1 | 7 | | 3 | 0 | 2 | | 4 | 2 | 6 | |
|

**Program Execution:**

Your program accepts two parameters. The first is that path to the processes text file. The second parameter is the time quantum to be used in the simulation. For example, if your program was written in java, the following could be how to test it.

|  |
| --- |
| $ javac target.java |
| $ java target processes.csv 2 |

**Suggestions:**

1. Research the Round Robin CPU scheduling algorithm.
2. Understand goals of scheduling:

|  |  |
| --- | --- |
| CPU Utilization | Maximize CPU utilization. Keep the CPU 100% busy. |
| CPU Throughput | Maximize the number of jobs processed. |
| Turnaround Time | Minimize the total time to complete processes. |
| Waiting Time | Minimize the total time processes wait in the ready queue. |
| Response Time | Minimize the time from when a request is submitted until the first response (not output) is produced. |

1. Understand the requirements and plan steps involved to meet the requirements.
2. Set up the environment for the scheduler.
3. Define classes/subclasses used by the scheduler (CPU, clock, process, process-creator, simulator, etc.), FIFO scheduler, and all the parameters involved.
4. Define queues (i.e., ready queue/FIFO and waiting queue) used by the scheduler, the data structure, and mechanisms used for each queue to determine when and which queue a process should enter, and when to be removed to resume execution or be terminated.
5. Keep track of context-switch time.
6. Provide input to your simulator (time quantum, process arrival time, number of process, burst time of each process) process-creator.
7. Output the sequence of events and show how the algorithm is executed and the performance of your scheduling algorithm for the same set of processes and their associated burst time, but different time quantum, in terms of the turnaround time, waiting time, CPU utilization, and throughput.
8. Empirical evaluation: Full accounting and statistics after the simulation completes. Create processes with different arrival time and burst time. Run the scheduling algorithm on these processes (i.e., the same set of processes), with five different Time Quantum. Create metrics of the results and get the average of these metrics.
9. Analyze the result.

**EXTRA CREDIT – Students who wish to work on combination of Priority and Round Robin Algorithms, will receive extra 5 points.**

**Some helpful info**

Watch the following YouTube video:

<https://www.youtube.com/watch?v=Sa600YsU16U>

**Arrival time:** the time at which process enters the Ready queue

**Burst time:** time required by a process to get executed on CPU

**Completion time:** the time at which process completes its execution

**Turnaround time:** Completion time - Arrival time

**Waiting time:** Turnaround time – Burst time

**Response time:** the time at which a process gets cpu first time – arrival time

The calculation for cpu utilization should just be: **1-(idle time / total execution time )**. A simpler way to find the idle time could be to instead keep track of the number of context switches and multiply the context switching time by that. This new formula would then look like: **1 - ((context switch time \* context switches) / total execution time)**.

**Students are expected to comply with NYIT’s Academic Integrity Policy and complete this assignment on their own.**

**Good Luck!**