**CS 3305A: Operating Systems**
**Department of Computer Science**
**Western University**
**Assignment 2**
**Fall 2017**
**Due Date: Nov 5th 2017**

**Purpose:**

The goals of this assignment are the following:

- Get hands-on experience with the process/thread related *function calls* and CPU *scheduling algorithms.*
- Gain more experience with the C programming language from an OS's *process/thread* and *CPU scheduling* perspective.

**Part I: Process vs Thread (**20 points)

You will be writing a C program to test the data sharing ability of *thread* and *process*. Your C program will do the following:

1. Your *parent* program will have three variables: *int x,y,z;* which to be initialized as 10, 20, and 0, respectively.

2. *parent* creating *child*: *parent* will create a *child* by *fork()* which will perform $z = x+y$ (i.e., add x and y and store the results in z). *parent* will wait for *child* to complete before parent proceeds. Upon completion of *child*, *parent* will print out the value of z. (8 points)

3. *parent* creating *thread*: After (2) is completed, *parent* process will now create a *thread* by *pthread_create()* which will do the exact same task done by *child* above (i.e., z = x+y). *parent* will wait for its *thread* to complete before *parent* proceeds. Upon completion of the *thread*, *parent* will print out the value of z. (12 points)

**Part II: Performance Evaluation of CPU Scheduling Algorithms** (80 points)

You will be developing a *Multilevel Queue CPU Scheduling Algorithm* using C programming language. An input file is provided here (see below part II_d) which must be used to develop the *Multilevel Queue CPU Scheduling Algorithm*.

**Part II_a: CPU Scheduling Environment Initialization (15 points)**

Your C program will perform the following tasks based on the given input file *multilevel_queue_CPU_scheduling_input_file.txt*:

1. Create the number of ready queues as stated in the given input file (3 points)
2. Assign the priority level for each of the ready queues (as per the input file spec) (3 points)

3. Assign each ready queue with a specific CPU scheduling algorithm (as per the input file spec, such as FCFS, SJF, RR etc.) (3 points)
4. Assign time quantum for (as per input file spec) for RR algorithm (3 points)
5. Create all the processes for each of the ready queues based on the input file specification (such as CPU burst time, arrival order etc.) (3 points)

**Part II_b: Scheduling Algorithm Execution (45 points)**

Your C program will perform the following tasks in order based on the given input file:

1. CPU scheduler will execute the top most priority queue (15 points)
2. CPU scheduler will execute the second priority queue (15 points)
3. CPU scheduler will execute the third priority queue (15 points)

**Part II_c: Results (20 points)**

Once the execution of all the three ready queues is completed, your C program should provide the following output (both showing on the screen and writing into a text file "multilevel_cpu_output.txt". Note that the order of the output below is important as your program should process the Queues based on their priority levels):

1. Order of the processes selected by CPU in Ready Queue 2 (2 points)
2. Turnaroud time for each process in Ready Queue 2 (2 points)
3. Order of the processes selected by CPU in Ready Queue 1 (2 points)
4. Individual waiting time for each process in Ready Queue 1(2 points)
5. Average waiting time for Ready Queue 1 (2 points)
6. Order of the processes selected by CPU in Ready Queue 3 (2 points)
7. Individual waiting time for each process in Ready Queue 3 (2 points)
8. Average waiting time for Ready Queue 3 (2 points)
9. Total time taken to process all three queues (4 points)

**Part II_d: Input File**

q 1 pr 2 fcfs p1 10 p2 5 p3 7 p4 20 p5 17 p6 9 p7 3 p8 11 p9 15 p10 1
q 2 pr 3 rr tq 5 p1 10 p2 5 p3 7 p4 20 p5 17 p6 9 p7 3 p8 11 p9 15 p10 1
q 3 pr 1 sjf p1 10 p2 5 p3 7 p4 20 p5 17 p6 9 p7 3 p8 11 p9 15 p10 1

Symbols used in the above input file:

---

q: Ready queue
pr: priority level, 1 lowest, 3 highest
fcfs: first come first serve
sjf: shortest job first
rr: round robin
tq: time quantum

Example:
---

q 1 pr 2 fcfs p1 10 p2 5 p3 7 p4 20 p5 17 p6 9 p7 3 p8 11 p9 15 p10 1

Ready Queue q1 has a priority level of 2. FCFS scheduling algorithm will be used for q1. Ready Queue q1 has a total of ten processes namely p1, p2, p3,p4, p5, p6, p7, p8, p9, and p10. The sequence of these processes represent their arrival order. for example, p1 arrives at the first and p10 arrives at the last in this list of processes. In "px y" format y refers to the CPU burst time for px.

**Assignment related technical resources**

Please visit the course website for specific technical instructions and relevant materials. Also, consult TAs, and Instructor for any question you may have regarding this assignment.