COMP 346 – Fall 2017
Theory Assignment 4

Answer all questions

Due Date	**By 11:59pm Friday, December 1, 2017**
Format	Assignments must be typed and submitted online to Moodle system. **Scanned hand-written assignments will be discarded**.
Late Submission:	none accepted
Purpose:	The purpose of this assignment is to help you learn the overview of computer operating systems and Input and output mechanisms.
CEAB/CIPS Attributes:	Design/Problem analysis/Communication Skills

Question # 1

[10 marks] Consider the following preemptive priority-scheduling algorithm based on dynamically changing priorities. Larger priority numbers indicate higher priority (e.g., priority 3 is higher priority than priority 1; priority 0 is higher priority than priority -1, etc.). When a process is waiting for the CPU in the ready queue, its priority changes at the rate of α per unit of time; when it is running, its priority changes at the rate of β per unit of time. All processes are assigned priority 0 when they enter the ready queue. Answer the following questions:

(a) What is the scheduling algorithm that results when $\beta > \alpha > 0$? Explain.

(b) What is the scheduling algorithm that results when $\alpha < \beta < 0$? Explain.

Question # 2

[10 marks] Somebody proposed a CPU scheduling algorithm that favors those processes that used the least CPU time in the recent past. For calculating the CPU time of a process in the recent past, a time window of size τ is maintained and the recent CPU time used by a process at time T is calculated as the sum of the CPU times used by the process between time T and T- τ. It is argued that this particular scheduling algorithm (a) will favor I/O-bound processes, and (b) will not permanently starve CPU-bound processes. Do you agree/disagree with (a) and (b)? Explain.

Question # 3

[10 marks] Consider a variant of the round robin (RR) scheduling algorithm in which the entries in the ready queue are pointers to the Process Control Blocks (PCBs), rather than the PCBs. A malicious user wants to take advantage and somehow, through a security loophole in the system, manages to put two pointers to the PCB of his/her process with the intention that it can run twice as much. Explain what serious consequence(s) it could have if the (malicious) intention goes undetected by the OS.

Question # 4

4 [10 marks] Consider the version of the dining philosopher's problem in which the chopsticks are placed in the center of the table and any two of them can be used by a philosopher. Assume that requests for chopsticks are made one chopstick at a time. Describe a simple rule for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers.

<u>Question # 5</u>

[10 marks] Consider Q.4 above. Assume now that each philosopher needs three chopsticks to eat. Resource requests are still issued one at a time. Describe some simple rules for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers.

<u>Question #6</u>

[10 marks] Consider a system consisting of m resources of the same type being shared by n processes. A process can request or release only one resource at a time. Show that the system is deadlock free if the following two conditions hold:

1. The maximum need of each process is between 1 resource and m resources.
2. The sum of all maximum needs is less than $m + n$.