COMP 346 – Winter 2017
Theory Assignment 1

Due Date & Submission Format: *Please see course outline.*

Answer all questions

Question # 1

I. What is an operating system? What are the main purposes of an operating system?

II. Define the essential properties of the following types of operating systems:

- Batch
- Time sharing
- Dedicated
- Real time
- Multiprogramming

III. Under what circumstances would a user be better of using a time-sharing system rather than a PC or single-user workstation?

Question # 2

Consider a computer system with a single-core processor. There are two processes to run in the system: P_1 and P_2. Process P_1 has a life cycle as follows: CPU burst time of 15 units, followed by I/O burst time of minimum 10 units, followed by CPU burst time of 10 units. Process P_2 has the following life cycle: CPU burst time of 10 units, followed by I/O burst time of minimum 5 units, followed by CPU burst time of 15 units. Now answer the following questions:

a) Considering a *single programmed* operating system, what is the minimal total time required to complete executions of the two processes? You should explain your answer with a diagram.

b) Now considering a *multiprogrammed* operating system, what is the minimal total time required to complete executions of the two processes? You should explain your answer with a diagram.

c) *Throughput* is defined as the number of processes (tasks) completed per unit time. Following this definition, calculate the throughputs for parts a) and b) above. How does multiprogramming affect throughput? Explain your answer.

Question # 3

I. What is the performance advantage in having device drivers and devices synchronize by means of device interrupts, rather than by polling (i.e., device driver keeps on polling the device to see if a specific event has occurred)? Under what circumstances can polling be advantageous over interrupts?

II. Is it possible to use a DMA controller if the system does not support interrupts? Explain why.

IV. The procedure *ContextSwitch* is called whenever there is a switch in context from a running program A to another program B. The procedure is a straightforward assembly language routine that saves and restores registers, and must be atomic. Something disastrous can happen if the routine *ContextSwitch* is not atomic.

(a) Explain why *ContextSwitch* must be atomic, possibly with an example.
(b) Explain how the atomicity can be achieved in practice.

Question # 4

I. If a user program needs to perform I/O, it needs to trap the OS via a system call that transfers control to the kernel. The kernel performs I/O on behalf of the user program. However, systems calls have added overheads, which can slow down the entire system. In that case, why not let user processes perform I/O directly, without going through the kernel?

II. Consider a computer running in the user mode. It will switch to the monitor mode whenever an interrupt or trap occurs, jumping to the address determined from the interrupt vector.

(a) A smart, but malicious, user took advantage of a certain serious loophole in the computer's protection mechanism, by which he could make run his own user program in the monitor mode! This can cause disastrous effects. What could have he possibly done to achieve this? What disastrous effects could it cause?
(b) Suggest a remedy for the loophole.

Question # 5

Suppose that a multiprogrammed system has a load of N processes with individual execution times of $t_1, t_2, \ldots, t_N$. Answer the following questions:

a) How would it be possible that the time to complete the N processes could be as small as: *maximum* $(t_1, t_2, \ldots, t_N)$?
b) How would it be possible that the total execution time, $T > t_1 + t_2 + \ldots + t_N$? In other words, what would cause the total execution time to exceed the sum of individual process execution times?

Question # 6

Which of the following instructions should be privileged? Explain why.

(i) Read the system clock,
(ii) Clear memory,
(iii) Reading from user space
(iv) Writing to user space

(v) Copy from one register to another
(vi) Turn off interrupts, and
(vii) Switch from user to monitor mode.

Question # 7

Assume you are given the responsibility to design two OS systems, a Network Operating System and a Distributed Operating System. Indicate the primary differences between these two systems. Additionally, you need to indicate if there any possible common routines between these systems? If yes, indicate some of these routines. If no, explain why common routines between these two particular systems do not make sense.