COMP 346 – Fall 2020
Assignment 1

Name: Hualin Bai

ID: 40053833

Written Questions (50 marks):

Question # 1:

(1)

Operating system is a program that acts as an intermediary between a user of a computer and the computer hardware.

The purpose of an OS is to provide an environment in which a user executes programs and make solving problems much easier and convenient. Besides, computer system and hardware will be used in an efficient and convenient manner.

(2)

- **Batch**: Similar jobs grouped together, one job at a time, job scheduling by OS.
- **Time sharing**: (multitasking) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing.
- **Dedicated**: Specific OS to the environment, such as workstations have dedicated resources but frequently use shared resources from servers.
- **Real time**: Real-time OS has well-defined fixed time constrains, which includes hard and soft real-time. The system must respond to inputs/commands within a fixed amount of time to ensure correct performance.
- **Multiprogramming**: Sharing the processor, when two or more programs reside in memory at the same time. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization via job scheduling so that the CPU always has one to execute.

(3)

Time-sharing is better than a PC or single-user workstation when there are few other users, the task is large, and the hardware is fast, since the full power of the system can be brought to bear on the user's problem. Another case is when lots of other users need resources concurrently.

Question # 2

	CPU	I/O	CPU
P1	15	10	10
P2	10	5	15

(a) single programmed OS

P1			P2		
15	10	10	10	5	15

As seen the table, the single programmed OS execute 2 processes one by one.

Thus, the minimal total time = 15 + 10 + 10 + 10 + 5 + 15 = 65

(b) multi-programmed OS

P1: CPU time	P2: CPU time	P1: CPU time	P2: CPU time
15	10	10	15

Thus, the minimal total time = 15 + 10 + 10 + 15 = 50

(c)

Throughput A = 2 tasks / 65 units = 0.03076923

Throughput B = 2 tasks / 50 units = 0.04

Therefore, multiprogramming will increase the throughput.

Question # 3:

(1)

The advantage of device interrupts (vectored interrupt system) is that it can save resources (e.g. CPU cycles) because it only reacts when something has acutely happened.

Polling is basically a busy loop or continuous checking. Thus, polling is reasonable if the device is fast and I/O is frequent, or if most of jobs are I/O bound jobs, or if system has no real time critical jobs to be done.

(2)

DMA is a special purpose processor for a device that does large transfers such as disk drives bypassing CPU to transfer data directly between I/O device and memory.

It is possible to use a DMA controller when the system does not support interrupts, but the CPU will keep checking the DMA controller, thus it wastes resources.

(3)

(a) Context switch must be atomic to ensure consistency. As the context switch requires saving the address of the old process to registers and loading the address of the next process, if it is not atomic then the addresses or state of the process may be incorrectly saved which results in inconsistency.

For example, the save operation requires several steps to save value of CPU registers, process state, memory management information, etc.

(b) The atomicity can be achieved by synchronization. For example, making the save method and probably the load method "synchronized", in order to make their steps execute in one block.

Question # 4:

(1) Letting user processes perform I/O directly will give the process power over CPU and main memory, which will cause a high risk.

(2)

(a) This could possibly be done by passing system calls or message passing. It gives the user/process access to other protected system files and hardware.

(b) I suggest using system calls or message passing for the loophole about switch the user and monitor mode.

Question # 5:

(a) When the N processes run in parallel, and context switches require little time.

(b) When the N processes run linearly, and context switches require much time.

Question # 6:

(i) **Read the system clock**: Unprivileged, since all processes should be able to read. The Instructions that can run only in User Mode are called Non-Privileged Instructions.

(ii) **Clear memory**: Privileged. Clearing memory may prove fatal, so this operating must be protected. The Instructions that can run only in Kernel Mode are called Privileged Instructions.

(iii) **Reading from user space**: Unprivileged, since it only affects user space.

(iv) **Writing to user space**: Unprivileged, since it only affects user space.

(v) **Copy from one register to another**: Unprivileged, since it also only affects user space.

(vi) **Turn off interrupts**: Privileged, since it can slow down the CPU and other processes. Any instruction which can modify the contents of the Timer is a Privileged Instruction.

(vii) **Switch from user to monitor mode**: Privileged, since it gives access to all memory and kernel.

Question # 7:

A network OS is multiple users operating together through their own OS on a network.

A distributed OS is multiple users running the same OS with several processors with independent memories and clocks.

Also, Distributed OS needs a high degree of transparency, but Network OS needs low. They have different ways of communication and resource management. Network OS has scalability, while Distributed OS is low.

Distributed OS needs more RAM and High-speed processor than Network OS. Network OS and Distributed OS have a common hardware base, but their software is different.

Network OS runs the software that associated network which can make users work together in the network, however Distributed OS runs in an ordinary centralized OS based on multiple independent CPUs.