Mid Semester Assignment
Spring 2020
Subject: Operating System Concepts

ANSWER SHEET

NAME   HAMZA IQBAL

ID#   14784

DEPARTEMENT   BS (SE)

PAPER   OPERATING SYSTEM

INSTRUCTOE: SIR DAUD KHAN

Section A

Question No: 1 (M - 1)

The hardware mechanism that enables a device to notify CPU is called an ----interurupt----
► Interrupt✓
► Signal
► Trap
► Process

Question No: 2 (M - 1)
The section of the process control block comprises of page and segment tables
► Memory related information✓
► Accounting information
► Register information
► Scheduling information

Question No: 3 (M - 1)
The ----wait----- system call suspends the calling process.
► Fork
► wait✓
► exec
► exit

Question No: 4 (M - 1)
In ---Asymmetric----addressing, the recipient is not required to name the sender.
► Symmetric
► Asymmetric✓
► Both symmetric and asymmetric
► None of the given options

Question No: 5 (M - 1)
---PS----- command gives a snapshot of the current processes.
► PS✓
► top
► who
► ls

Question No: 6 (M - 1)
-----fg------command to resume the execution of a suspended job in the foreground
► fg✓
► bg
► jobs
► kill

Question No: 7 (M - 1)
You can use the -----jobs-------- command to display the status of suspended and background processes
► fg
► bg
► jobs✓
► kill

Question No: 8 (M - 1)
You can terminate a foreground process by pressing ------<Ctrl-C>------
► <Ctrl-A>
► <Ctrl-C>✓
► <Ctrl-Z>
► None of the given options

Question No: 9 (M - 1)
A time sharing system is
► Multi-tasking
► Interactive
► Multi user
► All of these✓

Question No: 10 (M - 1)
The main characteristic of a Real time system is
► Efficiency✓
► Large Virtual Memory
► Large secondary storage device
► Usability

Question No: 11 (M - 1)
Shared libraries and kernel modules are stored in _____/lib_____ directory

► /bin
► /dev
► /boot
► /lib✓

Question No: 12 (M - 1)
___Long term_____ scheduler selects the process from the job pool and put them in main memory.

► Long term✓
► Short term
► Medium term
► Swapper

Question No: 13 (M - 1)
In indirect inter process communication, a sender _do not__ mention the name of the recipient.

► do
► do not

Question No: 14 (M - 1)
A _____semaphore_____ is an integer variable that, apart from initialization is accessible only through two standard atomic operations: wait and signal.
► Semaphore ✓
► Monitor
► Critical region
► Critical section

Question No: 15 (M - 1)
A semaphore that cause Busy-Waiting is termed as ___spinlock___.

► Spinlock✓
► Monitor
► Critical region
► Critical section

Question No: 16 (M - 1)
The execution of critical sections must NOT be mutually exclusive

► True
► False✓

Question No: 17 (M - 1)
The performance of Round Robin algorithm does NOT depends heavily on the size of the time quantum.

► True
► False✓

Question No: 18 (M - 1)
The following requirement for solving critical section problem is known as _____.
"There exists a bound on the number of times that other processes are allowed to enter their critical sections after a process has made a request to enter its critical section and before that request is granted."

► Progress
► Bounded Waiting✓
► Mutual Exclusion
► Critical Region

Question No: 19 (M - 1)
The critical section problem can be solved by the following except

► Software based solution
► Firmware based solution
► Operating system based solution
► Hardware based solution

Question No: 20 (M - 1)
___*Medium term scheduler*_____ *is also called Swapper.*

► Swap space
► Medium term scheduler✓
► Short term scheduler
► Long term scheduler

Question No: 21 (M - 2)
Write the formula/ procedure for calculating the waiting time in preemptive Shortest Job First scheduling?

➡*ANSWER :*

Pre-emptive Shortest Job First

In Preemptive Shortest Job First Scheduling, jobs are put into ready queue as they arrive, but as a process with **short burst time** arrives, the existing process is preempted or removed from execution, and the shorter job is executed first.

PROCESS	BURST TIME	ARRIVAL TIME
P1	21	0
P2	3	1
P3	6	2
P4	2	3

The GANTT chart for Preemptive Shortest Job First Scheduling will be,

P1	P2	P4	P2	P3	P1

0   1   3   5   6   12   32
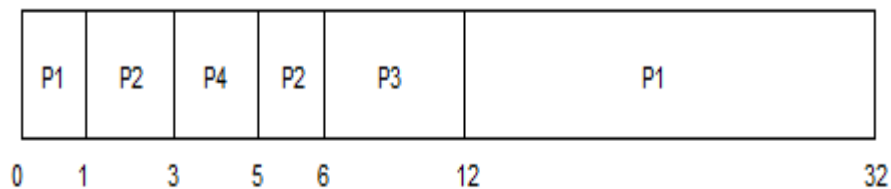
The average waiting time will be, ((5-3) + (6-2) + (12-1))/4 = 4.25 ms

The average waiting time for preemptive shortest job first scheduling is less than both, non-preemptive SJF scheduling and FCFS scheduling.

Question No: 22 (M - 3)
If a process exits and there are still threads of that process running, will they continue to run?

➡ANSWER:

Yes, because more state information must be saved to switch from one **process** to another. ... **If a process exits and there are still threads** of that **running**, **will they continue to run**? Yes. **When a process exits**, it takes everything with it—the KLTs, the process structure, the memory space, everything—including threads.

Process Control Information: Thread control block should handle scheduling and State Information. Data Structuring should also be handled by thread control block (mostly). Memory Management and Resource ownership should belong to process control block. Privileges and Inter-process communications can probably be divided between the 2 process blocks depending on the system.

Question No: 23 (M - 5)

Considering the Resource sharing feature of thread, what do you think is 'resource sharing' an advantage of a thread or disadvantage of a thread. Explain yours answer briefly?

➡ANSWER:

The process model that has been discussed in previous tutorials described that a process was an executable program that is having a single thread of control. The majority of the modern operating systems now offer features enabling a process for containing multiple threads of control. In this tutorial, there are many concepts associated with multithreaded computer structures. There are many issues related to multithreaded programming and how it brings effect on the design of any operating systems. Then you will learn about how the Windows XP and Linux OS maintain threads at the kernel level.

A thread is a stream of execution throughout the process code having its program counter which keeps track of lists of instruction to execute next, system registers which bind its current working variables. Threads are also termed as lightweight process. A thread uses parallelism which provides a way to improve application performance.

Resource sharing: Mostly **threads** share the memory and the **resources** of any process to which **they** fit in. The **advantage** of **sharing** code is that it allows any application to have multiple different **threads** of activity inside the same address space.

➔<mark>Benefits of threads:</mark>

The advantages of multithreaded programming can be categorized into four major headings -

- **Responsiveness**: Multithreading is an interactive concept for an application which may allow a program to continue running even when a part of it is blocked or is carrying a lengthy operation, which increases responsiveness to the user.
- **Resource sharing:** Mostly threads share the memory and the resources of any process to which they fit in. The advantage of sharing code is that it allows any application to have multiple different threads of activity inside the same address space.
- **Economy:** In OS, allocation of memory and resources for process creation seems costly. Because threads can distribute resources of any process to which they belong, it became more economical to create and develop context-switch threads.
- **Utilization of multiprocessor architectures**: The advantages of multithreading can be greatly amplified in a multiprocessor architecture, where there exist threads which may run in parallel on diverse processors.

➔**MULTITHREADING VS MODEL:**

All the threads must have a relationship between them (i.e., user threads and kernel threads). Here is a list which tells the three common ways of establishing this relationship.

- Many-to-One Model: In the many-to-one model plots several user-level threads to a single kernel thread.
- One-to-One Model: In the one-to-one model maps every particular user thread to a kernel thread and provides more concurrency compare to many-to-one model.
- Many-to-Many Model: In the many-to-many model, many user-level threads get mapped to a smaller or equal quantity of kernel threads. The number of kernel threads might be exact to either a particular application or to a particular machine.

THANK YOU SIR