Operating systems

Question & Answer (Unit 1)

1. *Describe the differences between symmetric and asymmetric multiprocessing. What are three advantages and one disadvantage of multiprocessor systems?*
<u>Answer:</u> Symmetric multiprocessing treats all processors as equals, and I/O can be processed on any CPU. Asymmetric multiprocessing has one master CPU and the remainder CPUs are slaves. The master distributes tasks among the slaves, and I/O is usually done by the master only. Multiprocessors can save money by not duplicating power supplies, housings, and peripherals. They can execute programs more quickly and can have increased reliability. They are also more complex in both hardware and software than uniprocessor systems.

2. *What is the purpose of interrupts? What are the differences between a trap and an interrupt? Can traps be generated intentionally by a user program? If so, for what purpose?*

3. *Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU's execution load.*

4.*Describe some of the challenges of designing operating systems for mobile devices compared with designing operating systems for traditional PCs.*

5. *The services and functions provided by an operating system can be divided into two main categories. Briefly describe the two categories, and discuss how they differ.*

6 *What are the two models of interprocess communication? What are the strengths and weaknesses of the two approaches?*
<u>Answer:</u> The two models of interprocess communication are message-passing model and the shared-memory model. Message passing is useful for exchanging smaller amounts of data, because no conflicts need be avoided. It is also easier to implement than is shared memory for intercomputer communication. Shared memory allows maximum speed and convenience of communication, since it can be done at memory transfer speeds when it takes place within a computer. However, this method compromises on protection and synchronization between the processes sharing memory.

7 *Why is the separation of mechanism and policy desirable?*

8. *What is the main advantage of the microkernel approach to system design? How do user programs and system services interact in a microkernel architecture? What are the disadvantages of using the microkernel approach?*

9.Describe the differences among short-term, medium-term, and long-term scheduling.

***10.** Describe the actions taken by a kernel to context-switch between processes.*
<u>Answer:</u> In general, the operating system must save the state of the currently running process and restore the state of the process scheduled to be run next. Saving the state of a process typically includes the values of all the CPU registers in addition to memory allocation. Context switches must also perform many architecture-specific operations, including flushing data and instruction caches.

***11.** Give an example of a situation in which ordinary pipes are more suitable than named pipes and an example of a situation in which named pipes are more suitable than ordinary pipes.*

***12.** What are the benefits and the disadvantages of each of the following? Consider both the system level and the programmer level.*
a. Synchronous and asynchronous communication
b. Automatic and explicit buffering
c. Send by copy and send by reference
d. Fixed-sized and variable-sized messages

***13** Why is it important for the scheduler to distinguish I/O-bound programs from CPU-bound programs?*

***14.** One technique for implementing **lottery scheduling** works by assigning processes lottery tickets, which are used for allocating CPU time. Whenever a scheduling decision has to be made, a lottery ticket is chosen at random, and the process holding that ticket gets the CPU. The BTV operating system implements lottery scheduling by holding a lottery 50 times each second, with each lottery winner getting 20 milliseconds of CPU time (20 milliseconds × 50 = 1 second). Describe how the BTV scheduler can ensure that higher-priority threads receive more attention from the CPU than lower-priority threads.*

15. *Consider the exponential average formula used to predict the length of the next CPU burst. What are the implications of assigning the following values to the parameters used by the algorithm?*
a. $\alpha = 0$ and $\tau_0 = 100$ milliseconds
b. $\alpha = 0.99$ and $\tau_0 = 10$ milliseconds
<u>Answer:</u> When $\alpha = 0$ and $\tau_0 = 100$ milliseconds, the formula always makes a prediction of 100 milliseconds for the next CPU burst. When $\alpha = 0.99$ and $\tau_0 = 10$ milliseconds, the most recent behavior of the process is given much higher weight than the past history associated with the process. Consequently, the scheduling algorithm is almost memory less, and simply predicts the length of the previous burst for the next quantum of CPU execution.

16. *A variation of the round-robin scheduler is the* ***regressive round-robin scheduler****. This scheduler assigns each process a time quantum and a priority. The initial value of a time quantum is 50 milliseconds. However, every time a process has been allocated the CPU and uses its entire time quantum (does not block for I/O), 10 milliseconds is added to its time quantum, and its priority level is boosted. (The time quantum for a process can be increased to a maximum of 100 milliseconds.) When a process blocks before using its entire time quantum, its time quantum is reduced by 5 milliseconds, but its priority remains the same. What type of process (CPU-bound or I/O-bound) does the regressive round-robin scheduler favor? Explain.*

17. Consider the following set of processes, with the length of the CPU Burst Time given in milliseconds

Process	*Burst Time*	*Priority*
P1	*2*	*2*
P2	*1*	*1*
P3	*8*	*4*
P4	*4*	*2*
P5	*5*	*3*

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5 all at time 0. Consider FCFS, SJF, nonpreemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1) scheduling algorithm

a. What is the turnaround time of each process for each of the scheduling algorithms ?
b. What is the waiting time of each process for each of these scheduling algorithms?
c. Which of the algorithms results in the minimum average waiting time (over all processes)?

Answer:

a. Turnaround time

	FCFS	**RR**	**SJF**	**Priority**
P1	**10**	**19**	**19**	**16**
P2	**11**	**2**	**1**	**1**
P3	**13**	**7**	**4**	**18**
P4	**14**	**4**	**2**	**19**
P5	**19**	**14**	**9**	**6**

b. Waiting time (turnaround time minus burst time)

	FCFS	**RR**	**SJF**	**Priority**
P1	**0**	**9**	**9**	**6**
P2	**10**	**1**	**0**	**0**
P3	**11**	**5**	**2**	**16**
P4	**13**	**3**	**1**	**18**
P5	**14**	**9**	**4**	**1**

c. Shortest Job First

18. The following processes are being scheduled using a preemptive, round-robin scheduling algorithm. Each process is assigned a numerical priority, with a higher number indicating a higher relative priority. In addition to the processes listed below, the system also has an idle task (which consumes no CPU resources and is identified as Pidle). This task has priority 0 and is scheduled whenever the system has no other available process to run. The length of a time quantum is 10 units. If a process is preempted by a higher-priority process, the preempted process is placed at the end of the queue

Thread	*Priority*	*Burst*	*Arrival*
P1	*40*	*20*	*0*
P2	*30*	*25*	*25*
P3	*30*	*25*	*30*
P4	*35*	*15*	*60*
P5	*5*	*10*	*100*
P6	*10*	*10*	*105*

a. What is the turnaround time for each process?

b. What is the waiting time for each process?

c. What is the CPU utilization rate?

<u>Answer:</u>

a. p1: 20-0 - 20, p2: 80-25 = 55, p3: 90 - 30 = 60, p4: 75-60 = 15, p5: 120-100 = 20, p6: 115-105 = 10

b. 1 p1: 0, p2: 40, p3: 35, p4: 0, p5: 10, p6: 0

c. 105/120 = 87.5 percent.

19. Which of the following scheduling algorithms could result in starvation?

a. First-come, first-served

b. Shortest job first

c. Round robin

d. Priority

<u>Answer:</u> Shortest job first and priority-based scheduling algorithms could result in starvation.

20. Consider a pre-emptive priority scheduling algorithm based on dynamically changing priorities. Larger priority numbers imply higher priority. When a process is waiting for the CPU (in the ready queue, but not running), its priority changes at a rate α; when it is running, its priority changes at a rate β. All processes are given a priority of 0 when they enter the ready queue. The parameters α and β can be set to give many different scheduling algorithms.

a. What is the algorithm that results from $\beta > \alpha > 0$?

b. What is the algorithm that results from $\alpha < \beta < 0$?

<u>Answer:</u>

a. FCFS

b. LIFO