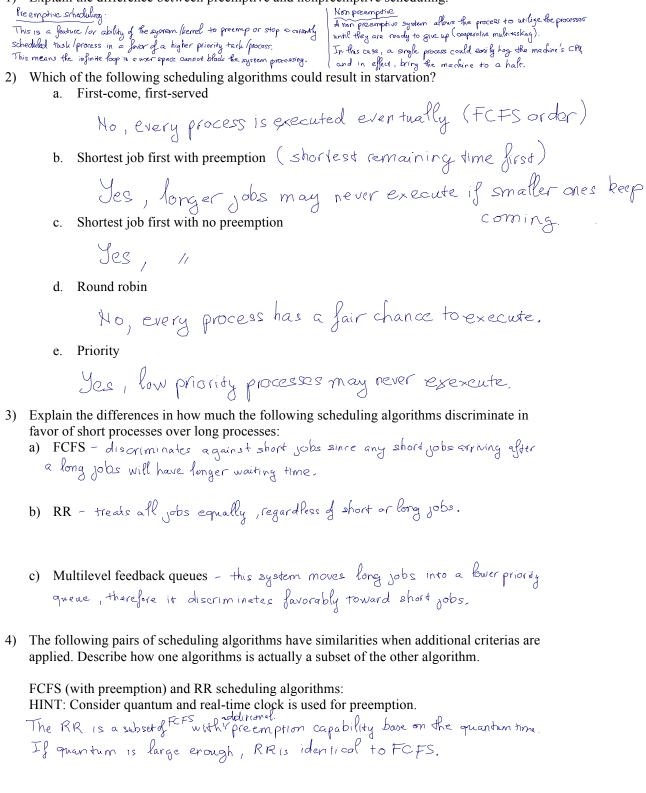
Operating System Design

Homework:

Operating System Review 2
 Chapter 5 to 7

NAME:		
STUDENT ID:		
DATE:		

Questions



1) Explain the difference between preemptive and nonpreemptive scheduling.

SJF is a priority scheduling where priority is the predicted. next CPU burst time.

Shortest Job First and Priority scheduling algorithms:

HINT: Consider the length of the CPU burst time as a priority and preemption.

5) Describe the critical section problem that occurs on multi-processor or multi-core computer systems. What is a solution for the critical section problem? Problem: In multi-processor or multi-core system, there are processes or threats that share common variables, or table, or files. Each process has a code segment. call critical Section (CS), in which the shared date is accessed. The problem is to ensure that when one process is executing in 12 CS, no other process 12 allowed to ensure that when one process is executing in 12 CS, no other process 13 allowed to ensure that when one process is executing in 12 CS, no other process 14 allowed to ensure their Critical Section can enter the Oracide it disc. 6) Linux implement multiple locking mechanisms. Describe the circumstances under which they to enter thir CS. Semaphores are the tods for synchronization. Subttons: Solving the problem is the clustering in testinal section, no other process is executing in its critical section. - fragees: process wishing to enter the Critical Section can enter to ensure that when a few is a bounded Warring. There is a bounded Amin'd on the number fitnes after processed are allowed to enter their CS, fixer a procession can enter the Oracide it does not a function of the content of the co
(for a short period of time) rather than incurring the
overhead of being put in a sleep queue duration.
7) What is the meaning of the term <i>busy waiting</i> ? What other kinds of waiting are there in an operating system? Can <i>busy waiting</i> be avoided altogether? Explain your answer.
Busy waiting means that a process is waiting for a condition to be satisfied in a tight bop without relinquishing the processor. Afternatively, a process could wait by relinquishing the processor, and block on a conditions wait to be awakened at some appropriete time in the Juture. It can be avoided but incurs
The overhead assoc. 8) A possible method for preventing deadlocks is to have a single, higher order resource that must be requested before any other resource. For example, if multiple threads attempt to access the synchronization objects A,B,C,D, and E, deadlock is possible (such synchronization objects maybe mutexes, semaphores, or condition variables). We can prevent the deadlock by adding a sixth object F. Whenever a thread wants to acquire the synchronization lock for any object A, B, C, D, and E, it must first acquire the lock for object F. This solution is known as containment: the locks for objects A, B, C, D, and E are contained within the lock for object F. Compare this scheme with the circular-wait scheme. Both schemes will prevent the deadlock. But the Circular-Wait algo will be more efficient than the Constainment algo. Since Circular-Wait allows multiple processors and their resonance as large, is only one process and hence no condition about Mutual Exclusion. Nor There is only one process and hence no condition about Mutual Exclusion.
10) Describe Priority Inversion and describe a technique that can be used to resolve Priority Inversion? Priority Inversion happens when a higher priority task waiting on a semaphore held by a lower priority task. Before the lower priority task release the semaphore, it is interrupted by another task and the

y a d the higher priority task has to wait.

Technique to resolve priority inversion; wary Priority Inheritance Protocol return to normal priority after releasing the.

11) Describe the methods that an Operating System can use to handle Deadlock condition in a Crisical Section process. Which method is generally used to handle Deadlock?

7	hore are 3 ways in goneral to handle	2 dead locks:
_	Deadlock prevention & avoidance.	- Donot aflow the system to get into deadlock state,
_	Deadlock detection & recovering	- About a process of prepart some consumer deadless of total
_	If dead lock rarely happers (1,2 time	mes a year), just igrore the problem altogether & simply reboot as nessesary than to incur the constant overhead & system penalties a succ. W/
		than to incur the constant overhead bounter penalties a stor wil
	Summer 2015	Silicon Valley University alead locks prevention or detection. This
		is the approach that both Windows & Unix take.