CS365 Operating System Final Exam

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Question Number | Score |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| Total: |  |

*Question1:*

Circle true or false: (total 23 pts, 1pt each)

|  |  |
| --- | --- |
| T F | 1. Many operating system merge I/O devices and files into a combined file because of the similarity of system calls for each. |
| T F | 1. System calls can be run in either user mode or kernel mode. |
| T F | 1. The difference between a program and a process is that a program is an active entity while a process is a passive entity. |
| T F | 1. Java's RMI is a feature similar to RPCs. |
| T F | 1. Each thread has its own register set and stack. |
| T F | 1. The single benefit of a thread pool is to control the number of threads. |
| T F | 1. Virtually all contemporary operating systems support kernel threads. |
| T F | 1. Linux distinguishes between processes and threads. |
| T F | 1. Race conditions are prevented by requiring that critical regions be protected by locks. |
| T F | 1. The value of a counting semaphore can range only between 0 and 1. |
| T F | 1. A deadlock-free solution eliminates the possibility of starvation |
| T F | 1. The local variables of a monitor can be accessed by only the local procedures. |
| T F | 1. Monitors are a theoretical concept and are not practiced in modern programming languages. |
| T F | 1. Mutex locks and binary semaphores are essentially the same thing. |
| T F | 1. Mutex locks and counting semaphores are essentially the same thing. |
| T F | 1. The circular-wait condition for a deadlock implies the hold-and-wait condition. |
| T F | 1. If a resource-allocation graph has a cycle, the system must be in a deadlocked state. |
| T F | 1. Protocols to prevent hold-and-wait conditions typically also prevent starvation. |
| T F | 1. The wait-for graph scheme is not applicable to a resource allocation system with multiple instances of each resource type. |
| T F | 1. Ordering resources and requiring the resources to be acquired in order prevents the circular wait from occurring and therefore prevents deadlock from occurring. |
| T F | 1. The banker's algorithm is useful in a system with multiple instances of each resource type. |
| T F | 1. A system in an unsafe state will ultimately deadlock. |
| T F | 1. Deadlock prevention and deadlock avoidance are essentially the same approaches for handling deadlock. |

*Question2:*

Explain the following terminology: (total 8pts, 1pt each)

1. **Preemptive Scheduling and Non-preemptive Scheduling**
2. **Dispatch latency**

1. **Turnaround time, waiting time, and response time**
2. **Spinlock**
3. **Mutex**
4. **Race Condition**
5. **Monitor**
6. **Deadlock**

*Question3:* (total 10 pts)

By “**Multilevel Queue Fixed priority**” scheduling algorithm, draw the CPU scheduling Gantt chart and complete the table for the give processes information.

Process Burst time Algorithm

* Foreground P1 53 (RR interval:20)

P2 17

P3 42

* Background P4 30 (FCFS)

P5 20

Gantt Chart:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | P1 | P2 | P3 | P4 | P5 |
| Waiting time |  |  |  |  |  |
| Turnaround time |  |  |  |  |  |
| Response time |  |  |  |  |  |

*Question4:* (total 20 pts)

1. A solution to the critical-section problem must satisfy which **three** requirements? (3pts)

2. We consider a system consisting of two processes, P0 and P1, each accessing two semaphores, S and Q, set to the value 1.

P0 P1

Wait(S) Wait(S)

Wait(Q) Wait(Q)

… …

… …

… …

Signal(S) Signal(Q)

Signal(S) Signal(S)

What kind of unwanted **situation(s)** will happen? Explain your answer. (3pts)

3. We consider a system consisting of two processes, P0 and P1, each accessing two semaphores, S and Q, set to the value 1.

P0 P1

Wait(S) Wait(S)

Wait(Q) Wait(Q)

… …

… …

… …

Signal(S) Signal(Q)

Signal(Q) Signal(S)

Does this have the chance to occur deadlock situation? Explain your answer? (3pts)

4. Allocation . Request . Available .

A B C D A B C D A B C D

P1 1 0 0 0 0 1 0 0 1 0 3 1

P2 1 1 0 0 0 0 1 0

P3 0 1 2 1 0 0 0 0

1) Draw the resource-allocation graph under the above situation. (3pts)

2) Is there a cycle? (1pt)

3) Is there a deadlock? (1pt)

5. Assume there are 5 processes *P*0 through *P*4;

3 resource types:

*A* (10 instances), *B* (5instances), and *C* (7 instances)

Snapshot at time *T*0:

*Allocation Max Available*

*A B C A B C A B C*

*P*0 0 1 0 7 5 3 3 3 2

*P*1 2 0 0 3 2 2

*P*2 3 0 2 9 0 2

*P*3 2 1 1 2 2 2

*P*4 0 0 2 4 3 3

Use banker’s algorithm to check (1) if the request (1,0,2) by ***P*1** can be granted? (3 pts) (2) Can request for (0,2,0) by ***P*0** be granted? (3 pts) Explain your answer.

*Question5:*

1. Give two examples of applications for which circuit switching and packet switching would be more suitable respectively. (4 points)

Circuit switching:

Packet switching:

1. Routers communicate using point-to-point instead of broadcast. That is, the broadcast is stopped at the gateway. Why would it be a bad idea for gateways to pass broadcast packets between networks? What would be the advantages of doing so? (2 points)

*Question 6:*

Circle the best answer. (20 pts, 1 pt for each)

1. A \_\_\_\_\_ is an example of a systems program.

A) command interpreter

B) Web browser

C) text formatter

D) database system

2. A message-passing model is \_\_\_\_.

A) easier to implement than a shared memory model for intercomputer communication

B) faster than the shared memory model

C) a network protocol, and does not apply to operating systems

D) only useful for small simple operating systems

3. The \_\_\_\_ of a process contains temporary data such as function parameters, return addresses, and local variables.

A) text section

B) data section

C) program counter

D) stack

4. The list of processes waiting for a particular I/O device is called a(n) \_\_\_\_.

A) standby queue

B) device queue

C) ready queue

D) interrupt queue

5. A process may transition to the Ready state by which of the following actions?

A) Completion of an I/O event

B) Awaiting its turn on the CPU

C) Newly-admitted process

D) All of the above

6. Which of the following is true of cooperative scheduling?

A) It requires a timer.

B) A process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state.

C) It incurs a cost associated with access to shared data.

D) A process switches from the running state to the ready state when an interrupt occurs.

7. \_\_\_\_ is the number of processes that are completed per time unit.

A) CPU utilization

B) Response time

C) Turnaround time

D) Throughput

8. \_\_\_\_ scheduling is approximated by predicting the next CPU burst with an exponential average of the measured lengths of previous CPU bursts.

A) Multilevel queue

B) RR

C) FCFS

D) SJF

9. The \_\_\_\_ scheduling algorithm is designed especially for time-sharing systems.

A) SJF

B) FCFS

C) RR

D) Multilevel queue

10. Which of the following scheduling algorithms must be nonpreemptive?

A) SJF

B) RR

C) FCFS

D) priority algorithms

11. Which of the following is true of multilevel queue scheduling?

A) Processes can move between queues.

B) Each queue has its own scheduling algorithm.

C) A queue cannot have absolute priority over lower-priority queues.

D) It is the most general CPU-scheduling algorithm.

12. \_\_\_\_\_\_ allows a thread to run on only one processor.

A) Processor affinity

B) Processor set

C) NUMA

D) Load balancing

13. \_\_\_\_\_ is a technique for handling critical sections in operating systems.

A) Nonpreemptive kernels

B) Preemptive kernels

C) Spinlocks

D) Peterson's solution

14. A semaphore \_\_\_\_.

A) is essentially an integer variable

B) is accessed through only one standard operation

C) can be modified simultaneously by multiple threads

D) cannot be used to control access to a thread's critical sections

15. A(n) \_\_\_\_\_\_\_ refers to where a process is accessing/updating shared data.

A) critical section

B) entry section

C) mutex

D) test-and-set

16. \_\_\_\_\_ can be used to prevent busy waiting when implementing a semaphore.

A) Spinlocks

B) Waiting queues

C) Mutex lock

D) Allowing the wait() operation to succeed

17. A deadlocked state occurs whenever \_\_\_\_.

A) a process is waiting for I/O to a device that does not exist

B) the system has no available free resources

C) every process in a set is waiting for an event that can only be caused by another process in the set

D) a process is unable to release its request for a resource after use

18. One necessary condition for deadlock is \_\_\_\_, which states that at least one resource must be held in a nonsharable mode.

A) hold and wait

B) mutual exclusion

C) circular wait

D) no preemption

19. In a system resource-allocation graph, \_\_\_\_.

A) a directed edge from a process to a resource is called an assignment edge

B) a directed edge from a resource to a process is called a request edge

C) a directed edge from a process to a resource is called a request edge

D) None of the above

20. A cycle in a resource-allocation graph is \_\_\_\_.

A) a necessary and sufficient condition for deadlock in the case that each resource has more than one instance

B) a necessary and sufficient condition for a deadlock in the case that each resource has exactly one instance

C) a sufficient condition for a deadlock in the case that each resource has more than once instance

D) is neither necessary nor sufficient for indicating deadlock in the case that each resource has exactly one instance