IAE 2 portions

CPU Scheduling: Basic concepts, Scheduling Criteria, Scheduling Algorithms. Thread Scheduling, Multiple-Processor Scheduling, Real-Time CPU Scheduling.

Synchronization: Background, Critical Section Problem, Mutex locks, Semaphores, Classic Problems of Synchronization.

Deadlocks: System Model, Deadlock characterization, Methods for handling deadlocks, Deadlock prevention, Deadlock avoidance, Deadlock Detection and Recovery from deadlock.

PART – A

Q.No	Questions	BT Level	Competence
1.	List the CPU scheduling algorithms.	BTL-1	Remembering
2.	Differentiate short term and long term scheduler.	BTL-4	Analyzing
3.	Analyse the critical section problem.	BTL-3	Applying
4.	Show the use of monitors in process synchronization.	BTL-4	Analyzing
5.	Mention the use of resource-allocation graph.	BTL-2	Understanding
6.	“Priority inversion is a condition that occurs in real time systems where a low priority process is starved because higher priority processes have gained hold of the CPU” – Comment on this statement.	BTL-5	Evaluating
7.	What is meant by 'starvation' in operating system?	BTL-2	Understanding
8.	Illustrate operation of semaphore with example procedure.	BTL-3	Applying
9.	What is the meaning of the term busy waiting?	BTL-1	Remembering
10.	Define deadlock.	BTL-1	Remembering
11.	Show what are the schemes used to handle deadlock.	BTL-3	Applying
12.	Give the four necessary conditions for deadlock to occur.	BTL-5	Evaluating
13.	“If there is a cycle in the resource allocation graph, it may or may not be in deadlock state“. Comment on this statement.	BTL-6	Creating
14.	List out the methods used to recover from the deadlock.	BTL-1	Remembering
15.	Show what are the various scheduling criteria .	BTL-3	Applying
16.	Point out the functions of Dispatcher Module.	BTL-4	Analyzing
17.	Is it possible to have deadlock with one process?justify.	BTL-5	Evaluating

PART – B

<table>
<tr><td>1.</td><td>(i) Define scheduling .Explain SJF scheduling algorithms . (8)
(ii) Compute the average waiting time for the processes using non-preemptive SJF scheduling algorithm.(5)
<table>
<tr><td>Process</td><td>Arrival time</td><td>Burst time</td></tr>
<tr><td>P1</td><td>0</td><td>7</td></tr>
<tr><td>P2</td><td>2</td><td>4</td></tr>
<tr><td>P3</td><td>4</td><td>1</td></tr>
<tr><td>P4</td><td>5</td><td>4</td></tr>
<tr><td>P5</td><td>3</td><td>4</td></tr>
</table>
</td><td>BTL-4</td><td>Analyzing</td></tr>
<tr><td>2.</td><td>Describe the differences among short- term, medium-term and long- term scheduling with suitable example. (13)</td><td>BTL-1</td><td>Remembering</td></tr>
</table>

<table>
<tr><td>3.</td><td>Discuss how the following pairs of scheduling criteria conflict in certain settings.
i. CPU utilization and response time. (4)
ii. Average turnaround time and maximum waiting time. (5)
iii. I/O device utilization and CPU utilization. (4)</td><td>BTL-1</td><td>Remembering</td></tr>
<tr><td>4.</td><td>Consider the following set of processes with the length of the CPU-burst time in given ms:
<table>
<tr><th>Process</th><th>Burst Time</th><th>Arrival time</th></tr>
<tr><td>P1</td><td>8</td><td>0</td></tr>
<tr><td>P2</td><td>4</td><td>1</td></tr>
<tr><td>P3</td><td>9</td><td>2</td></tr>
<tr><td>P4</td><td>5</td><td>3</td></tr>
<tr><td>P5</td><td>3</td><td>4</td></tr>
</table>
Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, priority and RR(quantum=2)scheduling. Also calculate waiting time and turnaround time for each scheduling algorithms.(13)</td><td>BTL-3</td><td>Applying</td></tr>
<tr><td>5.</td><td>Explain the differences in the degree to which the following scheduling algorithms discriminate in favor of short processes:
(i) RR (7)
(ii) Multilevel feedback queues. (6)</td><td>BTL-4</td><td>Analyzing</td></tr>
<tr><td>6.</td><td>Outline a solution to solve Bounded buffer, Readers and Writers problem and Dining philosopher problem. (13)</td><td>BTL-5</td><td>Evaluating</td></tr>
<tr><td>7.</td><td>Design how to implement wait() and signal() semaphore operations .(13)</td><td>BTL-6</td><td>Creating</td></tr>
<tr><td>8.</td><td>Explain Deadlock detection with suitable example. (13)</td><td>BTL-4</td><td>Analyzing</td></tr>
<tr><td>9.</td><td>Consider the snapshot of a system(13)
<table>
<tr><th></th><th>Max</th><th>Allocation</th><th>Available</th></tr>
<tr><td></td><td>A B C D</td><td>A B C D</td><td>A B C D</td></tr>
<tr><td>P0</td><td>2 0 0 1</td><td>4 2 1 2</td><td>3 3 2 1</td></tr>
<tr><td>P1</td><td>3 1 2 1</td><td>5 2 5 2</td><td></td></tr>
<tr><td>P2</td><td>2 1 0 3</td><td>2 3 1 6</td><td></td></tr>
<tr><td>P3</td><td>1 3 1 2</td><td>1 4 2 4</td><td></td></tr>
<tr><td>P4</td><td>1 4 3 2</td><td>3 6 6 5</td><td></td></tr>
</table>
Answer the following Using Banker’s algorithm,
(i) Illustrate that the system is in safe state by demonstrating an order in which the processes may complete?
(ii)If a request from process P1 arrives for(1,1,0,0) can the request be granted immediately?
(iii)if the request from p4 arrives for(0,0,2,0) can the request be granted immediately?</td><td>BTL-2</td><td>Understanding</td></tr>
<tr><td>10.</td><td>(i) Illustrate deadlock with neat example.(7)
(ii) The operating system contains 3 resources, the number of instance of each resource type are 7,7,10. The current resource allocation state is as shown below.
<table>
<tr><th></th><th colspan="3">Current Allocation</th><th colspan="3">Max Need</th></tr>
<tr><td></td><td>R1</td><td>R2</td><td>R3</td><td>R1</td><td>R2</td><td>R3</td></tr>
<tr><td>P1</td><td>2</td><td>2</td><td>3</td><td>3</td><td>6</td><td>8</td></tr>
<tr><td>P2</td><td>2</td><td>0</td><td>3</td><td>4</td><td>3</td><td>3</td></tr>
<tr><td>P3</td><td>1</td><td>2</td><td>4</td><td>3</td><td>4</td><td>4</td></tr>
</table></td><td>BTL-1</td><td>Remembering</td></tr>
</table>

	Is the current allocation in a safe state? (6)		
11.	Discuss in detail the critical section problem and write the algorithm for producer consumer problem.(13)	BTL-2	Understanding
12.	(i) Is it possible to have concurrency but not parallelism? Explain.(6) (ii) Consider a system consisting of four resources of the same type that are shared by three processes, each of which needs at most two resources. Show that the system is deadlock free. (7)	BTL-3	Applying
13	Describe what is deadlock. Write about deadlock condition and banker's algorithm in detail (13)	BTL-2	Understanding
14	For below Processes table, calculate the average waiting time for the algorithms: • First Come First Serve (FCFS) (4) • Shortest Job First (SJF) and (4) • Priority Scheduling (5) Process Burst Time Priority P1 10 3 P2 1 1 P3 2 4 P4 1 5 P5 5 2	BTL-3	Applying
15	Evaluate and explain the conditions for deadlock prevention. (13)	BTL-5	Evaluating
PART - C			
1.	Which of the following scheduling algorithms could result in starvation? (i)First-come, first-served (5) (ii) Shortest job first (5) (iii) Round robin (5) Detail with Justification.	BTL-6	Creating
2.	(i).Consider the following set of processes with the length of CPU burst time given in milliseconds. Process Burst Time priority Arrival Time P1 10 3 0 P2 1 1 1 P3 2 3 2 P4 1 4 1 P5 5 2 2 Draw the Gantt chart for the execution of these processes using FCFS, SJF, SRTS, pre-emptive and non pre-emptive priority and Round robin with the time slice of 2ms, Find average waiting time and turnaround time using each of the methods. (10). (ii).Explain –multi level queue and multi level feedback queue scheduling with suitable examples. (5)	BTL-5	Evaluating
3.	Consider a system consisting of 'm' resources of the same type, being shared by 'n' processes. Resources can be requested and released by processes only one at a time. Show that the system is deadlock free if the following two conditions hold: (15) a) The maximum need of each process is between 1 and m resources b) The sum of all maximum needs is less than m+n.	BTL-5	Evaluating

4.	Consider the following system snapshot using data structures in the Banker's algorithm with resources A,B,C and D and process P0 to P4: Max Allocation Available Need A B C D A B C D A B C D A B C D P0 6 0 1 2 4 0 0 1 3 2 1 1 P1 1 7 5 0 1 1 0 0 P2 2 3 5 6 1 2 5 4 P3 1 6 5 3 0 6 3 3 P4 1 6 5 6 0 2 1 2 Using Banker's algorithm, answer the following questions: (i)How many resources of type A,B,C and D are there? (3) (ii)What are the contents of the need matrix? (3) (iii) Is the system in a safe state? Why? (3) (iv) If a request from process P4 arrives for additional resources of (1,2,0,0) can the banker's algorithm grant the request immediately? Show the new system state and other criteria. (6)	**BTL-5**	Evaluating
5.	Consider the following set of processes with the length of the CPU-burst time in given ms: all 5 processess arrive at time 0 in the order given. Process Burst Time P1 10 P2 29 P3 03 P4 07 P5 12 Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, priority and RR(quantum=10)scheduling. Also calculate average waiting time and turnaround time for each scheduling algorithms. (15)	**BTL-6**	Creating

1.Define CPU scheduling.

CPU scheduling is the process of switching the CPU among various processes. CPU scheduling is the basis of multiprogrammed operating systems. By switching the CPU among processes, the operating system can make the computer more productive.

2.What is preemptive and nonpreemptive scheduling?

Under nonpreemptive scheduling once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or switching to the waiting state. Preemptive scheduling can preempt a process which is utilizing the CPU in between its execution and give the CPU to another process.

3.What is a Dispatcher?

The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler. This function involves:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program to restart that program.

4.What is dispatch latency?

The time taken by the dispatcher to stop one process and start another running is known as ispatch latency.

5.What are the various scheduling criteria for CPU scheduling?

The various scheduling criteria are

- CPU utilization

- Throughput
- Turnaround time
- Waiting time
- Response time

6.Define throughput?

Throughput in CPU scheduling is the number of processes that are completed per unit time. For long processes, this rate may be one process per hour; for short transactions, throughput might be 10 processes per second.

7.What is turnaround time?

Turnaround time is the interval from the time of submission to the time of completion of a process. It is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O.

8.Define race condition.

When several process access and manipulate same data concurrently, then the outcome of the execution depends on particular order in which the access takes place is called race condition. To avoid race condition, only one process at a time can manipulate the shared variable.

9.What is critical section problem?

Consider a system consists of 'n' processes. Each process has segment of code called a critical section, in which the process may be changing common variables, updating a table, writing a file. When one process is executing in its critical section, no other process can allowed to execute in its critical section.

10.What are the requirements that a solution to the critical section problem must satisfy?

The three requirements are

- Mutual exclusion
- Progress
- Bounded waiting

11.Define entry section and exit section.

The critical section problem is to design a protocol that the processes can use to cooperate. Each process must request permission to enter its critical section. The section of the code implementing this request is the entry section. The critical section is followed by an exit section. The remaining code is the remainder section.

12.Give two hardware instructions and their definitions which can be used for implementing mutual exclusion.

- TestAndSet

```
boolean TestAndSet (boolean &target)
{
boolean rv = target;
target = true;
return rv;
}
```

- Swap

```
void Swap (boolean &a, boolean &b)
{
boolean temp = a;
a = b;
b = temp;
}
```

13. What is semaphores?

A semaphore 'S' is a synchronization tool which is an integer value that, apart from initialization, is accessed only through two standard atomic operations; wait and signal. Semaphores can be used to deal with the n-process critical section problem. It can be also used to solve various Synchronization problems.

14. Define busy waiting and spinlock.

When a process is in its critical section, any other process that tries to enter its critical section must loop continuously in the entry code. This is called as busy waiting and this type of semaphore is also called a spinlock, because the process while waiting for the lock.

15. What resources are used when a thread is created? How do they differ from those used when a process is created?

Because a thread is smaller than a process, thread creation typically uses fewer resources than process creation. Creating a process requires allocating a process control block (PCB), a rather large data structure. The PCB includes a memory map, list of open files, and environment variables. Allocating and managing the memory map is typically the most time-consuming activity. Creating either a user or kernel thread involves allocating a small data structure to hold a register set, stack, and priority.

16. What advantage is there in having different time-quantum sizes on different levels of a multilevel queueing system?

Processes that need more frequent servicing, for instance, interactive processes such as editors, can be in a queue with a small time quantum. Processes with no need for frequent servicing can be in a queue with a larger quantum, requiring fewer context switches to complete the processing, making more efficient use of the computer

17. Define deadlock.

A process requests resources; if the resources are not available at that time, the process enters a wait state. Waiting processes may never again change state, because the resources they have requested are held by other waiting processes. This situation is called a deadlock.

18. What is the sequence in which resources may be utilized?

Under normal mode of operation, a process may utilize a resource in the following sequence:

- Request: If the request cannot be granted immediately, then the requesting process must wait until it can acquire the resource.
- Use: The process can operate on the resource.
- Release: The process releases the resource.

19. What are conditions under which a deadlock situation may arise?

A deadlock situation can arise if the following four conditions hold simultaneously in a system:

a. Mutual exclusion
b. Hold and wait
c. No pre-emption

20. What is a resource-allocation graph?

Deadlocks can be described more precisely in terms of a directed graph called a system resource allocation graph. This graph consists of a set of vertices V and a set of edges E. The set of vertices V is partitioned into two different types of nodes; P the set consisting of all active processes in the system and R the set consisting of all resource types in the system.

21. Define request edge and assignment edge.

A directed edge from process Pi to resource type Rj is denoted by Pi→Rj; it signifies that process Pi requested an instance of resource type Rj and is currently waiting for that resource. A directed edge from resource type Rj to process Pi is denoted by Rj→Pi, it signifies that an instance of resource type has been allocated to a process Pi. A directed edge Pi→Rj is called a request edge. A directed edge Rj→Pi is called an assignment edge.

22.What are the methods for handling deadlocks?

The deadlock problem can be dealt with in one of the three ways:

a. Use a protocol to prevent or avoid deadlocks, ensuring that the system will never enter a deadlock state.
b. Allow the system to enter the deadlock state, detect it and then recover.
c. Ignore the problem all together, and pretend that deadlocks never occur in the system.

23. Define deadlock prevention.

Deadlock prevention is a set of methods for ensuring that at least one of the four necessary conditions like mutual exclusion, hold and wait, no preemption and circular wait cannot hold. By ensuring that that at least one of these conditions cannot hold, the occurrence of a deadlock can be prevented.

24. Define deadlock avoidance.

An alternative method for avoiding deadlocks is to require additional information about how resources are to be requested. Each request requires the system consider the resources currently available, the resources currently allocated to each process, and the future requests and releases of each process, to decide whether the could be satisfied or must wait to avoid a possible future deadlock.

25. What are a safe state and an unsafe state?

A state is safe if the system can allocate resources to each process in some order and still avoid a deadlock. A system is in safe state only if there exists a safe sequence. A sequence of processes <P1,P2,....Pn> is a safe sequence for the current allocation state if, for each Pi, the resource that Pi can still request can be satisfied by the current available resource plus the resource held by all the Pj, with j<i. if no such sequence exists, then the system state is said to be unsafe.

26. What is banker’s algorithm?

Banker’s algorithm is a deadlock avoidance algorithm that is applicable to a resource-allocation system with multiple instances of each resource type. The two algorithms used for its implementation are:

a. Safety algorithm: The algorithm for finding out whether or not a system is in a safe state.

b. Resource-request algorithm: if the resulting resource allocation is safe, the transaction is completed and process Pi is allocated its resources. If the new state is unsafe Pi must wait and the old resource-allocation state is restored.