20CS302 – OPERATING SYSTEMS

MODULE II - QUESTION BANK

PART – A

1. Enlist some of the queues on the typical system. Analyze in which state, each queue play its role?

- All processes, upon entering into the system, are stored in the **Job Queue**.
- Processes in the Ready state are placed in the **Ready Queue**.
- Processes waiting for a device to become available are placed in **Device Queues**. There are unique device queues available for each I/O device.

2. Every scheduling algorithm has a type of a situation where it is the best choice. Find out the suitable scheduling algorithm for the given situations.

a) The incoming processes are short and there is no need for the processes to execute in a specific order.

b) The processes are a mix of long and short processes and the task will only be completed if all the processes are executed successfully in a given time.

a) In this case, **FCFS** works best when compared to SJF and RR because the processes are short which means that no process will wait for a longer time. When each process is executed one by one, every process will be executed eventually.

b) **Round Robin scheduling** works efficiently here because it does not cause starvation and also gives equal time quantum for each process.

3. Give the difference between preemptive and non preemptive scheduling.

Preemptive Scheduling	**Non-Preemptive Scheduling**
It allows a process to be interrupted in the midst of its execution, taking the CPU away and allocating it to another process.	It ensures that a process relinquishes control of the CPU only when it finishes with its current CPU burst.
It incurs a cost associated with access shared data.	It does not increase the cost.
It also affect the design of the operating system kernel	It does not affects the design of operating system kernel
It is more complex	Simple, but very inefficient
Example: Round robin	Example: FCFS

4. **What are the requirements that a solution to the critical section problem must satisfy?**
    - Mutual Exclusion
    - Progress
    - Bounded Waiting

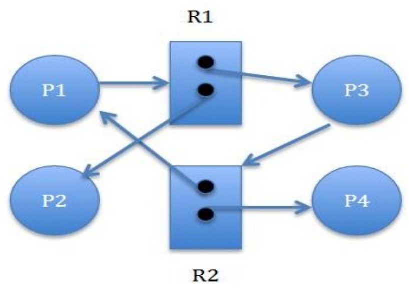
5. **When does a race condition arise and how it is resolved?**

    A situation where several processes access and manipulate the same data concurrently and outcome of the execution depends on the particular order in which the access takes place, is called a race condition.
    Resolved by using critical section problem.
    A Critical Section is a code segment that accesses shared variables and has to be executed as an atomic action.
    It means that in a group of cooperating processes, at a given point of time, only one process must be executing its critical section.
    If any other process also wants to execute its critical section, it must wait until the first one finishes.

6. **Differentiate short-term, medium-term and long-term scheduling.**

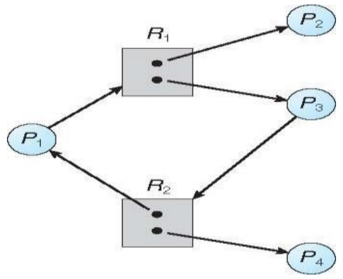
Long-Term Scheduler	**Short-Term Scheduler**	**Medium-Term Scheduler**
It is a job scheduler	It is a CPU scheduler It	is a process swapping scheduler.
Speed is lesser than short term scheduler	Speed is fastest among other two	Speed is in between both short and long term scheduler.
It controls the degree of multiprogramming	It provides lesser control over degree of multiprogramming	It reduces the degree of multiprogramming.
It is almost absent or minimal in time sharing system	It is also minimal in time sharing system	It is a part of Time sharing systems.
It selects processes from pool and loads them into memory for execution	It selects those processes which are ready to execute	It can re-introduce the process into memory and execution can be continued.

7. **Consider a system with four processes P1, P2, P3 and P4 and two resources R1 and R2 respectively. Each resource has two instances. Furthermore:**
    P1 allocates an instance of R2 and requests an instance of R1
    P2 allocates an instance of R1 and doesn't need any other resource.
    P3 allocates an instance of R1 and requests an instance of R2
    P4 allocates an instance of R2 and doesn't need any other resource.
    Draw the resource allocation graph. Is there any deadlock in this situation? Justify your answer.



There is a cycle P1 -> R1 -> P3 -> R2 -> P1, but no deadlock. P4 and P2 may release the resources once it is done. Those resources can be allocated to other requesting processes.

8. **"If there is a cycle in the resource allocation graph, it may or may not be in deadlock state". Justify the statement with an example**



Here cycle has occurred but since P4 and P2 may finish and release the resources the cycle may be broken. Hence the deadlock may not occur.

9. **The main memory has the following non contiguous free spaces: 50K, 100K, 150K, 200K and 250K. How will the OS allocate memory to a 260K process using best fit method?**
The main memory has non contiguous free spaces: 50K, 100K, 150K, 200K and 250K. In order to allocate the memory to a 260K process using best fit method, the process have to wait until there is an enough memory to satisfy the request.

10. **Compare Logical address and Physical address.**

Logical Address	Physical Address
It is the virtual address generated by CPU	The physical address is a location in a memory unit.
Set of all logical addresses generated by CPU in reference to a program is referred as Logical Address Space.	Set of all physical addresses mapped to the corresponding logical addresses is referred as Physical Address.
The user can view the logical address of a program.	The user can never view physical address of program
The user uses the logical address to access the physical address.	The user can not directly access Physical address.

11. On a system using simple segmentation, compute the physical address for each of the logical address is given in the following segment table. If the address generates a segment fault, indicate so.

Segment number	Base address	Length of the segment
0	330	124
1	876	211
2	111	99
3	498	302

a) 0,99

Offset = 99
Segment length = 124
Segment = 0
Offset 99 is less than segment length 124
Starting position of segment 0 is 330
Physical address = offset + segment base
= 99 + 330 = 429

b) 2,78

Offset = 78
Segment length = 99
Segment = 2
Offset 78 is less than segment length 99
Starting position of segment 2 is 111
Physical address = offset + segment base
= 111 + 78 = 189

c) 1,265

Offset = 265
Segment length= 211
Segment = 1
Offset 265 is greater than segment length 211
This address results in a segment fault.

d) 3,222

Offset = 222
Segment length = 302
Segment = 3
Offset 222 is less than segment length 302
Starting position of segment 3 is 498
Physical address = offset + segment base

= 498 + 222 = 720

e) 0,111

Offset = 111
Segment length = 124
Segment = 0
Offset 111 is less than segment length 124
Starting position of segment 0 is 330
Physical address = offset + segment base
= 330 + 111 = 441

12. **Interpret the situation when compaction cannot be applied for fragmentation.** Compaction is a process in which the free space is collected in a large *memory* chunk to make some space available for processes. If relocation is static and is done at load time, compaction cannot be done.

13. **Differentiate FIFO and Second chance algorithm in page replacement**
    FIFO replaces in the order of reference of pages even if the page is referred after their entry, but second chance algorithm gives a second chance to the oldest pages if it is referred after its entry in the page table.

14. **State the cause of thrashing? How does the system detect thrashing? Once it detects thrashing, what can the system do to eliminate this problem?**
    Thrashing is caused by under allocation of the minimum number of pages required by a process, forcing it to continuously page fault. The system can detect thrashing by evaluating the level of CPU utilization as compared to the level of multiprogramming. It can be eliminated by reducing the level of multiprogramming.

15. **Why are page sizes always a power of 2?**
    - Paging is implemented by breaking up an address into a page and offset number.
    - It is most efficient to break the address into X page bits and Y offset bits, rather than perform arithmetic on the address to calculate the page number and offset.
    - Because each bit position represents a power of 2, splitting an address between bits results in a page size that is a power of 2.

16. **To reduce the page fault rate, some systems employ proportional allocation. If such a system has a total of 64 frames and if two processes P1 and P2 of size 10 and 127 are ready for execution, how many frames would be allocated to each process?**
    Formula: ai = si / S * m
    m -> total number of available frames
    a1 = 10 / (127 + 10) * 64 ≈ 5 frames
    a2 = 127 / (127 +10) * 64 ≈ 59 frames

17. **Let total available frames be 20 and the requirement of each process is given as follows: P1 – 10; P2 – 5; P3 – 15; P4 – 20; Find the number of frames allocated to each process using equal allocation and proportional allocation.**

Frame Allocation = no of available frame / no of process
= 20 / 4
= 5 frames per process
Proportional allocation:
S = 10 + 5 + 15 + 20 = 50
P1 ➔10/50 * 20 = 4
P2 ➔5/50 * 20 = 2
P3 ➔15/50 * 20 = 6
P4 ➔20/50 * 20 = 8

18. **Consider a logical address space of 64 pages of 1,024 words each, mapped onto a physical memory of 32 frames.**
    a. How many bits are there in the logical address?
    b. How many bits are there in the physical address?

    a. There are 64 pages in logical address space
    so 2^6 = 64 Therefore page size = 2^n = 2^6
    Page size = 6 bits
    We have 1024 words 2^{10} = 10 bits
    So the total bits in logical address is 10 + 6= 16 bits

    b. There are 32 frames in physical memory
    so 2^5 = 32 Page size = 5 bits
    We have 1024 words 2^{10} = 10 bits
    So the total bits in physical address is 10 + 5= 15 bits

19. **Consider a paging hardware with a TLB. Assume that the entire page table and all the pages are in the physical memory. It takes 10 milliseconds to search the TLB and 80 milliseconds to access the physical memory. If the TLB hit ratio is 0.6, calculate the effective memory access time (in milliseconds)**
    T(eff) = hit ratio * (TLB access time + Main memory access time) + (1 – hit ratio) * (TLB access time + 2 * main memory time)
    = 0.6*(10+80) + (1-0.6)*(10+2*80)
    = 0.6 * (90) + 0.4 * (170)
    = 122
20. **Consider a paging system with the page table stored in memory. If a memory reference takes 200 nanoseconds, how long does a paged memory reference take? If we add associative registers and 75 percent of all page table references are found in the associative registers, what is the effective memory reference time? (Assume that finding a page table entry in the associative register takes zero time, if the entry is there.)**
    A paged memory reference would take 400 nanoseconds; 200 nanoseconds to access the page table and 200 nanoseconds to access the word in memory.
    Effective memory reference time = 75% × TLB hit time + 25% × TLB miss time
    = 75% × 200 ns + 25% × 400 ns
    = 250 ns
    TLB access time is 0 ns. So, when there is a TLB hit, we need only 200 ns to access memory. When there is a TLB miss, we need to look up page table, stored in

memory, requiring 200 ns time. After that, we need another 200 ns time for actual access.

21. **Compare binary semaphore with general semaphore.**
    A binary semaphore may only take on the values 0 and 1. A general semaphore may take on any integer value.

22. **A shared variable x, initialized to zero, is operated on by four concurrent processes W, X, Y, Z as follows. Each of the processes W and X reads x from memory, increments by one, stores it to memory, and then terminates. Each of the processes Y and Z reads x from memory, decrements by two, stores it to memory, and then terminates. Each process before reading x invokes the P operation (i.e., wait) on a counting semaphore S and invokes the V operation (i.e., signal) on the semaphore S after storing x to memory. Semaphore S is initialized to two. What is the maximum possible value of x after all processes complete execution?**
    Ans: 2

23. **A process in operating systems needs a resource. Elucidate the protocol for resource usage**
    Ans .
    1) Requests a resource
    2) Use the resource
    3) Releases the resource

24. **Person A and B have joint account with a balance amount of Rs.10,000. Person 'A' deposits Rs.3000 and person B withdraws Rs.2000. Demonstrate race condition for the given situation**

Sequence of operations of Person A	**Sequence of operations of Person B**
Read account as acc1	Read account as acc2
acc1 = acc1 + 3000	acc2 = acc2 - 2000
account = acc1	account= acc2

If the order of execution is
T0: Person A: Read account as acc1 [acc1=10000]
T1: Person A: acc1 = acc1 + 3000 [acc1=13000]
T2: Person B: Read account as acc2 [acc2=10000]
T3: Person B: acc2 = acc2 – 2000 [acc2=8000]
T4: Person A: account = acc1 [account=13000]
T5: Person B: account = acc2 [account=8000]
The correct value should be 11000. But the final value is 8000 because of the order of execution. Hence the race condition is demonstrated.

25. **Mutual exclusion property is preserved in Peterson's solution. Justify**
Each process Pi enters its critical section only if either flag[j] == false or turn == i. Even if both the processes have flag value of '1' the turn value can be set by any one process. Hence only one process will be inside the critical section thereby preserving mutual exclusion.

26. Compare test and set & compare and swap functions

Test-and-set modifies the contents of a memory location and returns its old value as a single atomic operation. Compare-and-swap atomically compares the contents of a memory location to a given value and, only if they are the same, modifies the contents of that memory location to a given new value.

27. Find the reason for spinlock in mutex locks

While a process is in its critical section, any other process that tries to enter its critical section must loop continuously in the call to acquire(). Hence spinlock (busy waiting) occurs.

28. Justify the need for priority inversion in processes.

"priority inversion" is a problematic scenario in scheduling in which a high priority task is indirectly preempted by a lower priority task effectively "inverting" the relative priorities of the two tasks. Under the policy of priority inheritance, whenever a high priority task has to wait for some resource shared with an executing low priority task, the low priority task is temporarily assigned the priority of the highest waiting priority task for the duration of its own use of the shared resource.

29. Relate the situation of dining philosopher problem to process synchronization

It is a simple representation of the need to allocate limited resources among several processes in a deadlock-free and starvation-free manner.

30. Compare monitor and semaphore

S.No	Semaphore	Monitor
1	Semaphores is an integer variable S.	Monitor is an abstract data type.
2	Has to explicitly acquire a lock before using a shared resource.	Automatically acquire the necessary locks.
3	Using semaphores incorrectly can lead to timing errors	The monitor construct ensures that only one process at a time is active within the monitor.
4	The x.signal() operation resumes exactly one suspended process(x is a condition variable. If no process is suspended, then the signal() operation has no effect;	signal() operation associated with semaphores, will always affects the state of the semaphore.

31. Deadlock cannot be prevented by denying mutual exclusion in certain situations. If the statement is true give reason.

The statement is true. We cannot prevent deadlocks by denying the mutual-exclusion condition, because some resources are intrinsically nonsharable.

32. Race conditions are possible in many computer systems. Consider a banking system that maintains an account balance with two functions: deposit (amount) and withdraw (amount). These two functions are passed the amount

that is to be deposited or withdrawn from the bank account balance. Assume that a husband and wife share a bank account. Concurrently, the husband calls withdraw () function and the wife calls deposit (). Describe how a race condition is possible and what might be done to prevent the race condition from occurring.
Answer:
Assume the balance in the account is 250.00 and the husband calls withdraw (50) and the wife calls deposit (100.)Obviously the correct value should be 300.00. Since these two transactions will be serialized, the local value of balance for the husband becomes 200.00, but before he can commit the transaction, the deposit (100) operation takes place and updates the shared value of balance to 300.00. We then switch back to the husband and the value of the shared balance is set to 200.00 – obviously an incorrect value.

33. **Elucidate the meaning of the term busy waiting. Can busy waiting be avoided altogether? Explain your answer.**
Answer:
While a process is in it critical section, any other process that tries to enter the critical section must loop continuously in the entry code. Busy waiting wasted CPU cycles that some other process might be able to use productively. This type of semaphore is also called a spinlock because the process “spins” while waiting for the lock. To overcome busy waiting, wait() and signal() semaphore operations are redefined . In wait() operation, if semaphore value is not positive, then instead of busy waiting, the process can block itself. block() operation places a process in waiting queue associated with semaphore. A process that is blocked, waiting on semaphore S, should be restarted when some other process executes a signal() operation. The process restarted using wakeup() operation.

34. **Is readers and writers problem a subset of producer’s consumer problem? Justify.**
Answer:
Readers and writers problem is not a subset of producer’s consumer problem but both problems are about resource sharing. Producer’s consumer problem share many resources between single producer and a single consumer which uses a queue. Producer fills the queue at the tail and consumer empties it from head.
In reader writer problem, there is only a resource shared by one or more readers. Two or more readers can access the resource concurrently between them but absolutely a reader couldn’t access to resource with writer. One at a time can access the resource and so there is total mutual exclusion when writer uses the resource.

35. **Demonstrate when a system is said to be in safe state.**
Answer:
A safe state is one in which the available resources can be allocated to each process in some order without causing deadlocks. A system is in a 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. That is, when Pi terminates Pi+1 can

obtain its needed resources, and so on. If no such sequence exists, then the system is in an unsafe state. An unsafe state may not always leads to deadlocks but has a higher possibility of causing deadlocks.

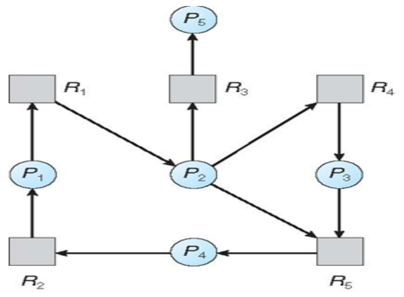
36. Compare and contrast between internal and external fragmentation.

	Internal Fragmentation	**External Fragmentation**
Basic	It occurs when fixed sized memory blocks are allocated to the processes.	It occurs when variable size memory spaces are allocated to the processes dynamically.
Occurrence	When the memory assigned to the process is slightly larger than the memory requested by the process this creates free space in the allocated block causing internal fragmentation.	When the process is removed from the memory, it creates the free space in the memory causing external fragmentation.
Solution	The memory must be partitioned into variable sized blocks and assign the best fit block to the process.	Compaction, paging and segmentation.

37. Interpret the situation when compaction cannot be applied for fragmentation
If relocation is static and is done at assembly or load time, compaction cannot be done.

38. Differentiate FIFO and Second chance algorithm in page replacement
FIFO replaces in the order of reference of pages even if the page is referred after their entry, but second chance algorithm gives a second chance to the oldest pages if it is referred after its entry in the page table.

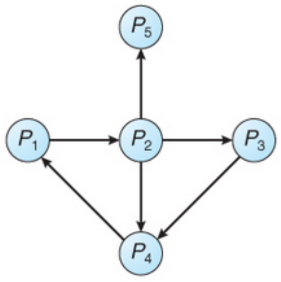
39. Depict the term wait_for_graph and construct the same for the following resource allocation graph.



Wait_For_Graph:

- If all resource types has only single instance, then we can use a graph called wait-for-graph, which is a variant of resource allocation graph.
- In wait-for graph
    - Nodes are processes
    - P_i P_j if P_i is waiting for P_j

- Periodically invoke an algorithm that searches for a cycle in the graph.
- If there is a cycle, there exists a deadlock



40. Under what circumstances user level threads are better than the kernel level threads?

User-Level threads are managed entirely by the run-time system (user-level library).The kernel knows nothing about user-level threads and manages them as if they were single-threaded processes. User-Level threads are small and fast, each thread is represented by a PC, register, stack, and small thread control block. Creating a new thread, switching between threads, and synchronizing threads are done via procedure call. i.e. no kernel involvement. User-Level threads are hundred times faster than Kernel-Level threads. User level threads are simple to represent, simple to manage and fast and efficient.

41. Distinguish between preemptive and non-preemptive Scheduling.

Under non preemptive 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.

42. Define: Belady's anomaly.

In computer storage, Belady's anomaly is the phenomenon in which increasing the number of page frames results in an increase in the number of page faults for certain memory access patterns. This phenomenon is commonly experienced when using the first-in first-out (FIFO) page replacement algorithm.

43. Define demand paging in memory management.

In virtual memory systems, demand paging is a type of swapping in which pages of data are not copied from disk to RAM until they are needed.

44. How the problem of external fragmentation can be solved.

Solution to external fragmentation:
1) Compaction: shuffling the fragmented memory into one contiguous location.
2) Virtual memory addressing by using paging and segmentation.

45. Outline about TLB.

A translation lookaside buffer (TLB) is a memory cache that is used to reduce the time taken to access a user memory location. It is a part of the chip's memory-management unit (MMU). The TLB stores the recent translations of virtual memory to physical memory and can be called an address-translation cache.

46. **What is Internal Fragmentation?**
When the allocated memory may be slightly larger than the requested memory, the difference between these two numbers is internal fragmentation.

47. **What is the use of Valid-Invalid Bits in Paging?**
When the bit is set to valid, this value indicates that the associated page is in the process's logical address space, and is thus a legal page. If the bit is said to invalid, this value indicates that the page is not in the process's logical address space. Using the valid-invalid bit traps illegal addresses.

PART – B

1. **Assume that the following processes arrive in the order with the length of the CPU-burst time given in milliseconds.**

Job	**Burst time (ms)**	**Priority**	**Arrival Time**
A	**5**	**2**	**0**
B	**3**	**2**	**2**
C	**25**	**1**	**4**
D	**7**	**4**	**6**
E	**15**	**3**	**7**

a) Give the Gantt chart illustrating the execution of processes using FCFS, Round Robin (quantum=2), preemptive priority and SRT scheduling.
b) Calculate the average waiting time and average turn-around time for each of the above algorithm.

2. **Consider the following snapshot of a system. Execute Bankers algorithm.**

	Allocation	**Max**	**Available**
P0	**1 2 0 1**	**2 3 4 3**	**2 3 3 2**
P1	**2 2 0 1**	**3 4 2 4**	
P2	**4 5 7 1**	**5 7 8 5**	
P3	**1 1 0 0**	**2 2 0 0**	
P4	**2 3 4 4**	**3 4 5 5**	

i) What is the content of need matrix
ii) Is the system in a safe state? If it is safe, write the safe sequence.
iii) If a request from p1 arrives for (1, 2, 1, 0) can the request be immediately granted? If granted, write the sequence of the process.

3. **Describe various techniques for structuring the page table in a page memory management scheme.**

4. **On a system using simple segmentation, compute the physical address for each of the logical address is given in the following segment table. If the address generates a segment fault, indicate so.**

Segment number	Base address	Length of the segment
0	330	124
1	876	211
2	111	99
3	498	302

i) 0, 99
ii) 2, 78
iii) 1, 265
iv) 3, 222
v) 0, 111

5. Explain the Banker's algorithm for deadlock avoidance.
6. When do page fault occurs? Describe the actions taken by the operating system when a page fault occurs.
7. Find the number of page faults for the following reference string 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6
   Let the number of frames be 3 & 4 and initially all are empty using following algorithms
   (a)First In First Out (FIFO)
   (b)Least Recently Used (LRU)
   (c)Optimal replacement
8. P1, P2, P3 given in the below table, arrives for execution in the given arrival time and Burst Time, determine the Average Waiting time and Average turnaround time using
   (a)First Come First Serve
   (b)Shortest Job First
   (c) Shortest Remaining Time
   (d)Round Robin Scheduling(Quantum = 3)

Process	Arrival Time	Priority	Burst time
P0	0	1	5
P1	1	2	3
P2	2	1	8
P3	3	3	6

9. Consider the following snapshot of a system. Execute Banker's algorithm and derive the following.

Process	Allocation				Maximum				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	0	0	1	2	0	0	1	2	1	5	2	0
P1	1	0	0	0	1	7	5	0				
P2	1	3	5	4	2	3	5	6				
P3	0	6	3	2	0	6	5	2				
P4	0	0	1	4	0	6	5	6				

   a) Find the Contents of Need matrix
   b) Is the system in safe state? Safe sequence if the system is safe.
   c) Can the request made by process P1 (0, 4, 2, 0) be granted immediately?

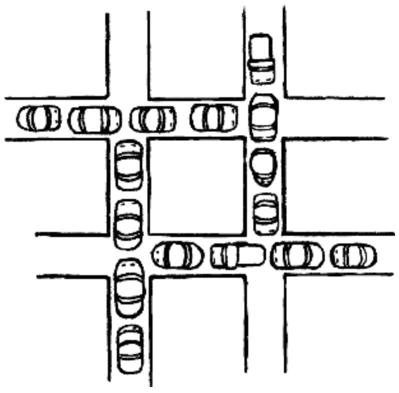
10. Describe the various classic problems of synchronization with examples.
11. Given six memory partitions of 300 KB, 600 KB, 350 KB, 200KB, 750KB,and 125KB(in order), how would the first-fit, best-fit, and worst-fit Main Memory algorithms place processes of size 115KB, 500KB, 358KB, 200KB,and 375KB(in order)? Rank the algorithms in terms of how efficiently they use memory.
12. Demonstrate the steps in handling page fault with a neat sketch.
13. Consider the following page reference string:
    A, B, C, D, A, B, E, A, B, C, D, E
    How many page faults would occur for the following replacement algorithms with 3 and 4 frames?
    i) FIFO
    ii) LRU
    iii) Optimal
    Whether this reference string suffers from Belady's anomaly?
14. Explain in detail the various memory allocation Techniques.
15. Design a solution for Dining Philosopher problem using Monitor.
16. Consider the following snapshot of a system. Execute Bankers algorithm.

	Allocation	Max	Available
P0	1 2 0 1	2 3 4 3	2 3 3 2
P1	2 2 0 1	3 4 2 4	
P2	4 5 7 1	5 7 8 5	
P3	1 1 0 0	2 2 0 0	
P4	2 3 4 4	3 4 5 5	

    i) What is the content of need matrix
    ii) Is the system in a safe state? If it is safe, write the safe sequence.
    iii) If a request from p1 arrives for (1, 2,1, 0) can the request be immediately granted? If granted, write the sequence of the process.
17. Write the functions for producer consumer in bounded buffer to implement critical section. Clearly justify why deadlocks cannot arise in a bounded buffer producers–consumers system.
18. Apply semaphore to solve Dining philosopher problem using semaphore deduce its disadvantages.
19. Given five memory partitions of 100Kb, 500Kb, 200Kb, 300Kb, 600Kb (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of 212 Kb, 417 Kb, 112 Kb, and 426 Kb (in order)? Which algorithm makes the most efficient use of memory?
20. Depict the term synchronization. Discuss the two process solution that satisfies the properties of critical section problem. Explain how semaphore can used as synchronization tool.
21. Consider a coke machine that has 10 slots. The producer is the delivery person and consumer is the student using the machine. It uses the following three semaphores :
    Semaphore mutex
    Semaphore fullBuffer /* Number of filled slots */

Semaphore emptyBuffer /* Number of empty slots */
The following operations are available on the semaphores: wait(semaphore s), signal(semaphore s).
Given functions delivery_person() and student():
i) What will be the initial values of the semaphores?
ii) Write a solution that guarantees the mutual exclusion and has no deadlocks.
iii) If the two wait() functions inside the student() section are interchanged [i.e., wait(fullbuffer) and wait(mutex) are interchanged], will your solution to the previous question will still be correct ? If not, explain your reason?

22. Consider the traffic deadlock depicted in the Figure given below.



a) Show that the four necessary conditions for deadlock hold in this example.
b) State a simple rule for avoiding deadlocks in this system.

23. A garden has four spades and three wheelbarrows.
Four gardeners are working:
- Gardener 1 needs to use two spades and two wheelbarrows.
- Gardener 2 needs to use two spades and one wheelbarrow.
- Gardener 3 needs to use one spades and two wheelbarrows.
- Gardener 4 needs to use three spades and three wheelbarrows.
At a certain point in time:
- Gardener 1 is using a wheelbarrow
- Gardener 2 is using a spade
- Gardener 3 is using a wheelbarrow
- Gardener 4 is using two spades
(a) Draw a resource allocation graph for this system
(b) Is the system in a safe state? Prove using the banker's algorithm.
(c) Gardener 1 wants to use another wheelbarrow. Should he be allowed to do this? Justify.

24. Explain the Banker's algorithm for deadlock avoidance.