Name: _____ **Roll No.:_____**

November 2023

BCA- III Semester

Introduction of Operating System

(BCA-DS-206/ BCA-17-201))

Common for BCA (General & Data Science)

Time: 3 Hours **Max Marks: 75**

Instructions:

1. *It is compulsory to answer all the questions (1.5 mark each) of **Part-A** in short.*
2. *Answer any four out of six questions from **Part-B** in detail.*
3. *Different **sub-parts** of a question are to be attempted adjacent to each other.*

PART-A

1. (a) Explain the difference between Logical and Physical address. [CO-1] (1.5)
(b) Define Thrashing [CO-1] (1.5)
(c) What are file attributes [CO-4] (1.5)
(d) Explain difference between Internal and External Fragmentation. [CO-5] (1.5)
(e) Define Turn Around Time [CO-1] (1.5)
(f) What is Context Switch? [CO-2] (1.5)
(g) Differentiate between LTS & STS [CO-6] (1.5)
(h) Mention the necessary conditions for Deadlock to occur. [CO-3] (1.5)
(i) What is Inter-Process Communication? [CO-1] (1.5)
(j) Why we use Multiprogramming? [CO-1] (1.5)

PART-B

2. (a) What is an Operating System? Explain Services provided by the Operating system. [CO-1] (10)
(b) Explain the different states of the process along with the process state transition diagram. [CO-2] (05)

3. Discuss the following in detail: [CO-2] (15)

- Real-time systems
- Parallel systems
- Inter-process communication.

4. (a) Consider Page Reference string 1,3,0,3,5,6,3,5,4. How many page faults while using FCFS, LRU and Optimal Page Replacement algorithm using 3 frames? [CO-3] (08)
(b) Explain Paging with an example. [CO-3] (07)

5. (a) Suppose that the following process arrive for execution at the time indicated.

Process	Burst Time	Arrival time	Priority
P1	8	0	3
P2	4	1	2
P3	1	4	1

What are the Average Waiting Time and Average Turnaround Time for these processes with:
(i) Preemptive SJF (ii) Preemptive Priority Scheduling Algo. (iii) Round Robin Algo. (TQ=2) [CO-2] (10)
(b) Explain the CPU Scheduling criteria. [CO-2] (05)

6. (a) Consider the following disk request sequence for a disk with 100 tracks 45, 21, 67, 90, 4, 50, 89, 52, 61, 87, 25 (Current head position is 50). What is the total head movement needed to satisfy the requests for the following algorithms, FCFS, SSTF, SCAN, C-SCAN, and LOOK? [CO-4] (10)

(b) Differentiate between Contiguous and Indexed file allocation methods with merit and demerits [CO-4] (05)

7. (a) What is Banker's algorithm in the operating system? Why is this algorithm named so? [CO-5] (05)

(b) Considering a system with five processes P0 through P4 and three resources of type A, B, C. Resource type A has 10 instances, B has 5 instances and type C has 7 instances. Suppose at time t0 following snapshot of the system has been taken: [CO-5] (10)

Process	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	

i. What will be the content of the Need matrix?

ii. Is the system in a safe state? If Yes, then what is the safe sequence?

*****************End of Question Paper**************

SOLUTION

1. (a) Explain the difference between Logical and Physical address. [CO-1] (1.5)

Ans:- Logical addresses are virtual memory addresses generated by the CPU during program execution. Physical addresses represent the actual locations in the physical memory where data is stored. The memory management unit (MMU) translates logical addresses to physical addresses for memory access.

(b) Define Thrashing. [CO-1] (1.5)

Ans: Thrashing occurs when a computer's performance severely drops due to excessive swapping of data between memory and disk, resulting in high disk I/O and little CPU utilization.

(c) What are file attributes? [CO-4] (1.5)

Ans: File attributes are metadata providing information about files, including permissions, size, creation/modification dates, and file type. They govern accessibility, visibility, and control over file operations within an operating system.

(d) Explain the difference between Internal and External Fragmentation. [CO-5] (1.5)

Ans: Internal fragmentation occurs when allocated memory within a partition remains unused due to space not being fully utilized by allocated blocks. External fragmentation arises when free memory exists, but it's scattered in small non-contiguous blocks, leading to inefficient use despite adequate total available space.

(e) Define Turn Around Time [CO-1] (1.5)

Ans: Turnaround time is the total elapsed time from job submission to its completion, encompassing waiting, execution, and waiting for output.

(f) What is Context Switch? [CO-2] (1.5)

Ans: A context switch is a process performed by an operating system to save and restore the state of a CPU so that it can execute multiple processes concurrently. It involves pausing the execution of one process, saving its state, and loading the state of another process to allow multitasking.

(g) Differentiate between LTS & STS [CO-6] (1.5)

Ans: The Long-Term Scheduler selects processes from the pool of incoming jobs, deciding which ones to admit to the system for processing based on system resource availability and scheduling policy, optimizing overall system performance. The Short-Term Scheduler, also known as the CPU Scheduler, selects from the ready queue of processes that are already in memory, determining which process will execute next on the CPU, focusing on efficient CPU utilization and minimizing response time.

(h) Mention the necessary conditions for Deadlock to occur. [CO-3] (1.5)

Ans: Deadlock occurs when the following four necessary conditions are simultaneously present in a system:

1. Mutual Exclusion: Resources cannot be simultaneously used or shared; at least one resource must be held in a non-shareable mode by a process.
2. Hold and Wait: Processes hold resources already allocated while waiting for additional resources that are held by other processes.
3. No Preemption: Resources cannot be forcibly taken from a process; they must be released voluntarily by the process holding them.
4. Circular Wait: A circular chain of two or more processes exists, where each process is waiting for a resource held by the next process in the chain, resulting in a cycle of waiting.

(i) What is Inter-Process Communication? [CO-1] (1.5)

Ans: Inter-Process Communication (IPC) refers to the mechanisms and techniques used by various processes running on a system to communicate and share data with each other. IPC enables processes to exchange information, synchronize their actions, and coordinate activities, facilitating collaboration and interaction between different processes within a system. Common IPC mechanisms include shared memory, message passing, pipes, sockets, and signals, allowing processes to communicate either within the same system or across different systems in a networked environment.

(j) Why we use Multiprogramming? **[CO-1] (1.5)**

Ans: Multiprogramming is utilized in computing systems for several reasons:

1. **Increased CPU Utilization:** By allowing multiple programs to reside in memory simultaneously, the CPU can switch between them, ensuring better utilization and minimizing idle time.
2. **Enhanced Throughput:** Multiprogramming enables overlapping of CPU and I/O operations, improving system throughput and overall performance.
3. **Resource Utilization:** It optimizes resource utilization by keeping the CPU busy even when some processes are waiting for I/O or other operations, thus maximizing the utilization of various system resources.
4. **Improved Response Time:** Users experience faster response times as the system can swiftly switch between different tasks, providing a perception of concurrent execution and responsiveness.
5. **Efficient Utilization of Memory:** Multiprogramming efficiently manages memory by allowing the sharing of memory space among multiple processes, reducing wastage and enhancing overall memory utilization.

PART-B

2. (a) What is an Operating System? Explain Services provided by the Operating system. [CO-1] (10)

Ans: An operating system (OS) is software that acts as an intermediary between users and the hardware of a computer system. It manages and coordinates all the resources and operations of the system, providing an interface for users to interact with the hardware efficiently.

Services provided by an Operating System:

1. **Process Management:** Allocating resources to processes, managing the execution of multiple processes concurrently, and handling process synchronization and communication.
2. **Memory Management:** Allocating and deallocating memory space for processes, managing memory utilization, and implementing virtual memory techniques.
3. **File System Management:** Organizing and managing files on storage devices, providing file access, directory management, and ensuring data integrity and security.
4. **Device Management:** Managing hardware devices such as printers, disks, keyboards, and networking devices, handling device drivers, and providing an interface for device communication.
5. **Security and Protection:** Enforcing access control, user authentication, and ensuring data security by implementing authentication mechanisms, encryption, and permissions.
6. **User Interface:** Providing a user-friendly interface for interaction, including command-line interfaces (CLI) and graphical user interfaces (GUI), allowing users to interact with the system and applications.

7. **Error Handling:** Detecting and handling errors and exceptions occurring in the system to prevent system crashes and ensure system stability.
8. **Resource Allocation and Scheduling:** Efficiently allocating resources like CPU time, memory, and I/O devices among processes and scheduling tasks for optimal system performance.

An operating system plays a crucial role in managing and coordinating various system resources, providing an environment for efficient and secure execution of applications, and ensuring the overall functionality and usability of the computer system.

(b) Explain the different states of the process along with the process state transition diagram.

[CO-2] (05)

Ans: Processes in an operating system go through various states during their lifecycle. The typical process states include:

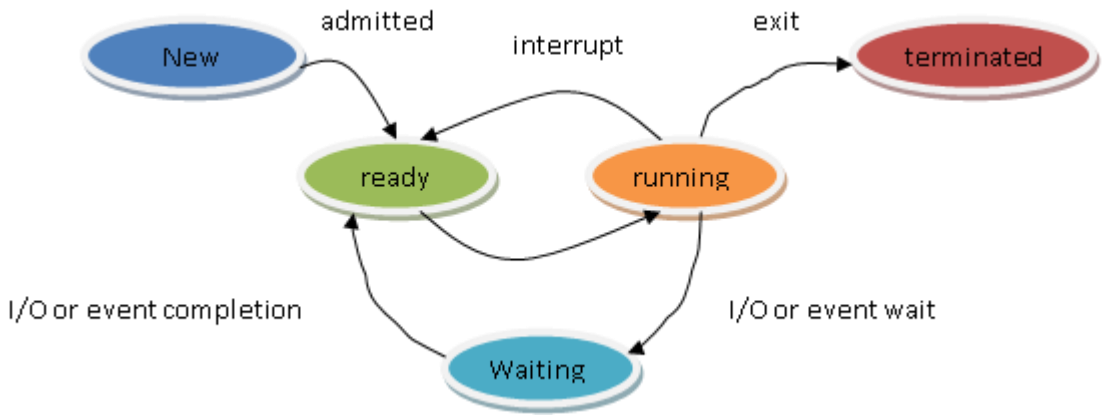
1. New:
- The process is being created or initialized.
- Resources are being allocated for the process.

2. Ready:
- The process is ready to execute and is waiting for CPU time.
- It's in the queue of ready-to-run processes.

3. Running:
- The CPU is executing the instructions of the process.
- The process is in execution mode.

4. Blocked (or Waiting):
- The process is waiting for an event or resource (e.g., I/O operation) to proceed.
- It's not using CPU time and is temporarily inactive.

5. Terminated (or Exit):
- The process has completed its execution.
- It's no longer running and has released all resources.



Process State Diagram

3. Discuss the following in detail: **[CO-2] (15)**
- Real-time systems
- Parallel systems
- Inter-process communication.

Ans: **Real-time systems**

Real-time systems are computer systems designed to handle tasks with strict timing constraints, ensuring that operations are performed within specified time frames. These systems are used in scenarios where

timely and predictable responses are critical, such as control systems, embedded systems, robotics, and critical infrastructure.

Characteristics of Real-time Systems:

1. **Deadline Sensitivity:** Tasks in real-time systems have specific deadlines within which they must complete their execution. Missing deadlines can lead to system failures or performance degradation.
2. **Predictability:** Real-time systems require predictable behavior, where response times and task completion times are deterministic and known in advance.
3. **Priority-Based Scheduling:** Tasks are assigned priorities based on their urgency and deadlines. Higher priority tasks are executed first to meet their deadlines.
4. **Reliability:** Real-time systems emphasize reliability and fault tolerance to ensure continuous operation even in the presence of faults or failures.
5. **Hard and Soft Real-Time Systems:** Hard real-time systems have strict and rigid timing requirements, while soft real-time systems are more flexible and allow some degree of timing violations.
6. **Embedded Systems:** Many real-time systems are embedded within larger systems, controlling specific functions like automotive control systems, medical devices, and aerospace systems.

Types of Real-time Systems:

1. **Hard Real-Time Systems:** Systems where missing deadlines can lead to catastrophic consequences. Examples include aircraft control systems and medical equipment.
2. **Soft Real-Time Systems:** Systems where missing deadlines may lead to reduced system performance or quality but not catastrophic consequences. Examples include multimedia streaming and online gaming.

4. (a) Consider Page Reference string 1,3,0,3,5,6,3,5,4. How many page faults while using FCFS, LRU and Optimal Page Replacement algorithm using 3 frames? [CO-3] (08)

Ans: 1. FIFO

Summary - FIFO algorithm

- Total frames: 3
- Algorithm: FIFO
- Reference string length: 9 references
- String: 1 3 0 3 5 6 3 5 4

Solution visualization

t	0	1	2	3	4	5	6	7	8	9
ref		**1**	**3**	**0**	**3**	**5**	**6**	**3**	**5**	**4**
f		1	3	0	0	5	6	3	3	4
f			1	3	3	0	5	6	6	3
f				1	1	3	0	5	5	6
hit		X	X	X	✓	X	X	X	✓	X
v						1	3			5

- Total references: 9
- Total distinct references: 6
- Hits: 2
- Faults: 7
- **Hit rate:** 2/9 = **22.222222222222%**
- **Fault rate:** 7/9 = **77.777777777778%**

2. LRU

Summary - LRU algorithm

- Total frames: 3
- Algorithm: LRU
- Reference string length: 9 references
- String: 1 3 0 3 5 6 3 5 4

Solution visualization

t	0	1	2	3	4	5	6	7	8	9
ref		**1**	**3**	**0**	**3**	**5**	**6**	**3**	**5**	**4**
f		1	3	0	3	5	6	3	5	4
f			1	3	0	3	5	6	3	5
f				1	1	0	3	5	6	3
hit		✗	✗	✗	✓	✗	✗	✓	✓	✗
v						1				6

- Total references: 9
- Total distinct references: 6
- Hits: 3
- Faults: 6
- **Hit rate:** 3/9 = **33.333333333333%**
- **Fault rate:** 6/9 = **66.666666666667%**

3. OPTIMAL

Summary - OPT algorithm

- Total frames: 3
- Algorithm: OPT
- Reference string length: 9 references
- String: 1 3 0 3 5 6 3 5 4

Solution visualization

t	0	1	2	3	4	5	6	7	8	9
ref		**1**	**3**	**0**	**3**	**5**	**6**	**3**	**5**	**4**
f		1	3	0	0	5	6	6	6	4
f			1	3	3	3	5	5	5	5
f				1	1	1	3	3	3	3
hit		✗	✗	✗	✓	✗	✗	✓	✓	✗
v							1			6

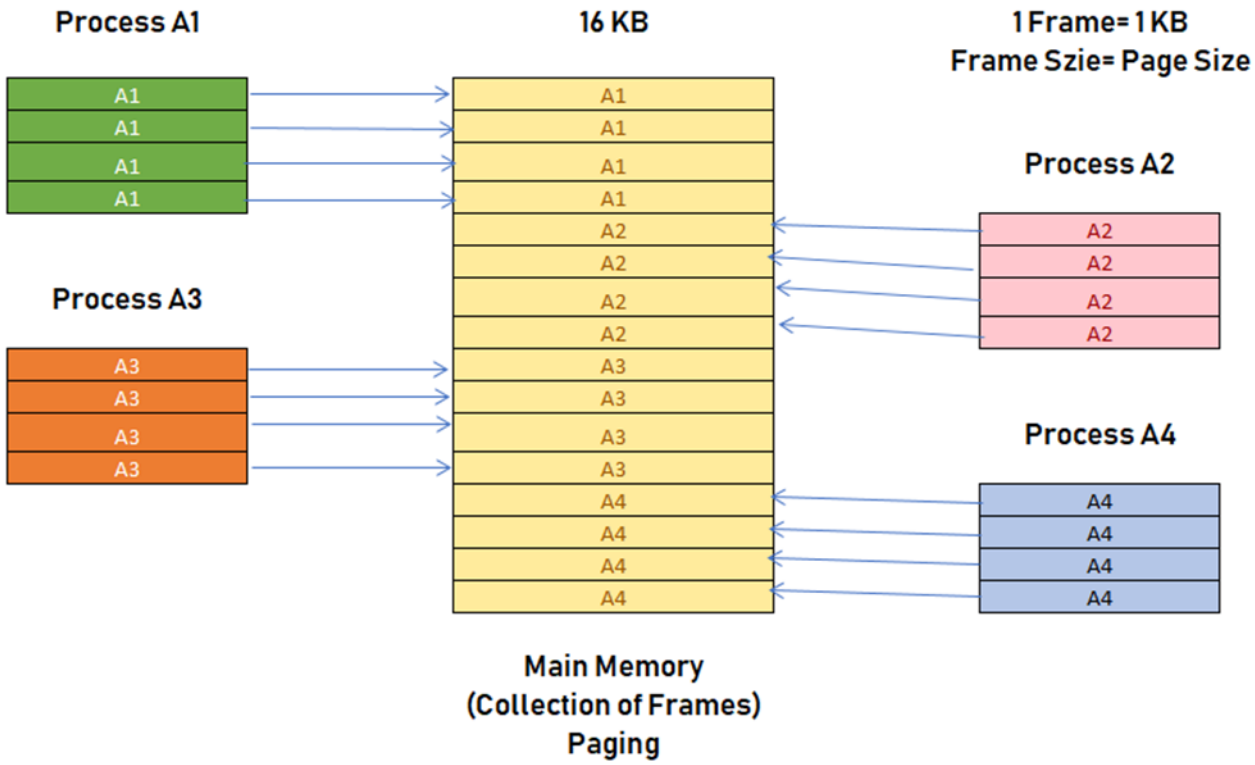
- Total references: 9
- Total distinct references: 6
- Hits: 3
- Faults: 6
- **Hit rate:** 3/9 = **33.333333333333%**
- **Fault rate:** 6/9 = **66.666666666667%**

(b) Explain Paging with an example. **[CO-3] (07)**

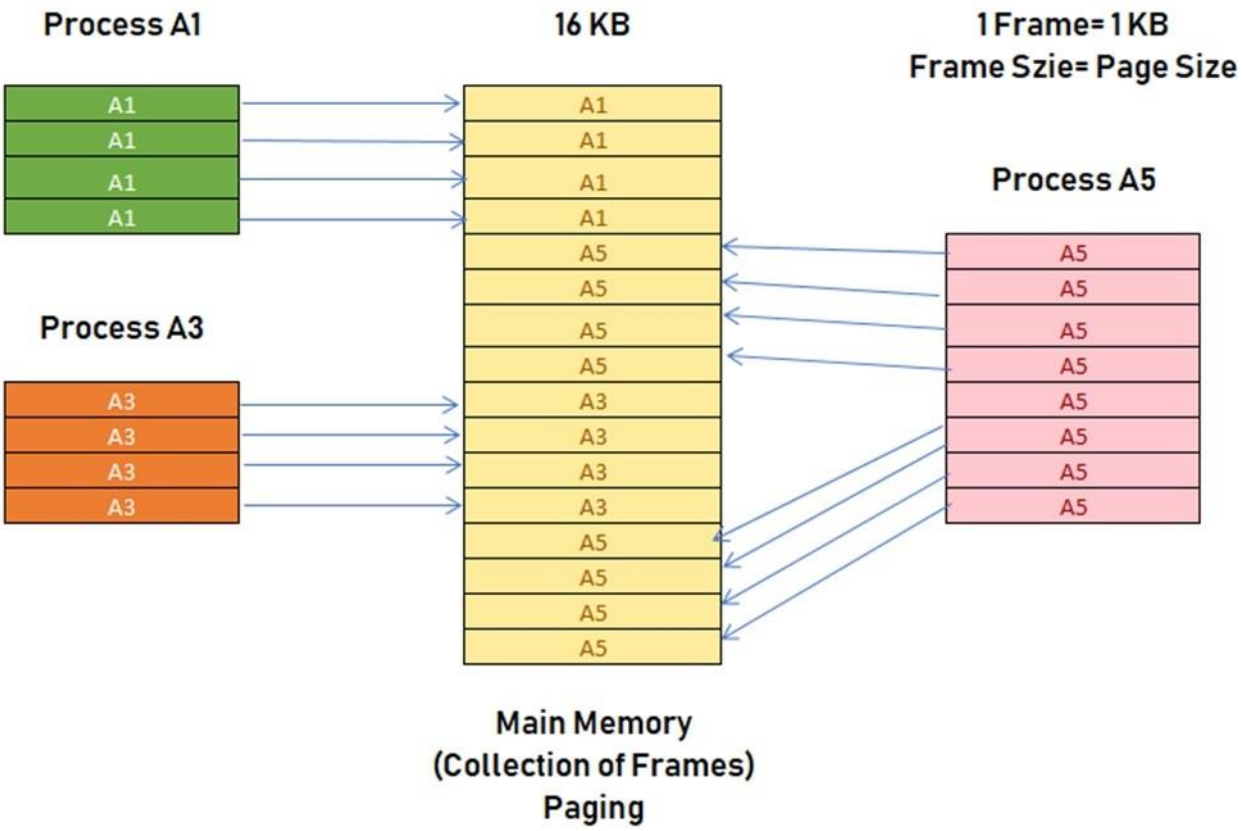
Ans: **Paging** is a storage mechanism that allows OS to retrieve processes from the secondary storage into the main memory in the form of pages. In the Paging method, the main memory is divided into small fixed-size blocks of physical memory, which is called frames. The size of a frame should be kept the same as that of a page to have maximum utilization of the main memory and to avoid external fragmentation. Paging is used for faster access to data, and it is a logical concept.

Example of Paging in OS

For example, if the main memory size is 16 KB and Frame size is 1 KB. Here, the main memory will be divided into the collection of 16 frames of 1 KB each.
There are 4 separate processes in the system that is A1, A2, A3, and A4 of 4 KB each. Here, all the processes are divided into pages of 1 KB each so that operating system can store one page in one frame.
At the beginning of the process, all the frames remain empty so that all the pages of the processes will get stored in a contiguous way.



In this example you can see that A2 and A4 are moved to the waiting state after some time. Therefore, eight frames become empty, and so other pages can be loaded in that empty blocks. The process A5 of size 8 pages (8 KB) are waiting in the ready queue.



In this example, you can see that there are eight non-contiguous frames which is available in the memory, and paging offers the flexibility of storing the process at different places. This allows us to load the pages of process A5 instead of A2 and A4.

5. (a) Suppose that the following process arrives for execution at the time indicated.

Process	Burst Time	Arrival time	Priority
P1	8	0	3
P2	4	1	2
P3	1	4	1

What are the Average Waiting Time and Average Turnaround Time for these processes with:
(ii) Preemptive SJF (ii) Preemptive Priority Scheduling Algo. (iii) Round Robin Algo. (TQ=2)
[CO-2] (10)

Ans: FIRST COME FIRST SERVE

Input

Process:	A	B	C
Arrival time:	0	1	4
Service time:	8	4	1
Priority:	3	2	1

Solution

t:	0	1	2	3	4	5	6	7	8	9	10	11	12
CPU	A 1/8	A 2/8	A 3/8	A 4/8	A 5/8	A 6/8	A 7/8	A 8/8	B 1/4	B 2/4	B 3/4	B 4/4	C 1/1
	↑ A	↑ B			↑ C								

Order of service (table):

Process	Arrival	Service	Priority	Started	Completion	turnaround time (TAT)	waiting time (WAT)
A	0	8	3	0	8	8	0
B	1	4	2	8	12	11	7
C	4	1	1	12	13	9	8
Avg:	-	4.33333	-	-	-	9.33333	5

Order of service (descriptive):

- Process **A**: arrived at 0 ms, service time:8 priority: 3 completed at **8**: TAT_A= 8 ms and WAT_A= 0 ms
- Process **B**: arrived at 1 ms, service time:4 priority: 2 completed at **12**: TAT_B= 11 ms and WAT_B= 7 ms
- Process **C**: arrived at 4 ms, service time:1 priority: 1 completed at **13**: TAT_C= 9 ms and WAT_C= 8 ms

Statistics:

- First request arrived at: 0 ms
- Last request completed at: 13 ms
- Total service time for all processes = **13** ms
- Average turnaround time (ATAT) = **9.33333** ms
- Average waiting time (AWAT) = **5** ms

CPU State

Process	Level	Start	Stop	Q1	TAT
A	1	0	8	$\{B_{0/4}, C_{0/1}\}$	8
B	1	8	12	$\{C_{0/1}\}$	11
C	1	12	13		9

2. PREEMPTIVE PRIORITY SCHEDULING

Input

Process:	A	B	C
Arrival time:	0	1	4
Service time:	8	4	1
Priority:	3	2	1

Solution

t:	0	1	2	3	4	5	6	7	8	9	10	11	12
CPU	A 1/8	B 1/4	B 2/4	B 3/4	C 1/1	B 4/4	A 2/8	A 3/8	A 4/8	A 5/8	A 6/8	A 7/8	A 8/8

Order of service (table):

Process	Arrival	Service	Priority	Started	Completion	turnaround time (TAT)	waiting time (WAT)
C	4	1	1	4	5	1	0
B	1	4	2	1	6	5	1
A	0	8	3	0	13	13	5
Avg:	-	4.33333	-	-	-	6.33333	2

Order of service (descriptive):

- Process **C**: arrived at 4 ms, service time:1 priority: 1 completed at **5**: TAT_C= 1 ms and WAT_C= 0 ms
- Process **B**: arrived at 1 ms, service time:4 priority: 2 completed at **6**: TAT_B= 5 ms and WAT_B= 1 ms
- Process **A**: arrived at 0 ms, service time:8 priority: 3 completed at **13**: TAT_A= 13 ms and WAT_A= 5 ms

Statistics:

- First request arrived at: 0 ms
- Last request completed at: 13 ms
- Total service time for all processes = **13** ms
- Average turnaround time (ATAT) = **6.33333** ms
- Average waiting time (AWAT) = **2** ms

CPU State

Process	Level	Start	Stop	Q1	TAT
A	1	0	1	$\{B_{0/4}, A_{1/8}\}$	
B	1	1	4	$\{C_{0/1}, A_{1/8}, B_{3/4}\}$	
C	1	4	5	$\{B_{3/4}, A_{1/8}\}$	1
B	1	5	6	$\{A_{1/8}\}$	5
A	1	6	13		13

3. ROUND ROBIN SCHEDULING (Time Quantum=02)

Input

Process:	A	B	C
Arrival time:	0	1	4
Service time:	8	4	1
Priority:	3	2	1

Solution

t:	0	1	2	3	4	5	6	7	8	9	10	11	12
CPU	A 1/8	A 2/8	B 1/4	B 2/4	A 3/8	A 4/8	C 1/1	B 3/4	B 4/4	A 5/8	A 6/8	A 7/8	A 8/8

Order of service (table):

Process	Arrival	Service	Priority	Started	Completion	turnaround time (TAT)	waiting time (WAT)
C	4	1	1	6	7	3	2
B	1	4	2	2	9	8	4
A	0	8	3	0	13	13	5
Avg:	-	4.33333	-	-	-	8	3.66667

Order of service (descriptive):

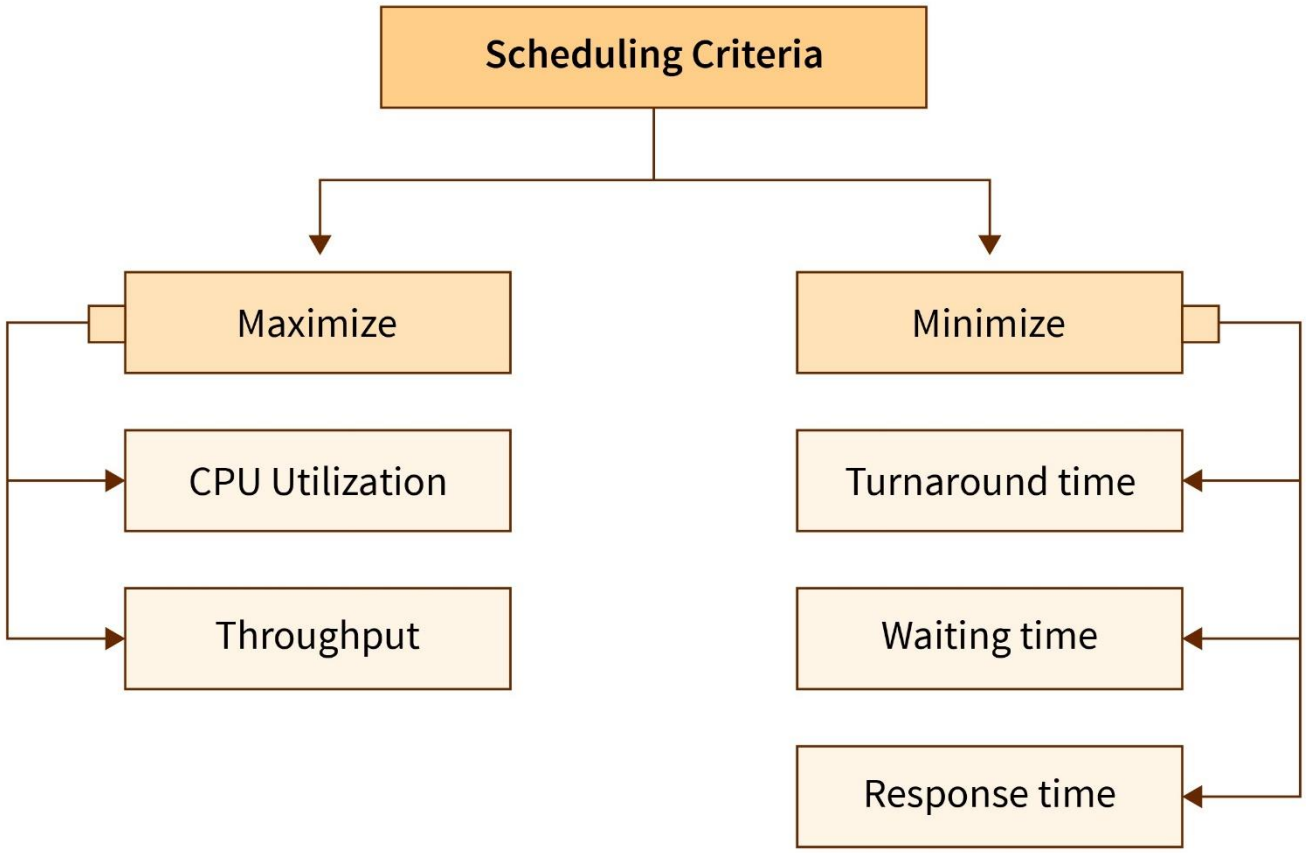
- Process **C**: arrived at 4 ms, service time:1 priority: 1 completed at **7**: TAT_C= 3 ms and WAT_C= 2 ms
- Process **B**: arrived at 1 ms, service time:4 priority: 2 completed at **9**: TAT_B= 8 ms and WAT_B= 4 ms
- Process **A**: arrived at 0 ms, service time:8 priority: 3 completed at **13**: TAT_A= 13 ms and WAT_A= 5 ms

Statistics:

- First request arrived at: 0 ms
- Last request completed at: 13 ms
- Total service time for all processes = **13** ms
- Average turnaround time (ATAT) = **8** ms
- Average waiting time (AWAT) = **3.66667** ms

(b) **Explain the CPU Scheduling criteria.** [CO-2] (05)

Ans:



(b) Differentiate between Contiguous and Indexed file allocation methods with merit and demerits

Ans: **Contiguous File Allocation:**

In the contiguous file allocation method, files are stored in consecutive blocks on the storage device. Each file occupies a contiguous block of space, meaning the blocks are allocated continuously without any gaps between them. This method is commonly used in systems that use disk storage.

Merits:

1. **Sequential Access:** Reading or writing data from a file is faster since the blocks are contiguous. This benefits applications that require sequential access, like media streaming.
2. **Simplicity:** It's relatively simple to implement and manage, requiring minimal metadata to track file locations.

Demerits:

1. **Fragmentation:** Over time, as files are created, deleted, and modified, free space becomes fragmented. This leads to external fragmentation, making it challenging to find contiguous blocks large enough for new files.
2. **Wastage of Space:** Fixed-size allocation may lead to internal fragmentation, where the last block of a file might not be fully utilized, wasting storage space.

Indexed File Allocation:

In indexed file allocation, each file has its own index block containing pointers to various data blocks spread across the storage space. This index structure holds information about the location of each block of the file.

Merits:

1. **No Fragmentation:** Since files are not stored contiguously, there is no external fragmentation. Files can be scattered across the storage device without affecting performance.
2. **Efficient Access:** Random access to files is more efficient as the index contains pointers to different blocks, allowing quick access to specific parts of a file.

Demerits:

1. **Overhead:** Maintaining and updating the index adds overhead in terms of storage space and processing power. Every file requires an index, which could consume additional space and impact performance.
2. **Complexity:** Implementing and managing the index structure can be complex, especially as the storage size increases or when handling a large number of files.

[CO-4] (05)

7. (a) What is Banker's algorithm in the operating system? Why is this algorithm named so? [CO-5] (05)

Ans: The Banker's algorithm is a resource allocation and deadlock avoidance algorithm used in operating systems to manage the allocation of resources among multiple processes in a safe manner. It ensures that the system does not enter a deadlock state by carefully allocating resources based on the maximum needs of processes.

The algorithm operates by simulating the allocation of resources to processes and checks whether this allocation will leave the system in a safe state. It considers the maximum and current resource allocation to processes and the maximum needs of each process to determine whether a particular request for resources can be granted without leading the system into an unsafe state, where a deadlock might occur.

The name "Banker's algorithm" comes from the analogy of a banker who only lends money if it leaves enough reserves to meet future needs and avoid bankruptcy. Similarly, this algorithm only allocates resources if it ensures that the system can always satisfy the future resource requests of processes, preventing a deadlock situation.

The concept is based on the idea of having a centralized authority (the banker) that manages and allocates resources in a way that guarantees the safety of the system, avoiding deadlock scenarios where processes are stuck waiting indefinitely for resources held by others.

Overall, the Banker's algorithm is essential in ensuring resource allocation in a way that prevents deadlock while efficiently managing available resources among multiple processes.

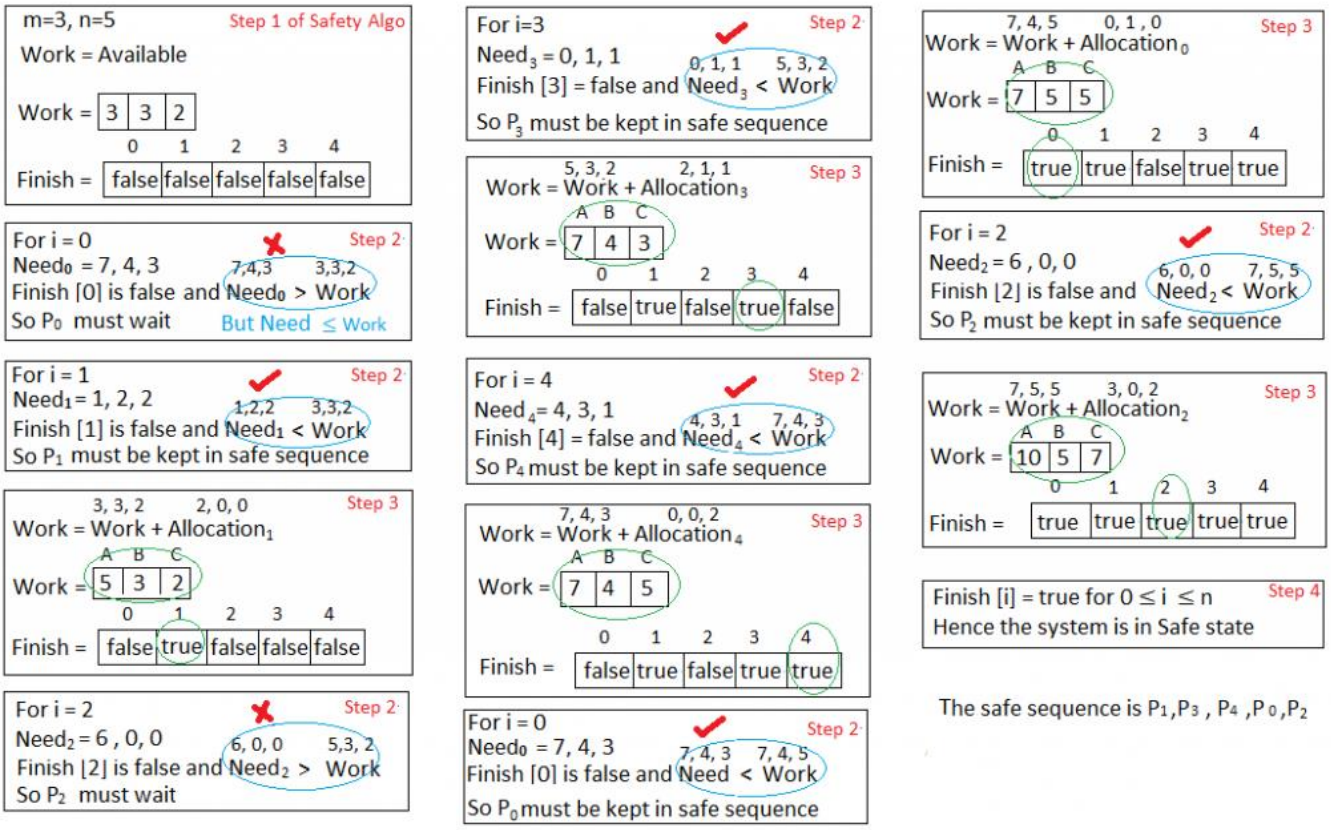
(b) Considering a system with five processes P0 through P4 and three resources of type A, B, C. Resource type A has 10 instances, B has 5 instances and type C has 7 instances. Suppose at time t0 following snapshot of the system has been taken: [CO-5] (10)

Process	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	

Q.1: What will be the content of the Need matrix?
Need [i, j] = Max [i, j] – Allocation [i, j]
So, the content of Need Matrix is:

Process	Need		
	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

Q.2: Is the system in a safe state? If Yes, then what is the safe sequence?
Applying the Safety algorithm on the given system,



Following is the SAFE Sequence

P1 -> P3 -> P4 -> P0 -> P2