AOS Assignment No. 3

❖ 2 Marks Questions

Q.1.) What is Memory Management?

Ans :- Memory management is the process of handling and optimizing a computer's memory usage, including allocation, tracking, and deallocation of memory for programs and processes to ensure efficient performance and avoid memory leaks or crashes.

Q.2.) What is Best fit?

Ans :- Best Fit is a memory allocation strategy in which the system assigns the smallest available partition that is large enough to accommodate a process. This minimizes wasted space but can lead to fragmentation over time.

Q.3.) What is Paging?

Ans :- Paging is a memory management technique that divides processes and physical memory into fixed-sized blocks called pages and frames, respectively. It eliminates external fragmentation and allows efficient memory allocation by mapping logical addresses to physical addresses using a page table.

Q.4.) What is the need of Paging?

Ans :- Paging is needed to efficiently manage memory by eliminating external fragmentation, enabling better utilization of available space, and allowing processes to use non-contiguous memory. It simplifies memory allocation, supports virtual memory, and ensures faster access through mapping logical addresses to physical memory using a page table.

Q.5.) What is Memory Management Unit?

Ans :- The Memory Management Unit (MMU) is a hardware component that handles virtual-to-physical address translation, enforces memory protection, and manages paging or segmentation. It enables efficient memory access, multitasking, and security in an operating system.

❖ 10 Marks Questions

Q.1.) Explain Paging in Operating System with an example.

Ans :- Paging is a memory management technique that eliminates external fragmentation by dividing both the process and physical memory into fixed-sized blocks. The process is divided into **pages**, while the main memory is divided into **frames** of the same size. The operating system maintains a **page table** that maps process pages to available frames in memory.

Example of Paging

Scenario:

Assume we have a process that requires **8 KB** of memory, and the system has a **page size of 2 KB**. This means the process will be divided into:

$$\frac{8KB}{2KB} = 4 \text{ pages}$$

Similarly, the physical memory has available **frames** of **2 KB** each. The OS places the process pages into available frames, not necessarily in a contiguous manner.

Page Table Mapping:

Page Number	Frame Number
Page 0	Frame 5
Page 1	Frame 2
Page 2	Frame 7
Page 3	Frame 3

If a process wants to access an address in **Page 2**, the OS looks up the **page table** and finds it stored in **Frame 7**. The MMU (Memory Management Unit) translates the logical address into a physical address using this mapping.

Advantages of Paging

- Eliminates external fragmentation
- Allows non-contiguous memory allocation
- Supports virtual memory efficiently

Disadvantages of Paging

- Adds overhead due to page table management
- May cause internal fragmentation if pages are not fully used

Q.2.) How the data is being stored in a computer system? Give an example.

Ans :- Data in a computer system is stored using **binary representation (0s and 1s)** in different types of storage devices, including **primary memory (RAM, cache), secondary storage (HDD, SSD), and tertiary storage (CDs, cloud storage, etc.)**. The data is structured and managed using file systems and storage hierarchies.

Example: Storing a Text File

1. **User Saves a File:** A user writes "Hello" in a text file and saves it.

2. **Binary Conversion:** The computer converts "Hello" into **ASCII binary values**:
    - H → 01001000
    - e → 01100101
    - l → 01101100
    - l → 01101100
    - o → 01101111
3. **Storage on Disk:**
    - If stored on an **HDD**, the magnetic disk uses magnetized and non-magnetized spots to represent binary values.
    - If stored on an **SSD**, electrical charges in NAND flash memory store the data.
4. **Retrieval:** When the user opens the file, the binary data is read from storage, converted back into text, and displayed on the screen.

Storage Types in a Computer System:

1. **Primary Storage (Volatile)**
    - RAM (Random Access Memory) – Temporary memory used for active processes.
    - Cache – Faster memory close to the processor for quick data access.
2. **Secondary Storage (Non-Volatile)**
    - HDD (Hard Disk Drive) – Uses spinning magnetic disks.
    - SSD (Solid State Drive) – Uses flash memory for faster access.
3. **Tertiary & Cloud Storage**
    - CD/DVDs, USB Drives, and Cloud storage for backup and remote access.

Q.3.) What is Fixed partition? Explain.

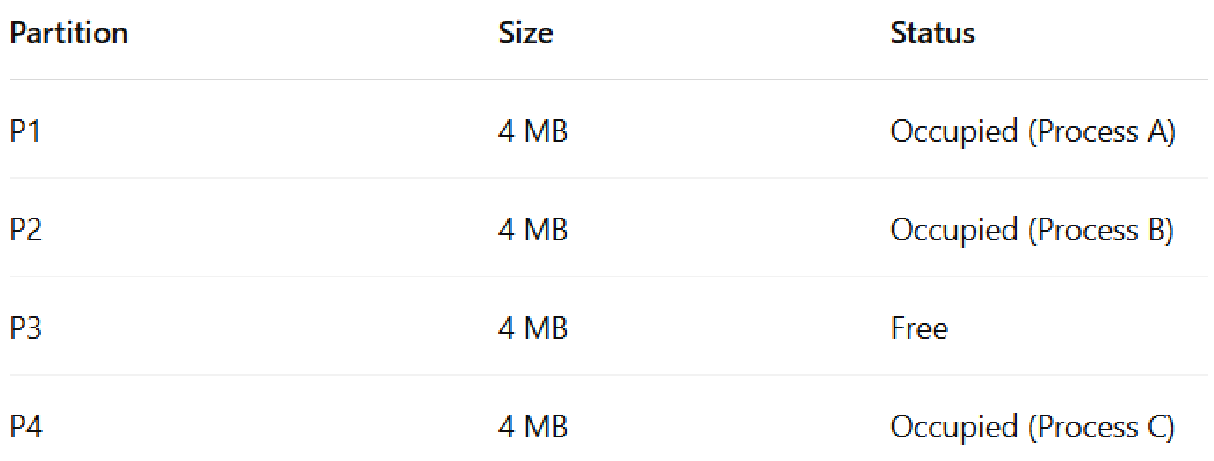
Ans :- **Fixed Partitioning** is a memory management technique where the main memory is divided into a **fixed number of partitions** of predefined sizes at system startup. Each partition can hold only one process at a time.

Explanation:

- The **partition sizes are set during system boot** and cannot be changed dynamically.
- If a process is **smaller** than the assigned partition, **internal fragmentation** occurs (wasted space within the partition).
- If a process is **larger** than any available partition, it **cannot be loaded into memory**.
- The **operating system maintains a table** to track which partitions are free or occupied.

Example of Fixed Partitioning

Consider a system with **16 MB** of RAM divided into four fixed partitions:



Partition	Size	Status
P1	4 MB	Occupied (Process A)
P2	4 MB	Occupied (Process B)
P3	4 MB	Free
P4	4 MB	Occupied (Process C)

- If a **2 MB process** wants to enter, it must occupy an entire **4 MB partition**, leading to **internal fragmentation** (2 MB wasted).
- If an **8 MB process** arrives, it **cannot be allocated**, as no single partition is large enough.

Advantages:

✅ Simple to implement.
✅ Fast allocation and deallocation.
✅ Low overhead for memory management.

Disadvantages:

❌ Causes **internal fragmentation** due to unused space in partitions.
❌ **Rigid and inefficient** as partition sizes are fixed.
❌ **Process size limitation** – Large processes may not fit.

Q.4.) Write the Advantages of Dynamic Partitioning over fixed partitioning.

Ans :- **Advantages of Dynamic Partitioning Over Fixed Partitioning**

1. **Efficient Memory Utilization**
   - Dynamic partitioning allocates memory as needed, reducing **internal fragmentation** (unused space within partitions).
   - In contrast, fixed partitioning wastes memory when a process does not fully use its allocated partition.
2. **No Predefined Partition Sizes**
   - Unlike fixed partitioning, where memory is divided into **pre-set sizes**, dynamic partitioning creates partitions **dynamically** based on process size.
   - This prevents situations where large processes cannot be accommodated due to fixed partition limits.
3. **Accommodates Larger Processes**
   - Since partitions are created dynamically, processes of **any size** (within available memory) can be allocated, unlike fixed partitioning where large processes may not fit in predefined partitions.
4. **Reduces Process Waiting Time**

    - In fixed partitioning, a process may have to wait if a suitable-sized partition is not available.
    - In dynamic partitioning, the process can be allocated **any free memory block** of appropriate size, reducing wait time.
5. **Optimized Multiprogramming**
    - More processes can be loaded into memory because partitions are adjusted based on process requirements, leading to **better CPU utilization**.
6. **Minimizes Wastage**
    - **Fixed partitioning** results in **unused** memory space (internal fragmentation), while **dynamic partitioning** minimizes waste by assigning exact memory sizes.

Limitation of Dynamic Partitioning

Despite its advantages, dynamic partitioning can cause **external fragmentation**, where free memory blocks become scattered, making it difficult to allocate large contiguous memory blocks to processes. This issue is mitigated using **compaction** (memory rearrangement).

Q.5.) Process requests are given as.

25K, 50K, 100K, 75K

Block Size are:

50K, 75K, 150, 175K,300K

Ans :- To allocate the processes to memory blocks, we can use different memory allocation strategies such as **First Fit, Best Fit, and Worst Fit**. Let's analyze how the processes will be assigned using **First Fit** and **Best Fit**.

Given Data:

Process Requests:

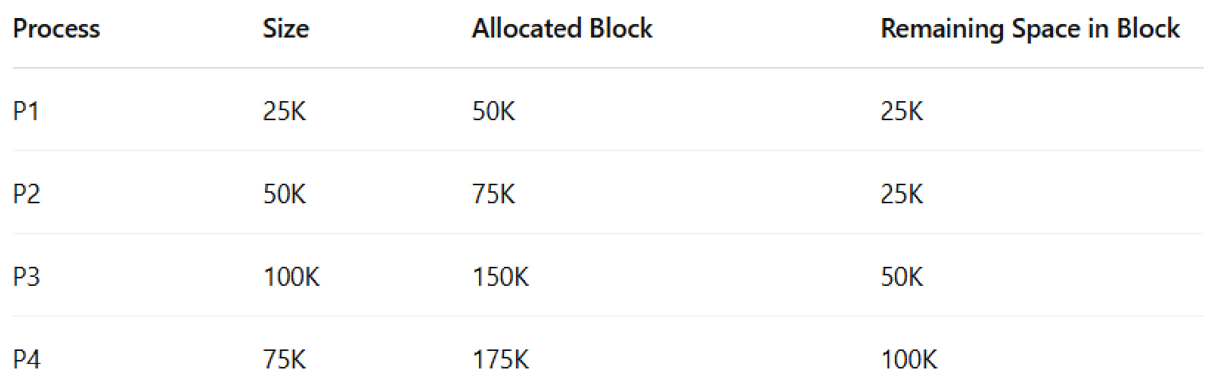
1. 25K

2. 50K
3. 100K
4. 75K

Memory Blocks:

1. 50K
2. 75K
3. 150K
4. 175K
5. 300K

1. First Fit Allocation (Allocates the first available block that fits)



Process	Size	Allocated Block	Remaining Space in Block
P1	25K	50K	25K
P2	50K	75K	25K
P3	100K	150K	50K
P4	75K	175K	100K

- Unallocated Block(s): 300K (Still free)
- Internal Fragmentation: (25K in Block 1, 25K in Block 2, 50K in Block 3, 100K in Block 4)

2. Best Fit Allocation (Allocates the smallest block that fits)

Process	Size	Allocated Block	Remaining Space in Block
P1	25K	50K	25K
P2	50K	75K	25K
P3	100K	150K	50K
P4	75K	175K	100K

- Unallocated Block(s): 300K (Still free)
- Internal Fragmentation: (25K in Block 1, 25K in Block 2, 50K in Block 3, 100K in Block 4)

Q.6.) Determine the algorithm which can optimally satisfy this requirement and why?

1. First Fit algorithm

2. Best Fit Algorithm

3. Neither of the two

4. Both

Ans :- We analyzed the **First Fit** and **Best Fit** allocation strategies and found that both produced the same results in this case. Now, let's determine which algorithm is **optimal** for memory utilization.

Comparing the Algorithms:

1. **First Fit:**
   - Scans memory from the beginning and assigns the first available block that fits.
   - **Fast execution** but may leave small unusable memory holes (**fragmentation**).

2. **Best Fit:**
    - Searches for the **smallest available block** that fits the process.
    - **Reduces internal fragmentation** but may lead to **external fragmentation** as it leaves many small unusable gaps.
3. **Neither of the Two:**
    - If **external fragmentation is a major concern**, then **compaction** or **paging** might be required.
4. **Both:**
    - If both algorithms produce the **same allocation** and **efficient memory usage**, either can be used.

Optimal Choice:

Best Fit Algorithm is the better choice because it **minimizes wasted space** by selecting the smallest available block that fits the process.

Reason:

- **Reduces internal fragmentation** (unused space inside a block).
- **Allocates memory efficiently**, ensuring more processes can fit.
- **In this specific case**, Best Fit does not perform worse than First Fit and has the potential to leave larger blocks free for future allocations.