# ABSTRACT

Memory management is an operating system feature that handles or controls primary memory and transfers processes between main memory and disc during execution. Memory management keeps track of any memory location, regardless of whether it is used by a process or is free. It determines how much memory should be allocated to each operation. It determines which processes will be given memory and when. It keeps track of when memory is freed or unallocated and updates the status accordingly.

# INTRODUCTION

Memory management is a method for controlling and coordinating computer memory, as well as assigning blocks to various running programs for overall system performance optimization. It's found in the Operating System's (OS) hardware, as well as in software and programs.

***LITERATURE REVIEW:***

The operating system (OS) is the software that controls a device's hardware and software and delivers shared services to the applications. But for firmware, all programs need the operating system to function. The primary function of a computer machine is to run programs. At the time of execution, these programs and their accusing data must be located in main memory.

Memory management is one of the OS's key tasks, and is accomplished by the use of the memory management unit (MMU). The MMU is a program feature of the operating system that is located in the kernel. The operating system controls all kinds of memory, which are classified as main and secondary.

The primary memory is responsible for storing data and programs used for programs execution, while secondary memory is non-volatile and provides long-term data and program storage.

Memory control includes the administration of hardware, operating systems, virtual memory, and programs memory. Memory is efficiently managed by the operating system by a memory management unit (MMU). Since running programs can only recognize logical addresses, the MMU converts virtual addresses to physical addresses. Memory, on the other hand, is minimal. Since main memory is limited, the operating system generates virtual memory to substitute for the need for more memory.

In virtual memory, virtual addresses can be identified. Virtual memory is a term that gives programs a wide memory address space to allow for multiprogramming. That implies that the processor works under the assumption of accessing a broad addressable space, which is not the case, a term endorsed by the operating system. The CPU is essential in generating virtual addresses, which are then translated into physical addresses by the MMU.

It is worth noting that these addresses must be identical when a programs is loaded for execution. Many of these tasks are handled by the operating system through the memory manager. These addresses, however, change as the programs runs. The operating system makes use of the concepts of paging, relocation, and segmentation to ensure equal allocation and system throughput.

Allocating and de-allocating memory leaves gaps in memory, resulting in internal and external memory fragmentation. Internal and external fragmentation, on the other hand, can be avoided by using a variety of resource allocation techniques to maximize memory use.

The OS performs all of these memory control tasks by using the MMU, which is an integral device module in the OS's kernel.

To achieve the overall memory management goal, the OS system, takes on a supervisory role through the memory manager.

The only memory management approach that does not provide a linear and contiguous address space to the user's program is segmented memory. Swapping is a technique for temporarily moving a process from the main memory to the backing store. It will be brought back into memory later for further execution. Memory allocation is the method of allocating memory or space to computer programs.

When we are purchasing a new phone or laptop, memory is one of the most important factors to consider. Users want a non-volatile memory that is private, infinitely large, and infinitely fast. However, this is too costly for us, and existing technology does not allow us to do so. The memory manager is a function of the operating system. The memory manager's function is to effectively manage memory which involve  Keeping track of which pieces of memory are currently in use.

* Processes should be given memory.
* After the processes have used up all of the memory, free it up.
* To prevent the machine from being stuck in a deadlock, the OS and memory manager must handle this memory between applications. If it does get stuck in a deadlock, the operating system knows how to deal with it.

**Need of memory management in Operating system:**

**Reallocation:**

Several processes are usually operating in the background while we operate on a multiprogramming device. We can't predict which other programs will be in main memory when we execute our processes, so the memory manager manages the executed and pending processes and allocates and frees memory accordingly, ensuring that processes run smoothly and efficiently.

**Safeguard:**

When multiple processes are running at the same time, one of them can write into another's address space. As a result, each process must be shielded from unnecessary intervention from other processes.

**Memory paging:**

It's possible that dynamic memory allocation will result in non-contiguous memory usage. In that case, we use a method known as paging to effectively control memory. Paging is a memory management technique that allows for non-contiguous memory allocation. The paging technique, in other words, is the mapping from a virtual to a physical address.

In paging, the mapping is stored in a page table. A page table is a virtual memory data structure that stores the mapping between logical and physical addresses. The CPU is in charge of creating logical addresses, which are then used by the processes.

 **OBJECTIVES OF MEMORY MANAGEMENT SYSTEM:**

Objectives of MMS are:

***Relocation*:**

* ***Relocatability*** - the ability to move process around in memory without it affecting its execution
* OS manages memory, not programmer, and processes may be moved around in memory
* MM must convert program's logical addresses into physical addresses
* Process's first address is stored as virtual address 0
* ***Static Relocation*** - Program must be relocated before or during loading of process into memory. Program must always be loaded into same address space in memory, or relocator must be run again.
* ***Dynamic Relocation*** - Process can be freely moved around in memory. Virtual-tophysical address space mapping is done at run-time.

***Protection:***

* ***Write Protection*** - to prevent data & instructions from being over-written.
* ***Read Protection*** - To ensure privacy of data & instructions.
* OS needs to be protected from user processes, and user processes need to be protected from each other.
* Memory protection (to prevent memory overlaps) is usually supported by the hardware (limit registers), because most languages allow memory addresses to be computed at run-time.

***Sharing:***

* Sometimes distinct processes may need to execute the same process (e.g., many users executing same editor), or even the same data (when one process prepares data for another process).
* When different processes signal or wait the same semaphore, they need to access the same memory address.
* OS has to allow sharing, while at the same time ensure protection.

***Logical Organisation of Memory:***

* Uni-dimensional address space
* 
* If memory was *segmented* then it would be possible to code programs and subroutines separately, each with its own degree of protection.
* The MM would manage inter-segment references at run-time, and could allow a segment to be accessed by many different processes.

***Physical Organisation of Memory:***

* PM is expensive, so tends to be limited - but the amount of PM helps to determine the *degree of multiprogramming* (the number of runnable processes that can be simultaneously maintained)
* A two-level storage scheme (one RAM, the other slower secondary disk) can be used to virtually increase the overall amount of PM.
* Processes can be kept in secondary storage and only brought into PM when needed. MM and OS have to manage operation of moving processes between the two levels.

 **WHAT IS MEMORY ALLOCATION?**

Memory allocation is a process by which computer programs are assigned memory or space.

Here, main memory is divided into two types of partitions

1. Low Memory - Operating system resides in this type of memory.
2. High Memory- User processes are held in high memory.

* **MEMORY MANAGEMENT TECHNIQUES:**

Here, are some most crucial memories management techniques:

* ***Single Contiguous Allocation:***

It is the easiest memory management technique. In this method, all types of computer's memory except a small portion which is reserved for the OS is available for one application. For example, MS-DOS operating system allocates memory in this way. An embedded system also runs on a single application.

* ***Partitioned Allocation:***

It divides primary memory into various memory partitions, which is mostly contiguous areas of memory. Every partition store all the information for a specific task or job. This method consists of allotting a partition to a job when it starts & unallocated when it ends.

* ***Paged Memory Management:***

This method divides the computer's main memory into fixed-size units known as page frames. This hardware memory management unit maps pages into frames which should be allocated on a page basis.

* ***Segmented Memory Management:***

Segmented memory is the only memory management method that does not provide the user's program with a linear and contiguous address space. Segments need hardware support in the form of a segment table. It contains the physical address of the section in memory, size, and other data like access protection bits and status.

# METHADOLOGY

We have discovered ***PAGING TECHNIQUE*** for the project of memory management system.

 **PAGING TECHNIQUE:**

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. It is a memory management scheme that eliminates the need for contiguous allocation of physical memory. This scheme permits the physical address space of a process to be non-contiguous.

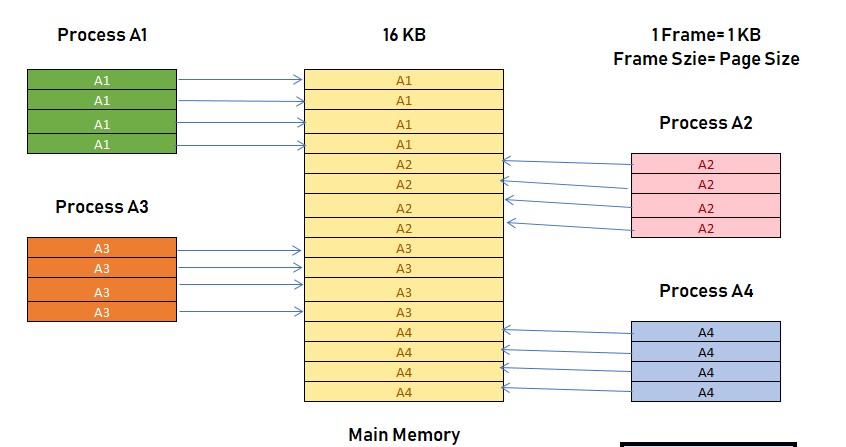
* Logical Address or Virtual Address (represented in bits): An address generated by the CPU
* Logical Address Space or Virtual Address Space (represented in words or bytes): The set of all logical addresses generated by a program
* Physical Address (represented in bits): An address actually available on memory unit
* Physical Address Space (represented in words or bytes): The set of all physical addresses corresponding to the logical addresses

 **Example:**

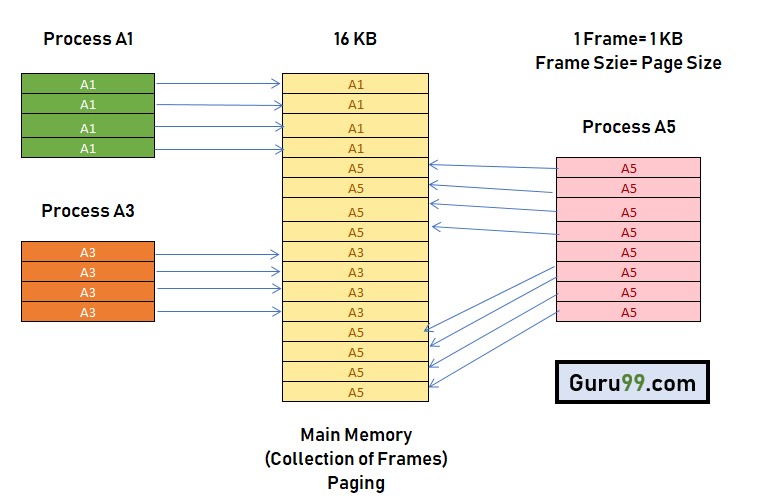
For example, if the main memory size is 16 KB and Frame size is 1 KB. Here, the main memory will be divided into the collection of 16 frames of 1 KB each.

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 block. 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 the different places. This allows us to load the pages of process A5 instead of A2 and A4.

 **ADVANTAGES OF PAGING:**

Here, are advantages of using Paging method:

* Easy to use memory management algorithm
* No need for external Fragmentation
* Swapping is easy between equal-sized pages and page frames.

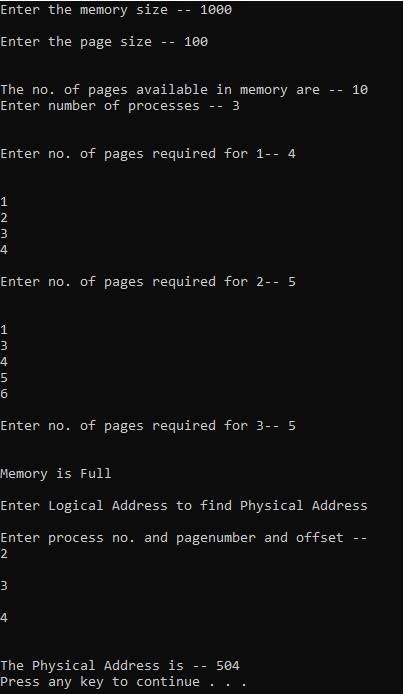
 **DISADVANTAGES OF PAGING**

Here, are disadvantages of Paging:

* May cause Internal fragmentation
* Complex memory management algorithm  Page tables consume additional memory.
* Multi-level paging may lead to memory reference overhead.

***RESULTS AND DISCUSSIONS***

We will have the following output from the above-mentioned code of the implementation of paging technique:



From the above output, we have the following result by paging implementation of MMS:

***Input:***

It has asked the user to enter memory size and the user has entered 1000

* It has asked the user to enter size of pages that is required and the user has entered 100
* Memory has 10 pages available
* It has asked user to enter the number of processes, and he entered 3
* Then it has asked to enter page table in process 1, he entered 4 (1,2,3,4)
* Then it has asked to enter page table in process 2, he entered 5 (1,3,4,5,6)  Then it has asked to enter page table in process 3, he entered 5 ***Output:***
* As there are only 10 pages available in memory and p1 and p2 already consumed 9

pages so it cannot enter 5 more pages in memory so it will give “memory is full”

* It has asked the user to enter logical address to find physical address. Logical address is: enter process no. and page\_number and offset, the user entered 2(process no), 3(page number) and 4(offset)
* The Physical Address is 504

# CONCLUSION

The only memory management approach that does not provide a linear and contiguous address space to the user's program is segmented memory. Swapping is a technique for temporarily moving a process from the main memory to the backing store. It will be brought back into memory later for further execution. Memory allocation is the method of allocating memory or space to computer programs. Paging is a storage mechanism that allows the operating system to retrieve processes from secondary storage and store them as pages in the main memory. Fragmentation describes the state of a disk in which files are fragmented and distributed across the disk. The segmentation approach is very similar to paging. The only difference is that parts can be of any length, while pages in the paging system are all of the same size. A program's dynamic loading routine is one that isn't loaded before the program calls it. Linking is a way for the operating system to collect and combine different code and data modules into a single executable file.

# REFERENCE

* <https://ivypanda.com/essays/computers-memory-management/>
* <https://aircconline.com/cseij/V6N4/6416cseij01.pdf>
* [https://www.researchgate.net/publication/340816041\_Memory\_M anagement\_in\_Operating\_System](https://www.researchgate.net/publication/340816041_Memory_Management_in_Operating_System)

* [https://www.ijtsrd.com/computer-science/operatingsystem/18342/memory-management-in-operating-system/durgeshraghuvanshi](https://www.ijtsrd.com/computer-science/operating-system/18342/memory-management-in-operating-system/durgesh-raghuvanshi)
* [https://www.usenix.org/legacy/event/usenix08/tech/full\_papers/cr aciunas/craciunas.pdf](https://www.usenix.org/legacy/event/usenix08/tech/full_papers/craciunas/craciunas.pdf)
* [https://technobyte.org/memory-management-os-simpleexplanation/](https://technobyte.org/memory-management-os-simple-explanation/)
* [https://www.ukessays.com/essays/information-technology/thememory-management-of-operating-systems-informationtechnology-essay.php](https://www.ukessays.com/essays/information-technology/the-memory-management-of-operating-systems-information-technology-essay.php)
* <https://dl.acm.org/doi/10.1145/3232116.3232136>
* [https://www.researchgate.net/publication/336133290\_MEMORY\_](https://www.researchgate.net/publication/336133290_MEMORY_MANAGEMENT_IN_OPERATING_SYSTEM)

[MANAGEMENT\_IN\_OPERATING\_SYSTEM](https://www.researchgate.net/publication/336133290_MEMORY_MANAGEMENT_IN_OPERATING_SYSTEM)