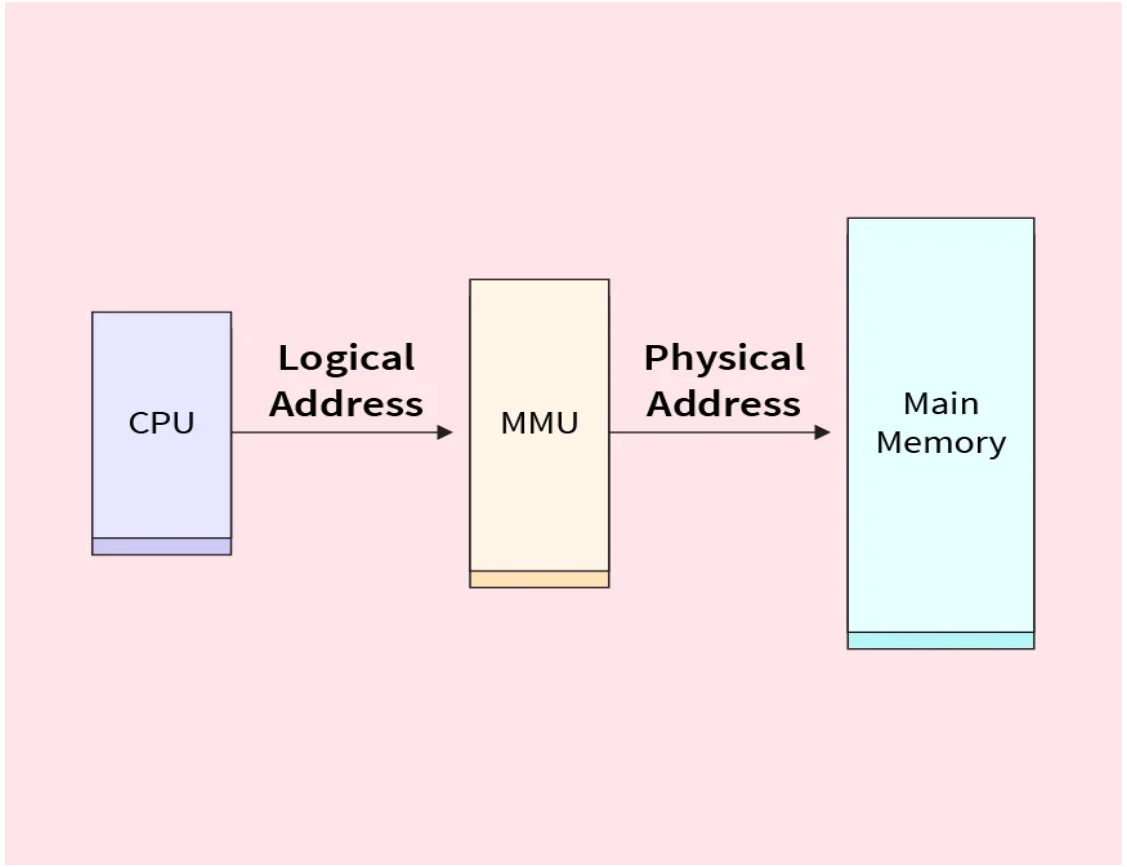
Unit : 3
Memory managment

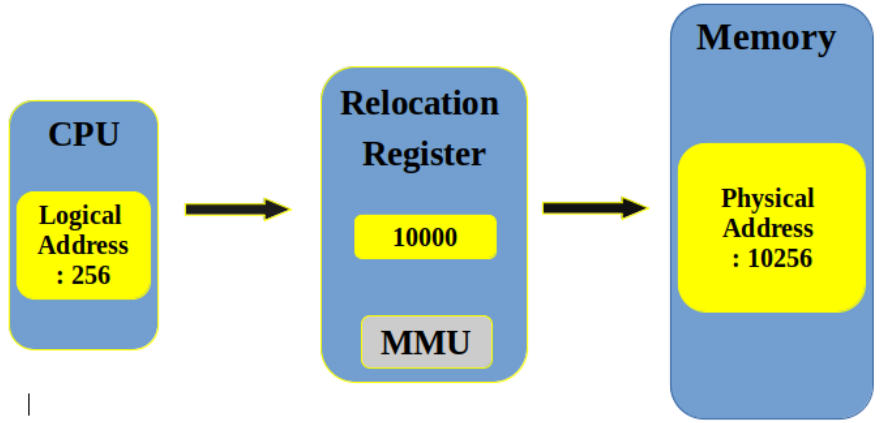
- Bhavika Parmar

Address space

- Address space is a space in computer memory. And process Address Space means a space that is allocated in memory for a process. Every process has an address space.
- Address Space can be of two types

1. **Physical Address Space**
2. **Virtual Address Space**





Logical(Virtual) address space

- **Logical Address** is generated by CPU while a program is running. The logical address is virtual address as it does not exist physically, therefore, it is also known as **Virtual Address**.
- This address is used as a reference to access the physical memory location by CPU. The term Logical Address Space is used for the set of all logical addresses generated by a program’s perspective.
- The hardware device called **Memory-Management Unit(MMU)** is used for mapping logical address to its corresponding physical address.

Physical address space

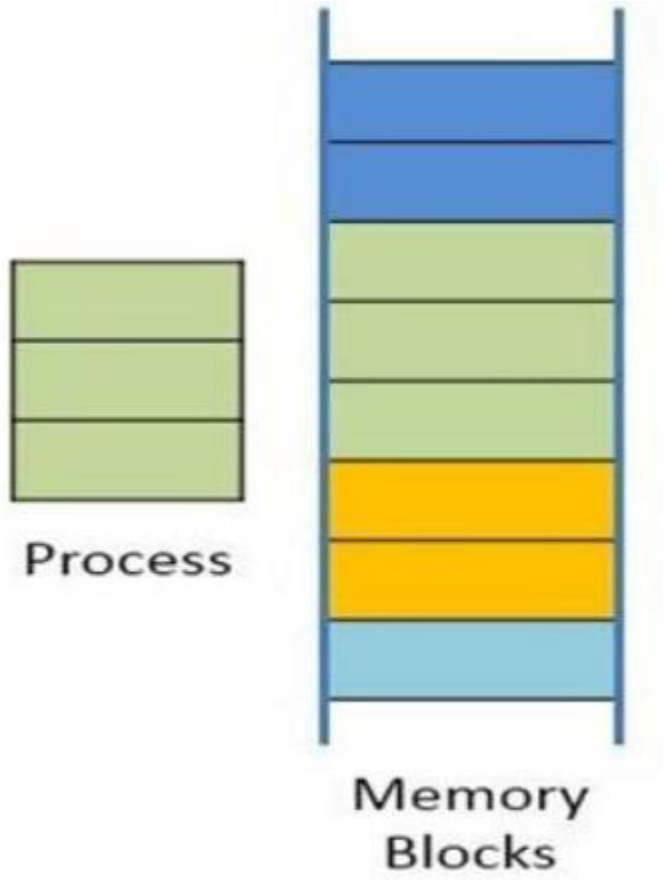
- The physical address is the actual location in the memory that exist physically.
- System access the data in the main memory, with the help of physical address.
- Every thing in the computer has a unique physical address. We needs to mapped it to make the address accessible. MMU is responsible for mapping.

Contiguous and Non-Contiguous Memory Allocation in Operating System

- Memory is a huge collection of bytes, and memory allocation refers to allocating space to computer applications.
- There are mainly two types of memory allocation:
    1) **contiguous memory allocation**
    2) **non-contiguous memory allocation.**
- Contiguous memory allocation allows a single memory space to complete the tasks.
- On the other hand, non-contiguous memory allocation assigns the method to distinct memory sections at numerous memory locations.

What is Contiguous Memory Allocation?

- It is the type of ***memory allocation method.*** When a process requests the memory, a single contiguous section of memory blocks is allotted depending on its requirements.
- It is completed by partitioning the memory into fixed-sized partitions and assigning every partition to a single process. However, it will limit the degree of multiprogramming to the number of fixed partitions done in memory.
- For example, suppose a fixed-sized memory block assigned to a process is slightly bigger than its demand. In that case, the remaining memory space in the block is referred to as internal fragmentation. When a process in a partition finishes, the partition becomes accessible for another process to run.



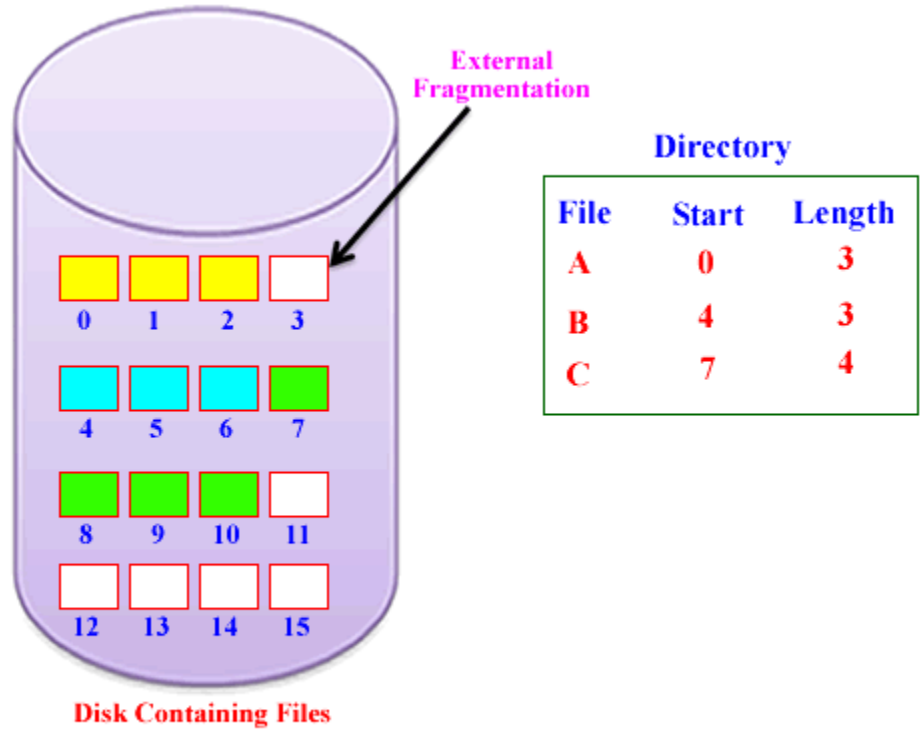


Fig: Showing Contiguous Allocation of Disk

Advantages and Disadvantages of Contiguous Memory Allocation

- **Advantages**

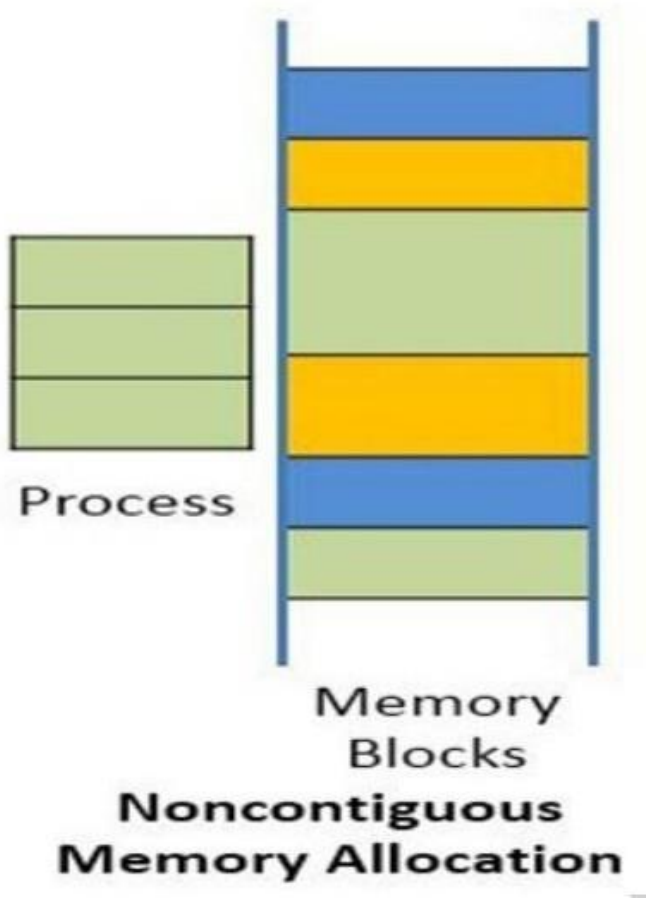
1. It is simple to keep track of how many memory blocks are left, which determines how many more processes can be granted memory space.
2. The read performance of contiguous memory allocation is good because the complete file may be read from the disk in a single task.
3. The contiguous allocation is simple to set up and performs well.

- **Disadvantages**

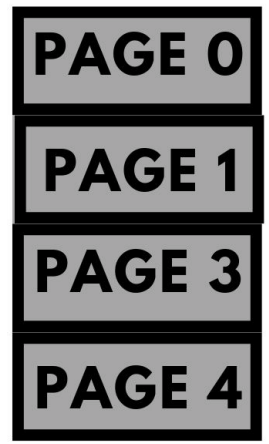
1. Fragmentation isn't a problem because every new file may be written to the end of the disk after the previous one.
2. When generating a new file, it must know its eventual size to select the appropriate hole size(difficult to grow file).

What is Non-Contiguous Memory Allocation?

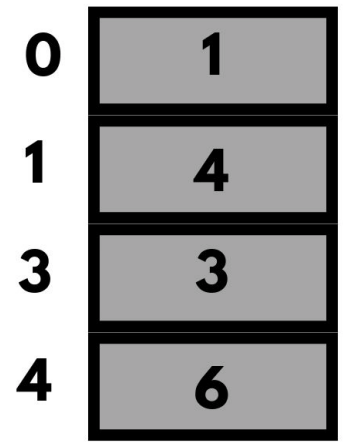
- It allows a process to obtain multiple memory blocks in various locations in memory based on its requirements. The non-contiguous memory allocation also reduces memory wastage caused by ***internal*** and ***external*** fragmentation because it uses the memory holes created by internal and external fragmentation.
- The two methods for making a process's physical address space non-contiguous are paging and segmentation. Non-contiguous memory allocation divides the process into blocks ***(pages or segments)*** that are allocated to different areas of memory space based on memory availability.



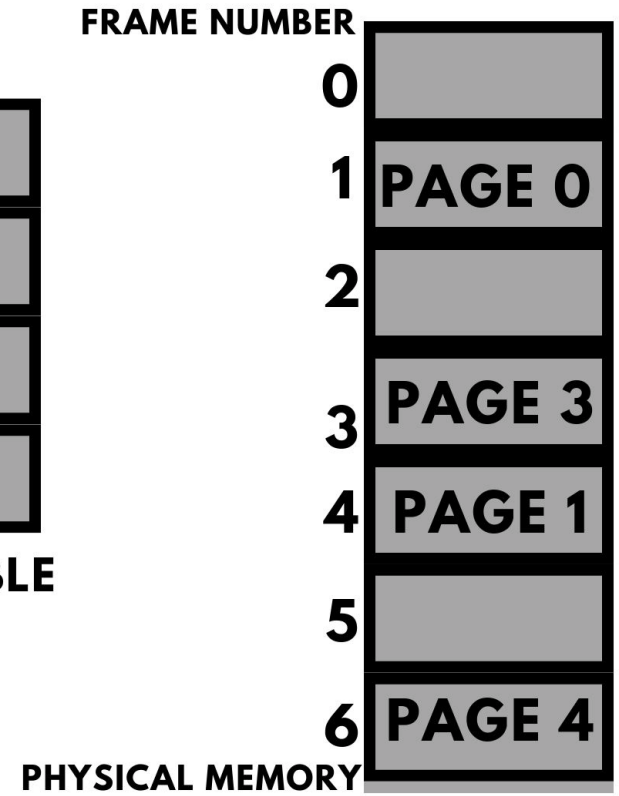
Noncontiguous Memory Allocation



LOGICAL MEMORY



PAGE TABLE



PHYSICAL MEMORY

- **Advantages**

1. It has the advantage of reducing memory waste, but it increases overhead because of the address translation.
2. It slows down the memory execution because time is consumed in address translation.

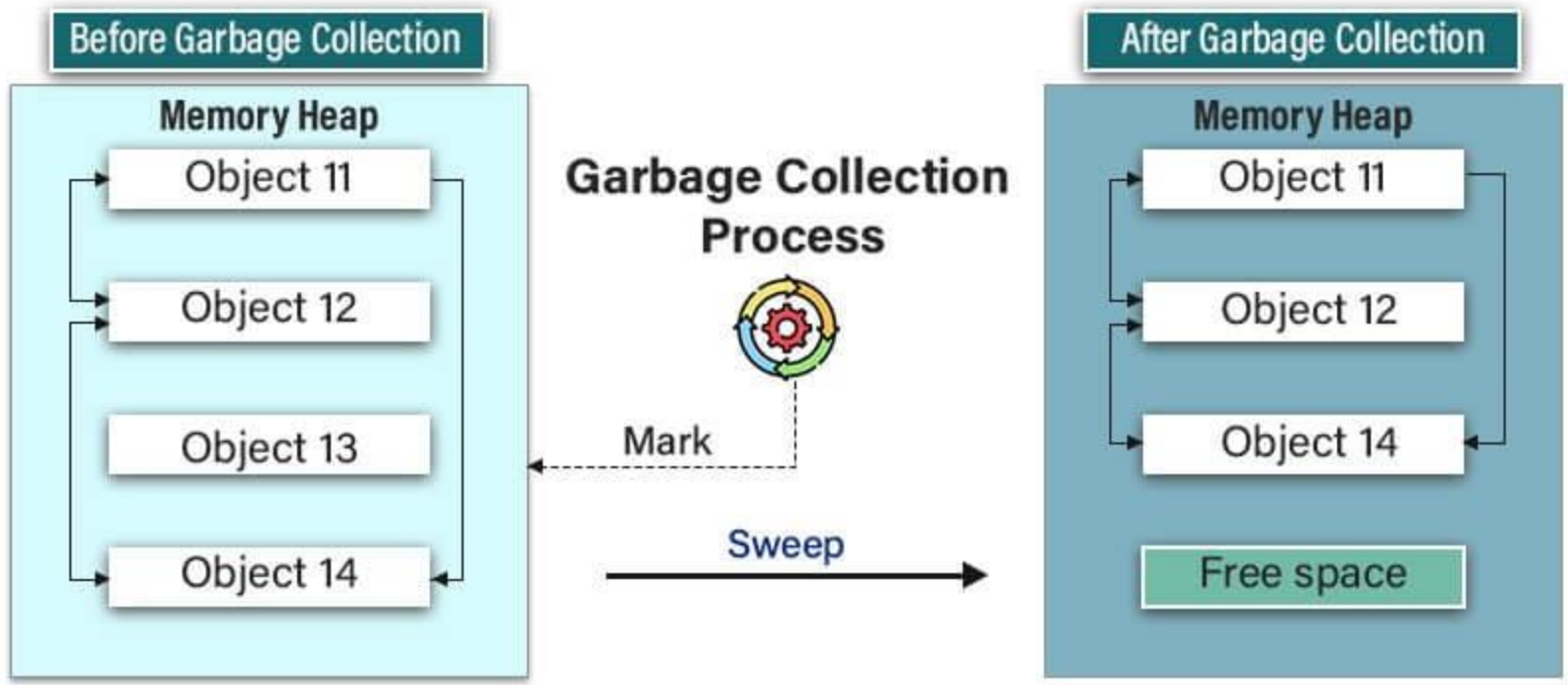
- **Disadvantages**

1. The downside of this memory allocation is that the access is slow because you must reach the other nodes using pointers and traverse them.

Garbage Collection (Managing Free Space)

- Garbage collection's dynamic approach to automatic heap allocation addresses common and costly errors that often result in real world program defects when undetected.
- Because they are difficult to identify and repair, allocation errors are costly. Thus, garbage collection is considered by many to be an essential language feature that makes the programmer’s job easier with lower manual heap allocation management.

Garbage Collection in Data Structure

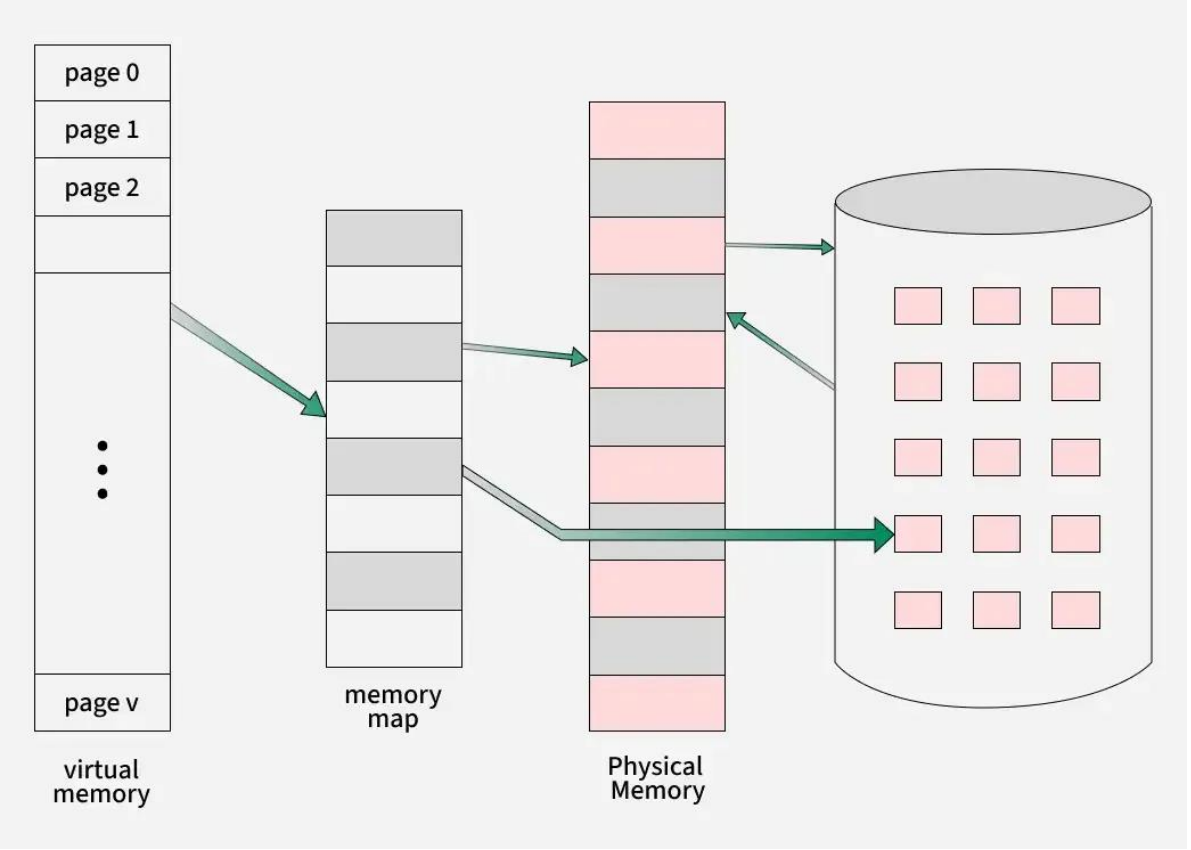


EDUCBA

- However, GC is not perfect, and the following drawbacks should be considered:
- When freeing memory, GC consumes computing resources.
- When unused object references are not manually disposed, GC causes logical memory leaks.
- GC does not always know when to process within virtual memory environments of modern desktop computers.
- The GC process interacts poorly with cache and virtual memory systems, resulting in performance-tuning difficulties.

Virtual Memory

- Virtual memory is a memory management technique used by operating systems to give the appearance of a large, continuous block of memory to applications, even if the physical memory (RAM) is limited. It allows larger applications to run on systems with less RAM.
- Virtual Memory a process are logical addresses that are dynamically translated into physical addresses at run time.



Types of Virtual Memory

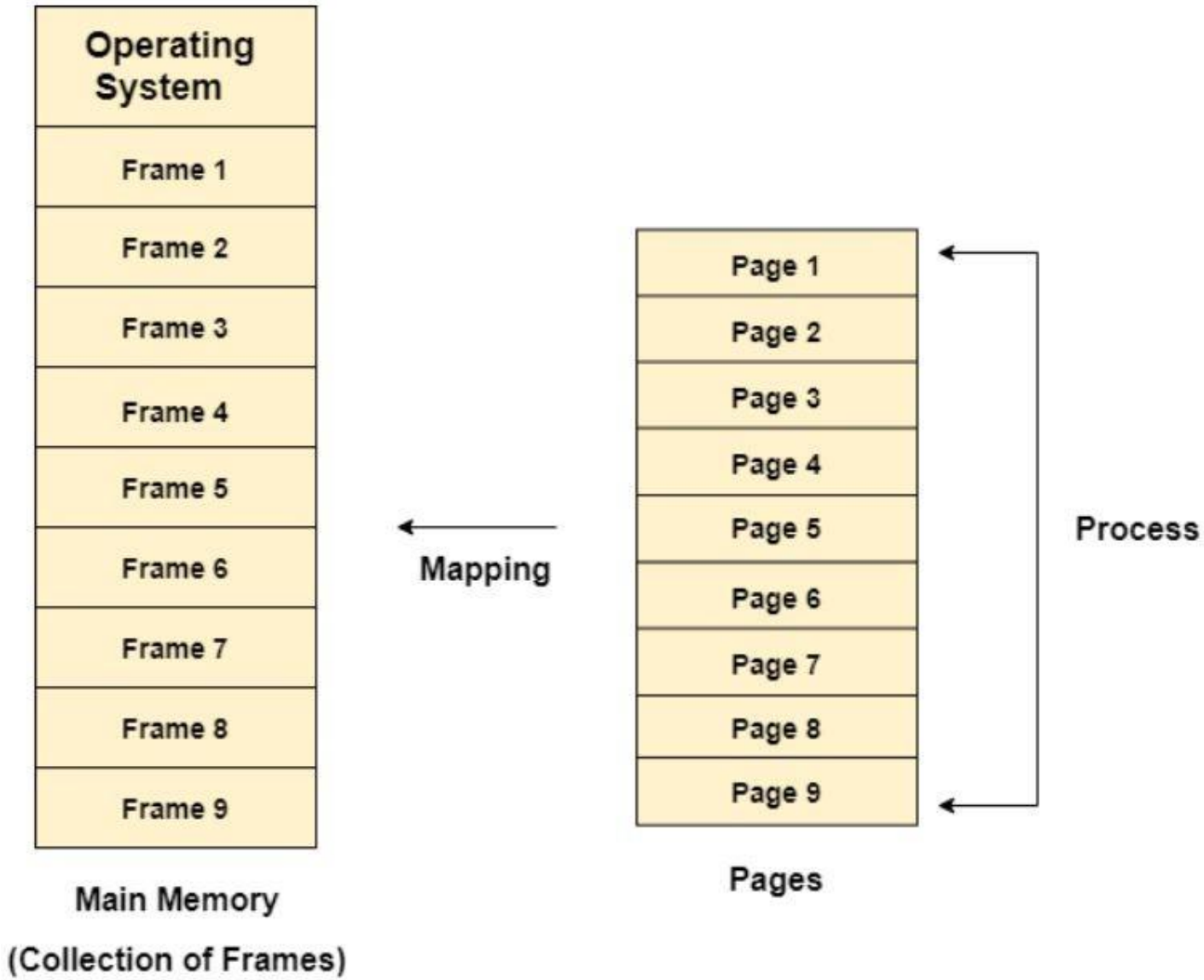
- In a computer, virtual memory is managed by the Memory Management Unit (MMU), which is often built into the CPU. The CPU generates virtual addresses that the MMU translates into physical addresses.
- There are two main types of virtual memory:

1. **Paging**
2. **Segmentation**

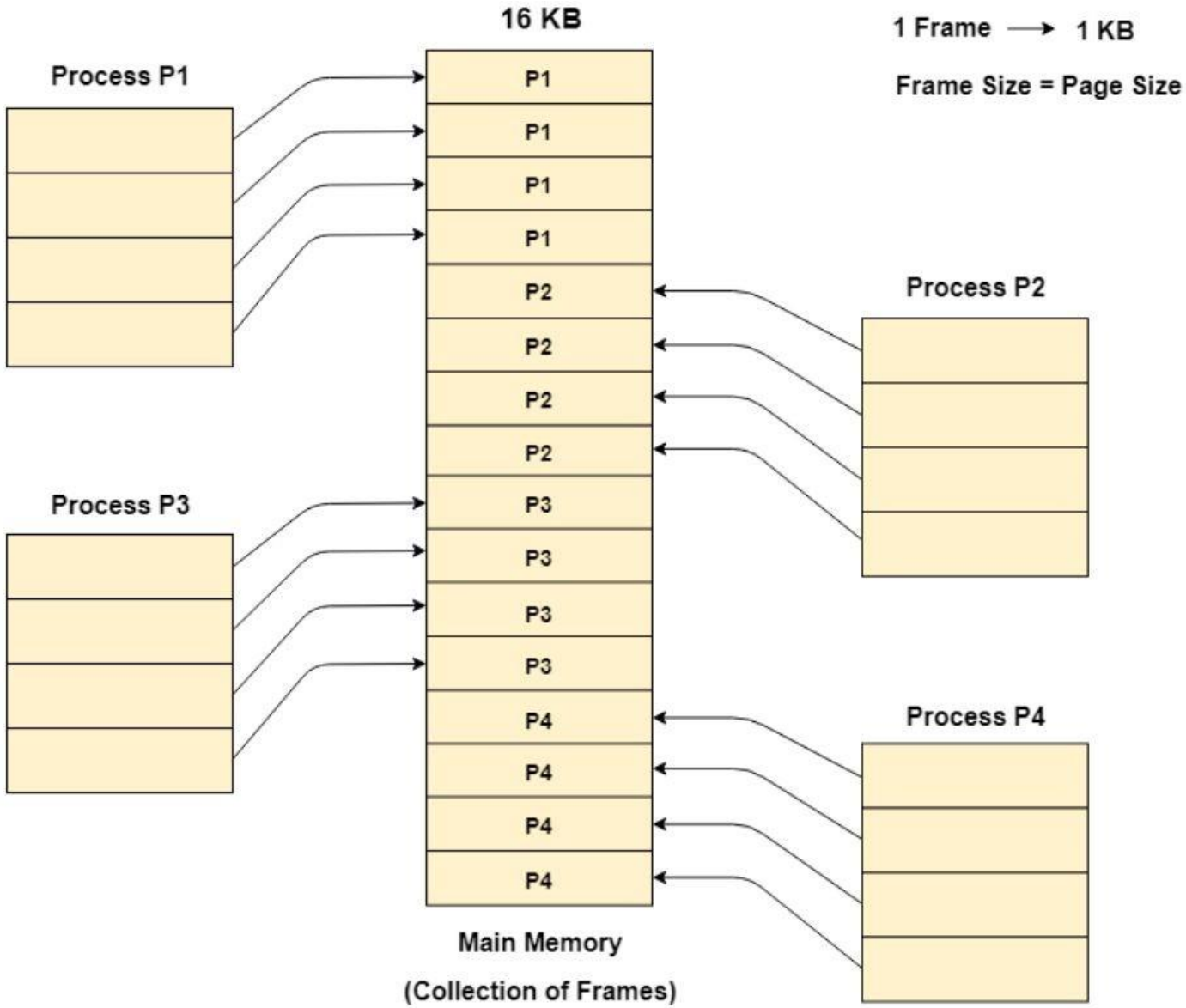
Paging

- Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory.
- **The process of retrieving processes in the form of pages from the secondary storage into the main memory is known as paging**. The basic purpose of paging is to separate each procedure into pages.
- Additionally, frames will be used to split the main memory.This scheme permits the physical address space of a process to be non – contiguous.

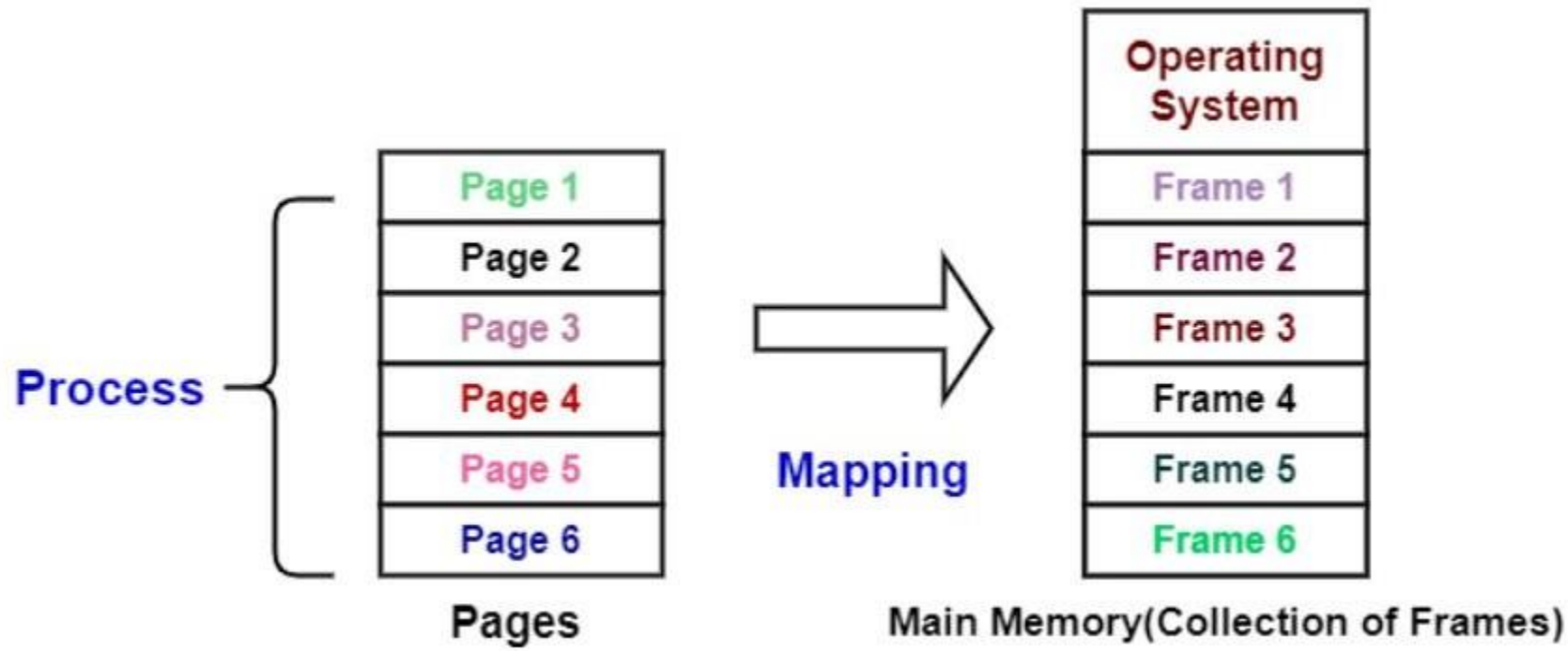
- Let us look some important terminologies:
- **Logical Address or Virtual Address (represented in bits):** An address generated by the CPU
- **Logical Address Space or Virtual Address Space(represented in words or bytes):** The set of all logical addresses generated by a program
- **Physical Address (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


Operating System
Frame 1
Frame 2
Frame 3
Frame 4
Frame 5
Frame 6
Frame 7
Frame 8
Frame 9
Main Memory
(Collection of Frames)
Mapping
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Pages
Process

- Example
- Let us consider the main memory size 16 Kb and Frame size is 1 KB therefore the main memory will be divided into the collection of 16 frames of 1 KB each.
- There are 4 processes in the system that is P1, P2, P3 and P4 of 4 KB each. Each process is divided into pages of 1 KB each so that one page can be stored in one frame.
- Initially, all the frames are empty therefore pages of the processes will get stored in the contiguous way.


16 KB
1 Frame → 1 KB
Frame Size = Page Size
Process P1
Process P2
Process P3
Process P4
P1
P1
P1
P1
P2
P2
P2
P2
P3
P3
P3
P3
P4
P4
P4
P4
Main Memory
(Collection of Frames)

- Each process is mainly divided into parts where the size of each part is the same as the page size.
- There is a possibility that the size of the last part may be less than the page size.
- Pages of a process are brought into the main memory only when there is a requirement otherwise they reside in the secondary storage.
- One page of a process is mainly stored in one of the frames of the memory. Also, the pages can be stored at different locations of the memory but always the main priority is to find contiguous frames.


Process
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Pages
Mapping
Operating System
Frame 1
Frame 2
Frame 3
Frame 4
Frame 5
Frame 6
Main Memory(Collection of Frames)

- The CPU always generates a logical address.
- In order to access the main memory always a physical address is needed.
- The **logical address generated by CPU always consists of two parts:**

1. Page Number(p)
2. Page Offset (d)

- where,
- **Page Number** is used to specify the specific page of the process from which the CPU wants to read the data. and it is also used as an index to the page table.
- and **Page offset** is mainly used to specify the specific word on the page that the CPU wants to read.

Page Table in OS

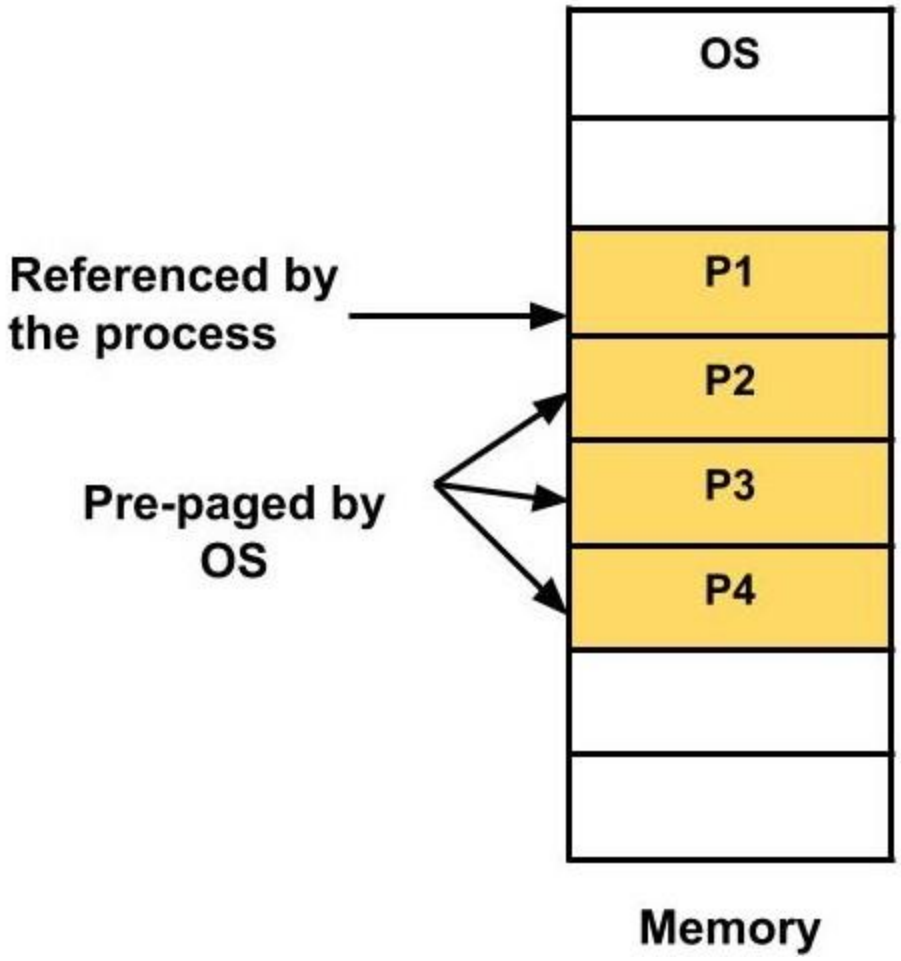
- The Page table mainly **contains the base address of each page** in the Physical memory. The base address is then combined with the page offset in order to define the **physical memory address** which is then sent to the memory unit.
- Thus page table mainly provides the corresponding frame number (base address of the frame) where that page is stored in the main memory.
- As we have told you above that the **frame number** is combined with the **page offset** and forms the **required physical address.**

What is Demand Paging?

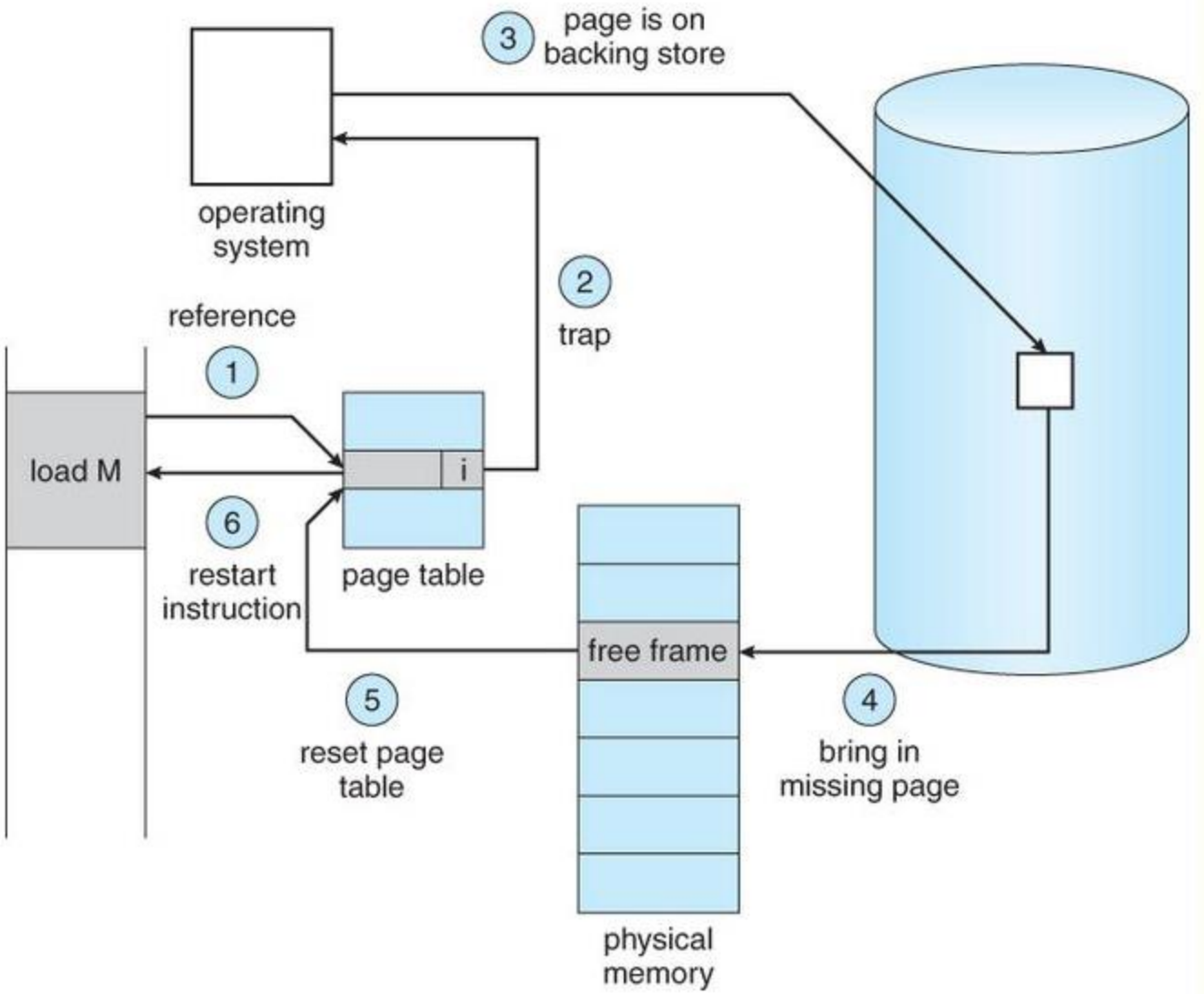
Demand paging is a technique used in virtual memory systems where pages enter main memory only when requested or needed by the CPU. In demand paging, the operating system loads only the necessary pages of a program into memory at runtime, instead of loading the entire program into memory at the start.

What is Page Fault ?

A page fault is an exception that occurs when a program tries to access a memory page that is not in physical memory.



Demand Paging



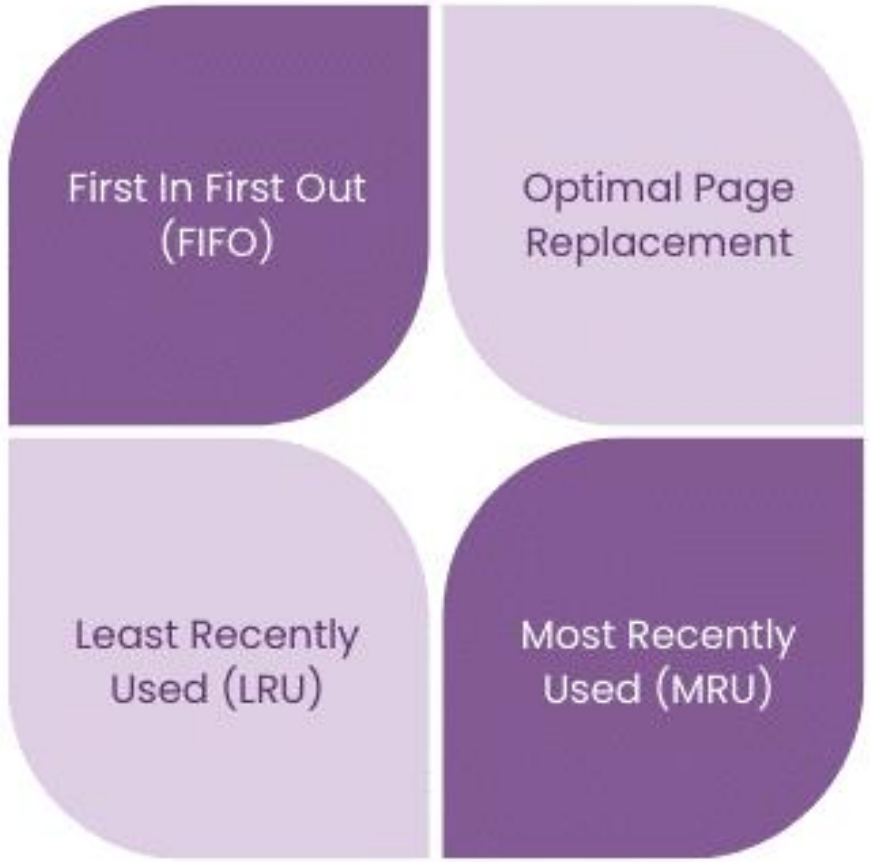
Page Fault

Page Replacement Algorithms:

Page replacement algorithms are techniques used in operating systems to manage memory efficiently when the virtual memory is full.

When a new page needs to be loaded into physical memory, and there is no free space, these algorithms determine which existing page to replace.

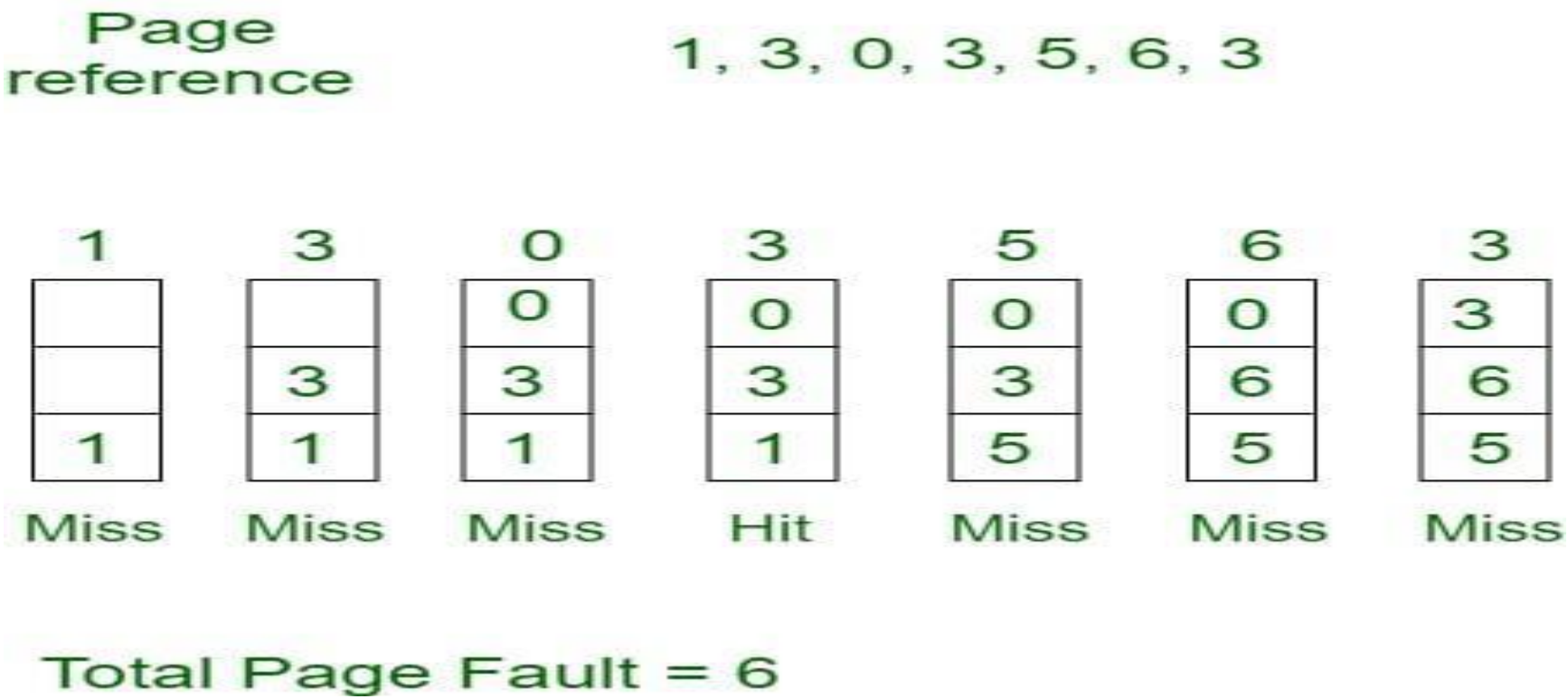
Common Page Replacement Techniques



First In First Out (FIFO)

- FIFO algorithm is the simplest of all the page replacement algorithms. In this, we maintain a queue of all the pages that are in the memory currently. The oldest page in the memory is at the front-end of the queue and the most recent page is at the back or rear-end of the queue.
- Whenever a page fault occurs, the operating system looks at the front-end of the queue to know the page to be replaced by the newly requested page. It also adds this newly requested page at the rear-end and removes the oldest page from the front-end of the queue.

- **Example:** Consider page reference string 1, 3, 0, 3, 5, 6, 3 with 3-page frames. Find the number of page faults using FIFO Page Replacement Algorithm.



- **Belady's anomaly:** Generally if we increase the number of frames in the memory, the number of page faults should decrease due to obvious reasons. Belady’s anomaly refers to the phenomena where increasing the number of frames in memory, increases the page faults as well.
- **Advantages**
- Simple to understand and implement
- Does not cause more overhead

Optimal Page Replacement in OS

- Optimal Page Replacement is one of the Algorithms of Page Replacement. In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.
- Two conditions :

1. If referred page is already present, increment hit count.
2. If not present, find if a page that is never referenced in future. If such a page exists, replace this page with new page. If no such page exists, find a page that is referenced farthest in future. Replace this page with new page.

Example : Consider the page references 6, 7, 8, 9, 6, 7, 1, 6, 7, 8, 9, 1, 7, 9, 6 with 3-page frame. Find number of page fault using Optimal Page Replacement Algorithm.

Pages >>	6	7	8	9	6	7	1	6	7	8	9	1	7	9	6
Frame 3			8	9	9	9	1	1	1	1	1	1	1	1	1
Frame 2		7	7	7	7	7	7	7	7	7	7	7	7	7	7
Frame 1	6	6	6	6	6	6	6	6	6	8	9	9	9	9	6
	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Hit	Hit	Miss	Miss	Hit	Hit	Hit	Miss

- **6, 7, 8, 9:** First four pages fill the frames → Misses = 4
- **6:** Already in the frame → Hit
- **7:** Already in the frame → Hit
- **1:** replaces 9 → Miss
- **6, 7:** Already in the frame → Hit
- **8, 9:** 8 replaces 6 and 9 replaces 8 → Misses = 2
- **1, 7, 9:** already in frames → Hits = 3
- **6:** 6 replaces 9 → Miss

Hit page = 7

Page Fault = 8

- **Advantages**
- Excellent efficiency
- Less complexity
- Easy to use and understand
- Simple data structures can be used to implement

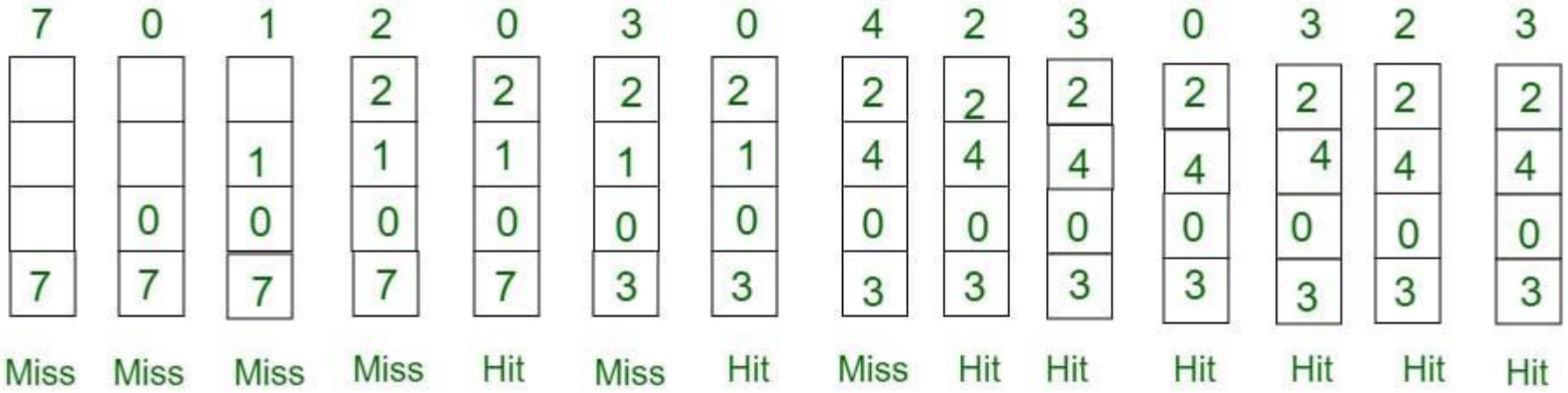
- **Disadvantages**
- More time consuming
- Difficult for error handling
- Need future awareness of the programs, which is not possible every time

Least Recently Used (LRU) Page Replacement Algorithm

- In this algorithm, page will be replaced which is least recently used.
- In this algorithm, **when a page fault occurs, then the page that has not been used for the longest duration of time is replaced by the newly requested page**.
- Now in the above example, the LRU causes the same page faults as the FIFO, but this may not always be the case as it will depend upon the series, the number of frames available in memory, etc. In fact, on most occasions, LRU is better than FIFO.

- **Example :** Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3 with 4-page frames. Find number of page faults using LRU Page Replacement Algorithm.

Page reference 7,0,1,2,0,3,0,4,2,3,0,3,2,3 No. of Page frame - 4



Total Page Fault = 6

Here LRU has same number of page fault as optimal but it may differ according to question.

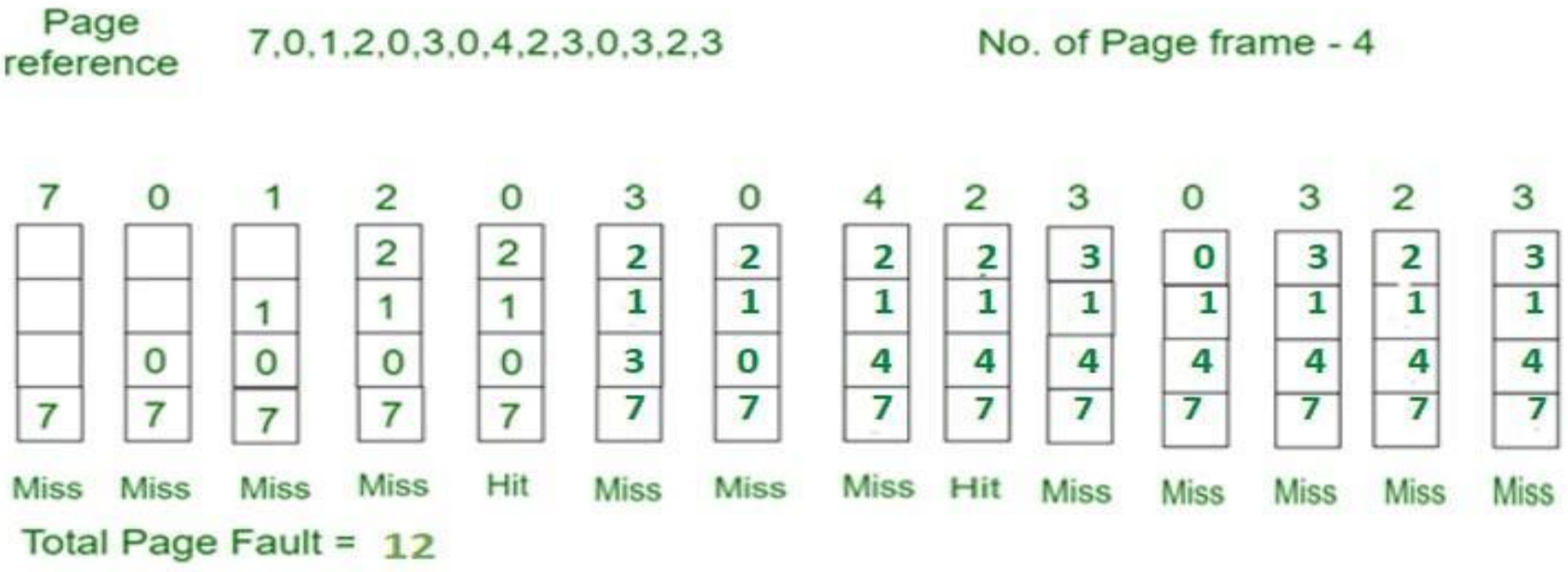
Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**
0 is already there so —> **0 Page fault.** when 3 came it will take the place of 7 because it is least recently used —> **1 Page fault**
0 is already in memory so —> **0 Page fault**.
4 will takes place of 1 —> **1 Page Fault**
Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

- **Advantages**
- It is open for full analysis
- Doesn’t suffer from Belady’s anomaly
- Often more efficient than other algorithms
- **Disadvantages**
- It requires additional data structures to be implemented
- More complex
- High hardware assistance is required

Most Recently Used (MRU)Page Replacement Algorithm

- In this algorithm, page will be replaced which has been used recently. Belady's anomaly can occur in this algorithm.

Example : Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3 with 4-page frames. Find number of page faults using MRU Page Replacement Algorithm.



Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**

0 is already their so–> **0 page fault**

when 3 comes it will take place of 0 because it is most recently used —> **1 Page fault**

when 0 comes it will take place of 3 —> **1 Page fault**

when 4 comes it will take place of 0 —> **1 Page fault**

2 is already in memory so —> **0 Page fault**

when 3 comes it will take place of 2 —> **1 Page fault**

when 0 comes it will take place of 3 —> **1 Page fault**

when 3 comes it will take place of 0 —> **1 Page fault**

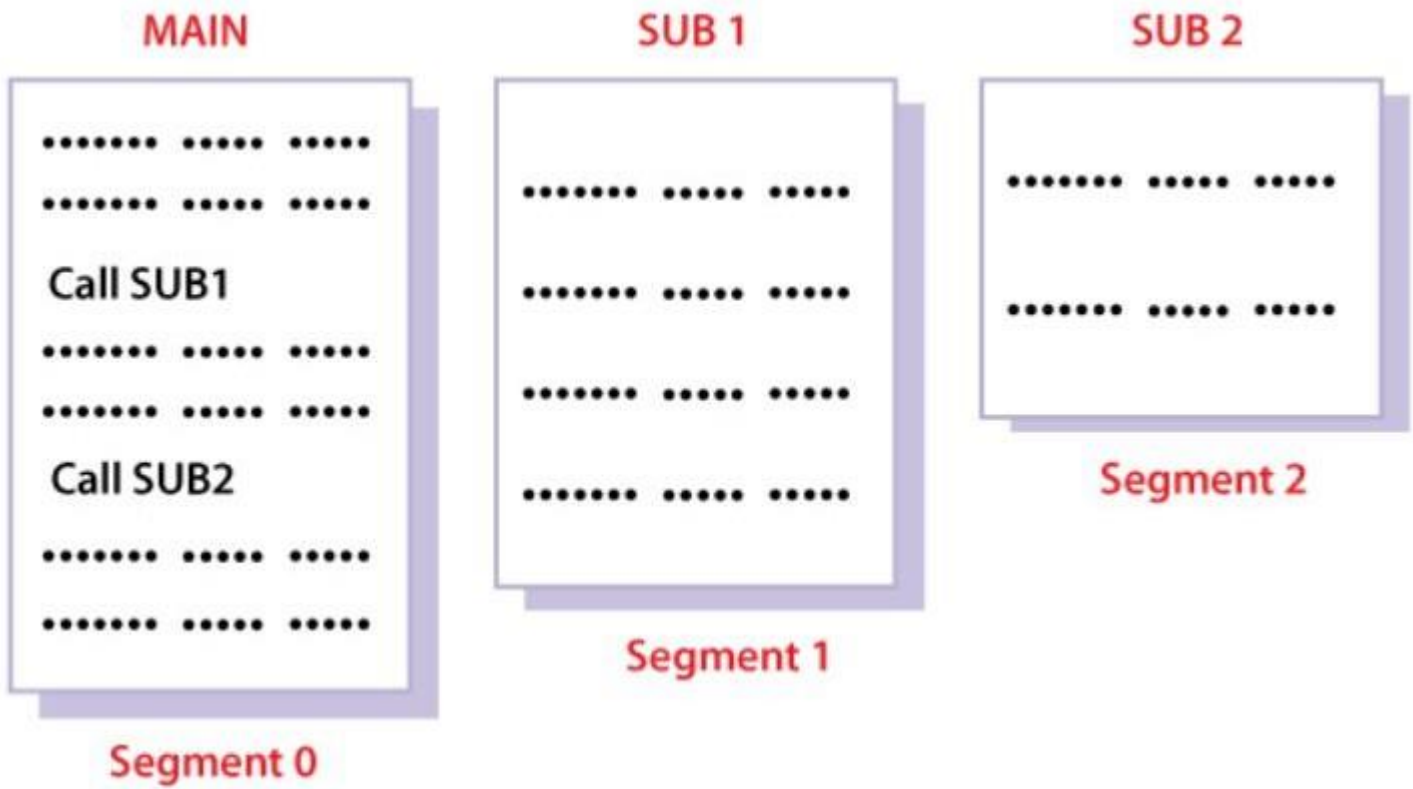
when 2 comes it will take place of 3 —> **1 Page fault**

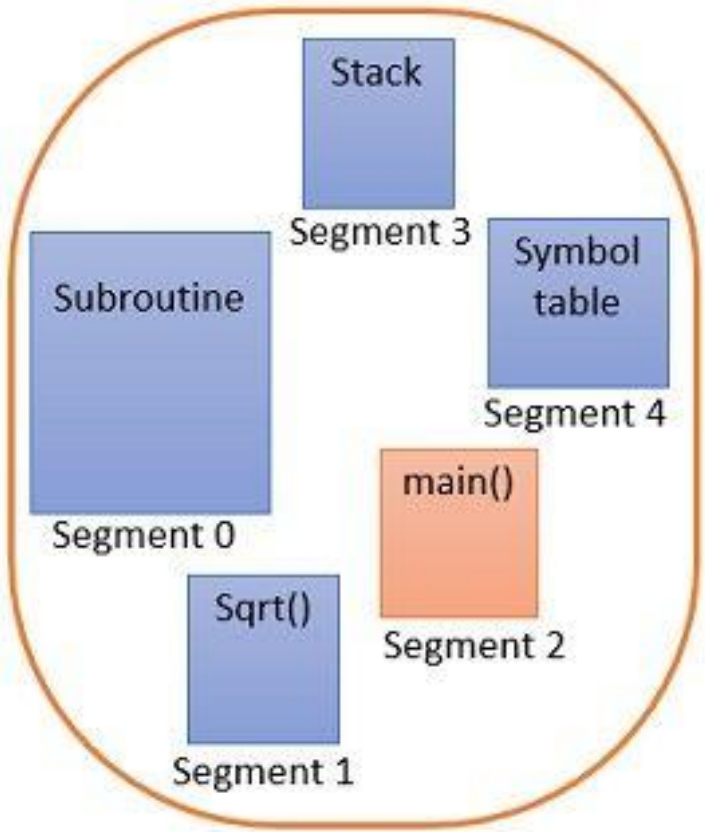
when 3 comes it will take place of 2 —> **1 Page fault**

What is Segmentation?

- In Operating Systems, Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a segment which can be allocated to a process.
- The details about each segment are stored in a table called a segment table. Segment table is stored in one (or many) of the segments.
- Segment table contains mainly two information about segment:

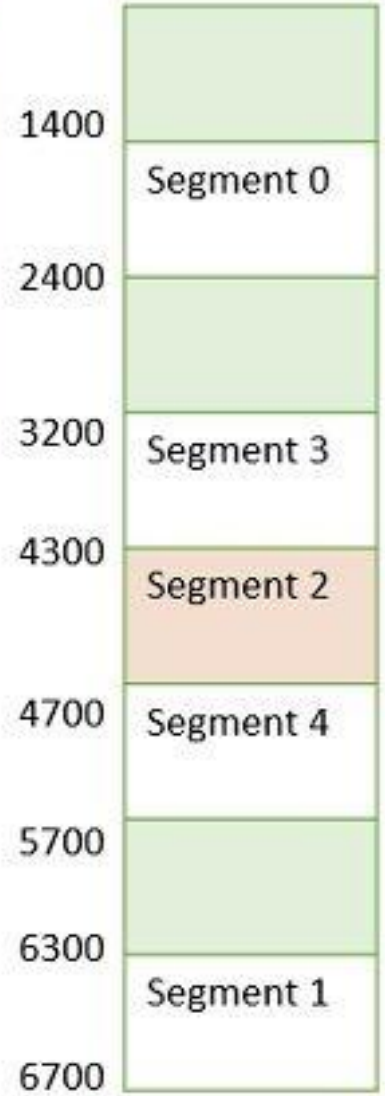
1. Base: It is the base address of the segment
2. Limit: It is the length of the segment.


MAIN
Call SUB1
Call SUB2
Segment 0
SUB 1
Segment 1
SUB 2
Segment 2



	limit	Base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

Segment Table



Physical Memory

Example of Segmentation

Difference between Paging and Segmentation

Paging	Segmentation
Fix sized pages	Variable sized segments
Internal fragmentation	No internal fragmentation
Hardware decides the page size	The user decides the segment size
Faster memory access	Memory access is slower as compared to paging
Page table stores data	Segmentation table stores data
Cannot distinguish and secure procedures and data	Can separate and secure procedures and data
1-D address space	Multiple independent address spaces

THANK YOU