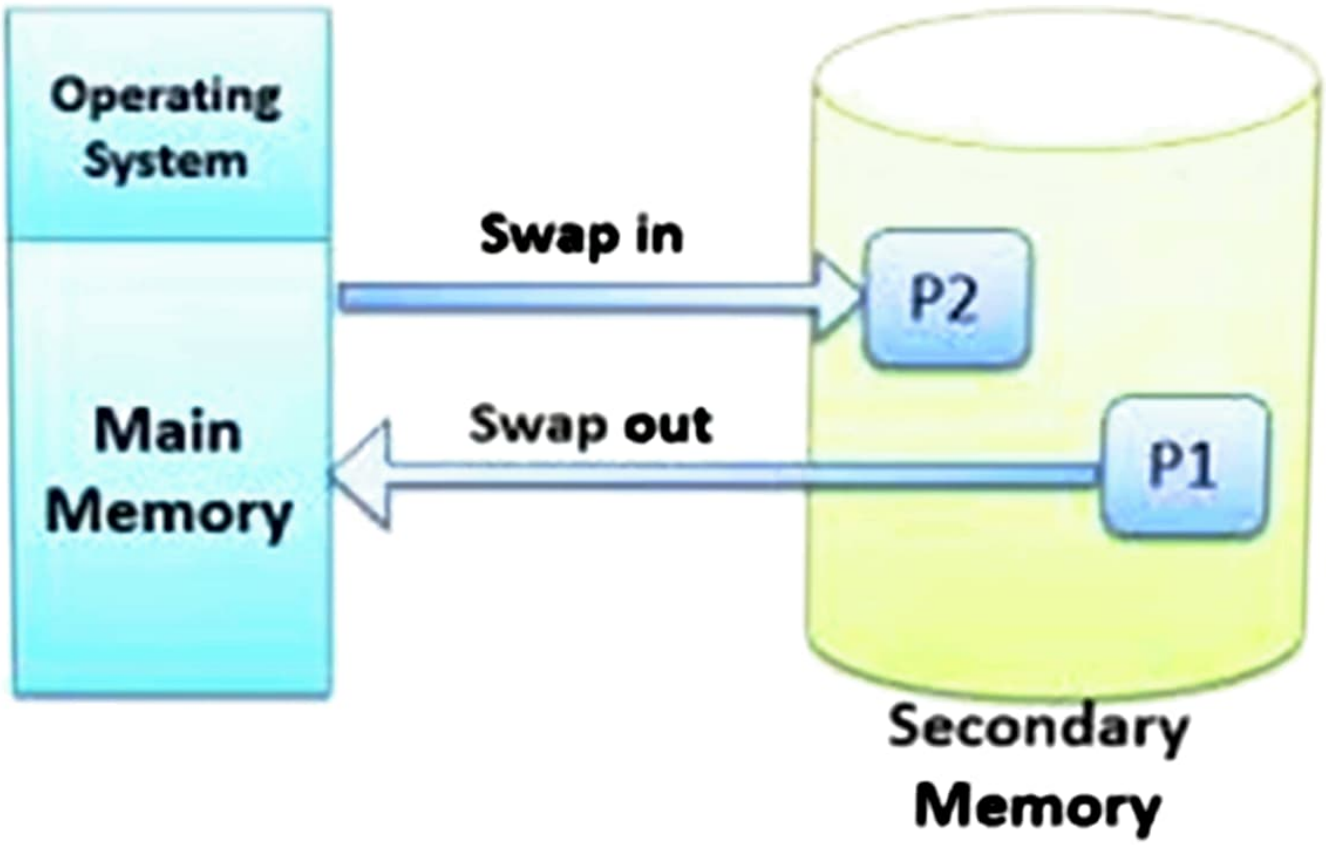
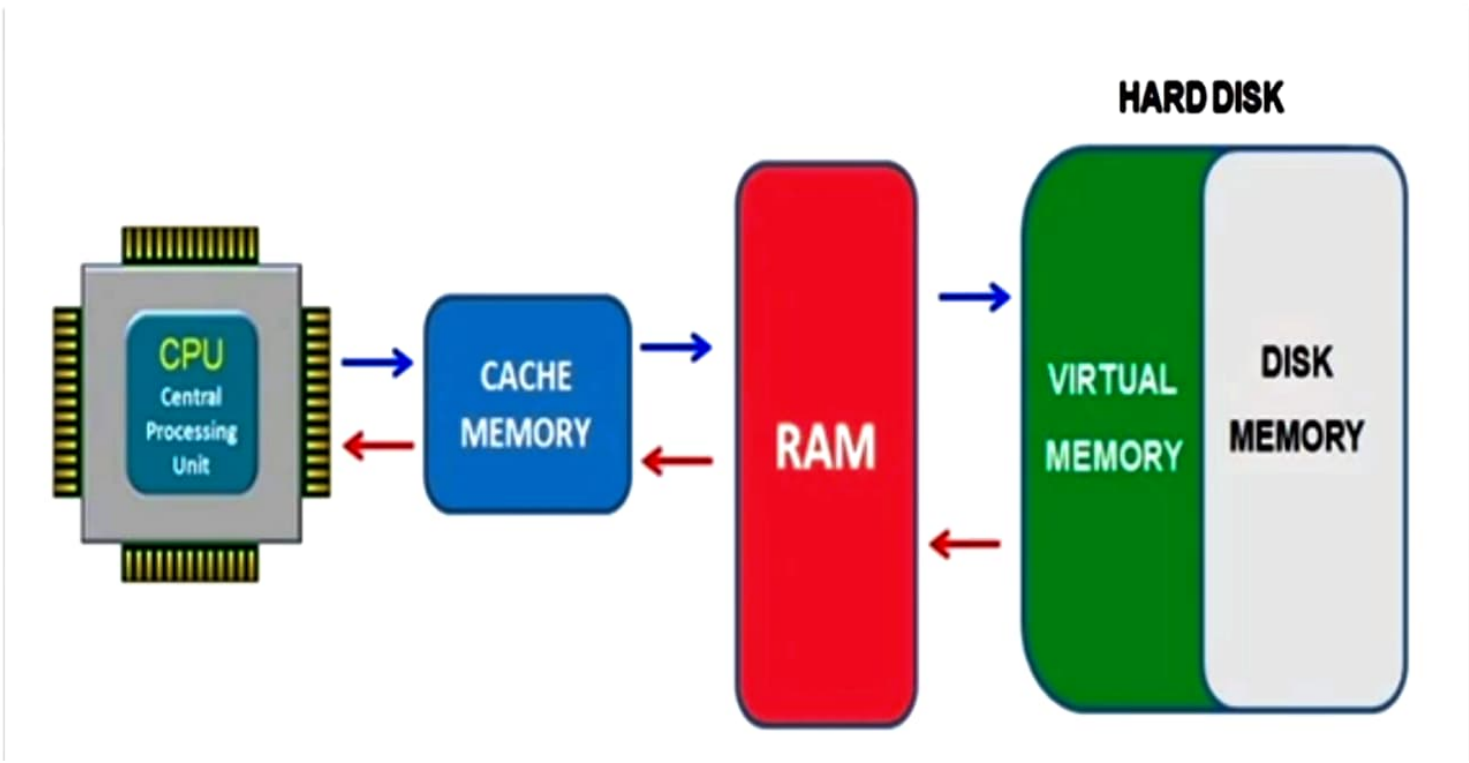
# Memory Management

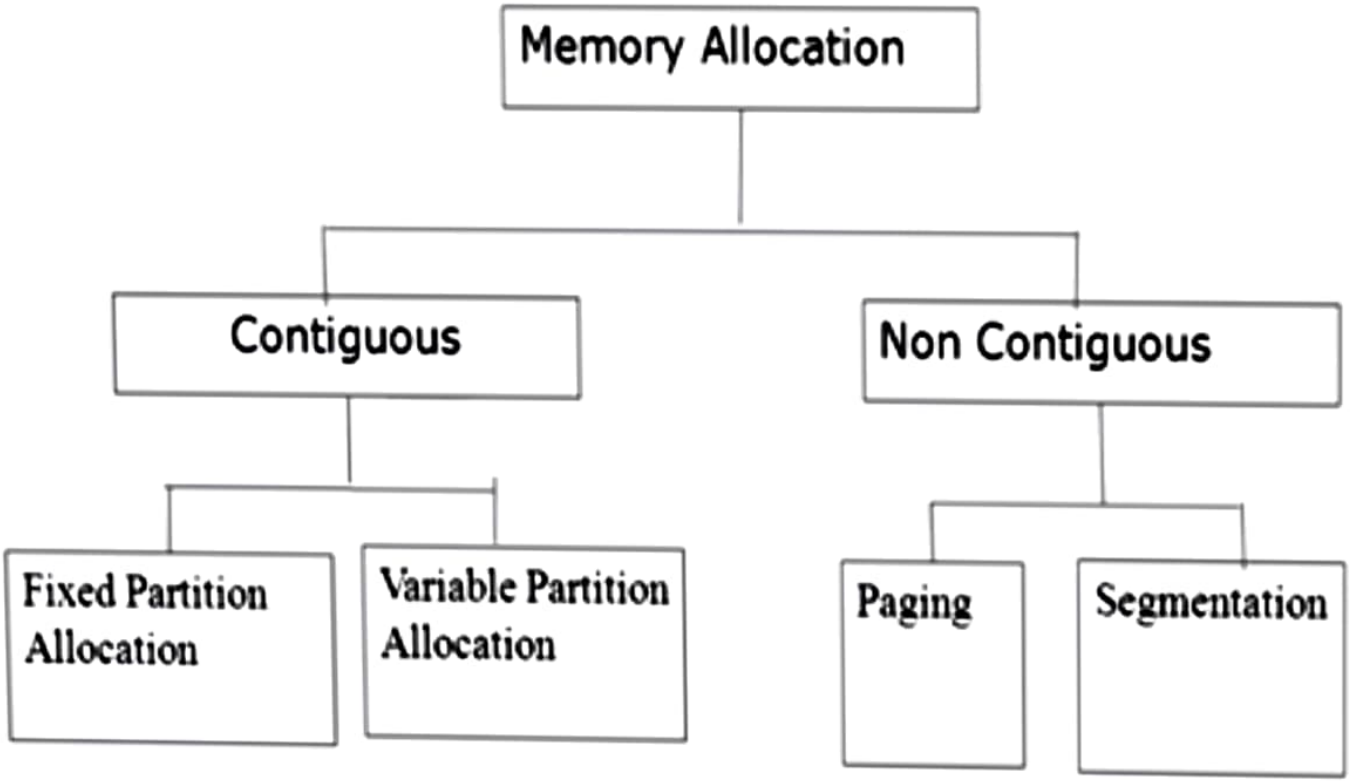
- **Memory Management** is the process of controlling and coordinating computer memory, assigning portions known as blocks to various running programs to optimize the overall performance of the system.
- It is the most important function of an operating system that manages primary memory.
- It helps OS to keep track of every memory location, irrespective of whether it is allocated to some process or it remains free.


Operating System
Main Memory
Swap in
Swap out
P2
P1
Secondary Memory

Swapping

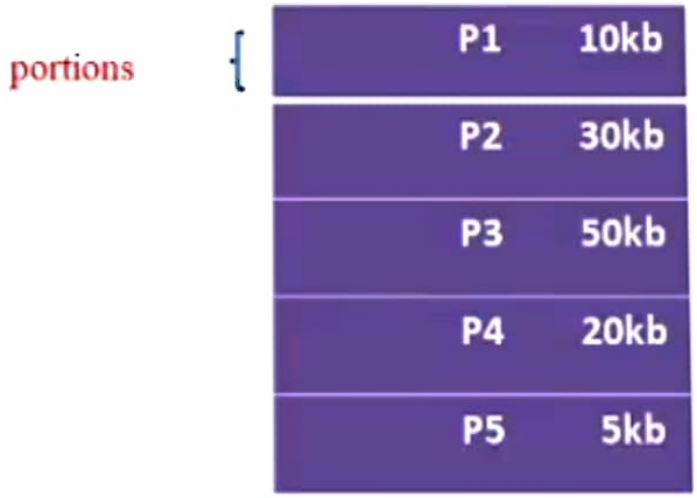

HARD DISK
CPU
Central
Processing
Unit
CACHE
MEMORY
RAM
VIRTUAL
MEMORY
DISK
MEMORY

# Memory Management Techniques



# Fixed Partition 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.



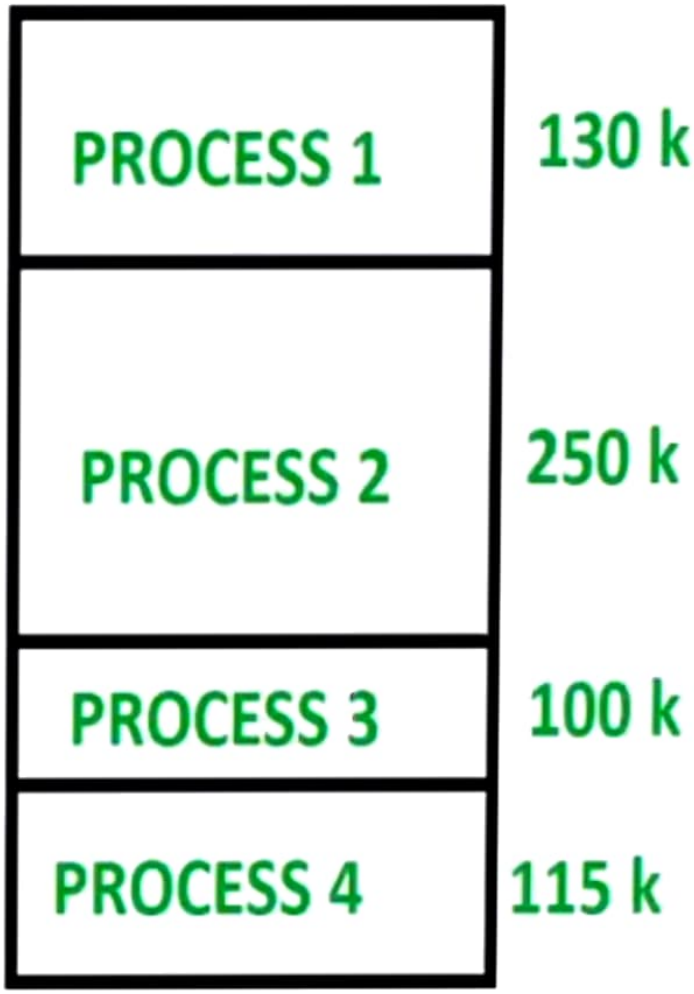
- Multi-programming with fixed partitioning is a contiguous memory management technique in which the main memory is divided into fixed sized partitions which can be of equal or unequal size.

- Whenever we have to allocate a process memory then a free partition that is big enough to hold the process is found. Then the memory is allocated to the process.

- If there is no free space available then the process waits in the queue to be allocated memory. It is one of the most oldest memory management technique which is easy to implement.

# Varaible Partition Allocation

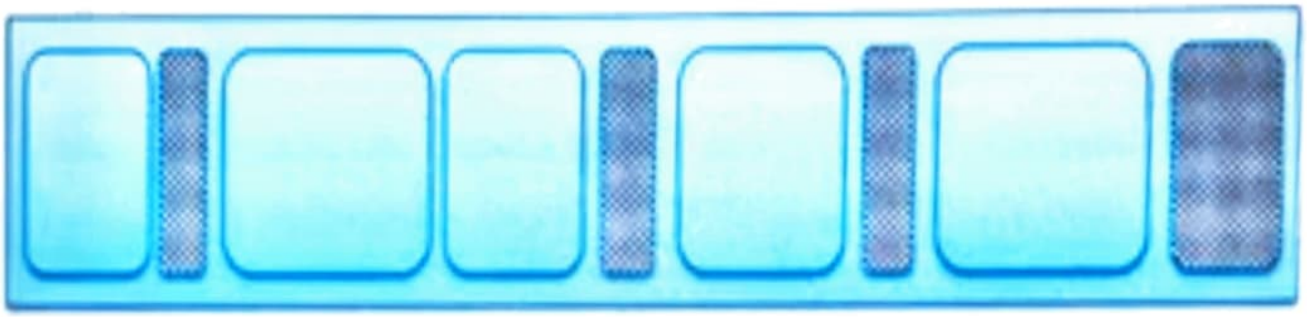
- In **variable Partitioning**, space in main memory is allocated strictly according to the need of process.
- Multi-programming with variable partitioning is a contiguous memory management technique in which the main memory is not divided into partitions and the process is allocated a chunk of free memory that is big enough for it to fit.
- The space which is left is considered as the free space which can be further used by other processes.


PROCESS 1
130 k
PROCESS 2
250 k
PROCESS 3
100 k
PROCESS 4
115 k

# Compaction

- **Compaction** is a process in which the free space is collected in a large memory chunk to make some space available for processes.
- In **memory management**, swapping creates multiple fragments in the memory because of the processes moving in and out. **Compaction** refers to combining all the empty spaces together and processes.
- A possible remedy to the problem of external fragmentation is compaction.

Fragmented memory before compaction



Memory after compaction

# Problem with Compaction

- The efficiency of the system is decreased in the case of compaction due to the fact that all the free spaces will be transferred from several places to a single place.

- Huge amount of time is invested for this procedure and the CPU will remain idle for all this time. Despite of the fact that the compaction avoids external fragmentation, it makes system inefficient.

# Memory Allocation Methods

- For both fixed and dynamic memory allocation schemes, the operating system must keep list of each memory location noting which are free and which are busy.

- Then as new jobs come into the system, the free partitions must be allocated.

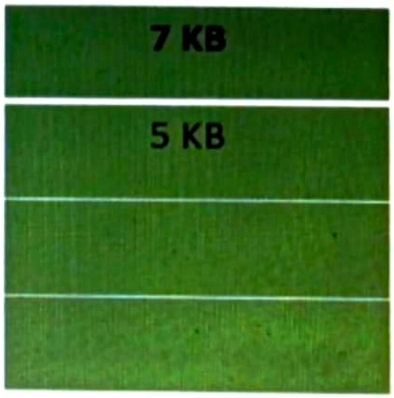
- These partitions may be allocated by 4 ways:

1. First-Fit Memory Allocation
2. Best-Fit Memory Allocation
3. Worst-Fit Memory Allocation
4. Next-Fit Memory Allocation

- These are **Contiguous** memory allocation techniques.

# First-Fit Memory Allocation

- In this method, first job claims the first available memory with space more than or equal to it's size.




- The operating system doesn't search for appropriate partition but just allocate the job to the nearest memory partition available with sufficient size.

- **Advantages of First-Fit Memory Allocation:**

  It is fast in processing. As the processor allocates the nearest available memory partition to the job, it is very fast in execution.

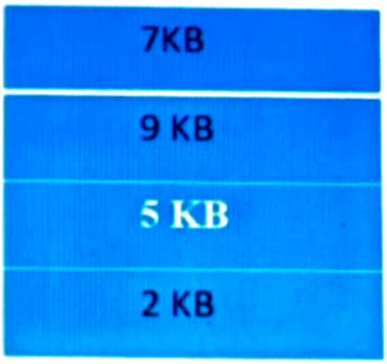
- **Disadvantages of First-Fit Memory Allocation :**

  It wastes a lot of memory. The processor ignores if the size of partition allocated to the job is very large as compared to the size of job or not.

  It just allocates the memory. As a result, a lot of memory is wasted and many jobs may not get space in the memory, and would have to wait for another job to complete.

# Best-Fit Allocation

- In this method, the operating system first searches the whole of the memory according to the size of the given job and allocates it to the closest-fitting free partition in the memory, making it able to use memory efficiently.




- Here the jobs are in the order from smallest job to largest job.

- **Advantages of Best-Fit Allocation :**

- Memory Efficient. The operating system allocates the job minimum possible space in the memory, making memory management very efficient.
- To save memory from getting wasted, it is the best method.

- **Disadvantages of Best-Fit Allocation :**

- It is a Slow Process. Checking the whole memory for each job makes the working of the operating system very slow.
- It takes a lot of time to complete the work.

# Worst-Fit Allocation


- In this allocation technique, the process traverses the whole memory and always search for the largest hole/partition and then the process is placed in that hole/partition.

- It is a slow process because it has to traverse the entire memory to search the largest hole.

- **Advantages of Worst-Fit Allocation :**

- Since this process chooses the largest hole/partition, therefore there will be large internal fragmentation.
- Now, this internal fragmentation will be quite big so that other small processes can also be placed in that leftover partition.

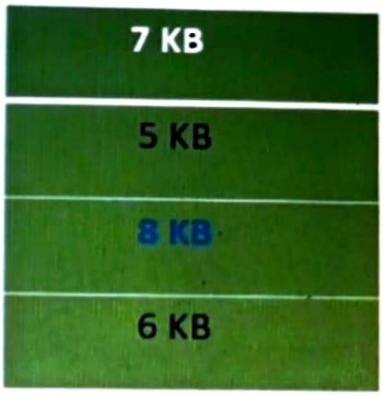
- **Disadvantages of Worst-Fit Allocation :**

- It is a slow process because it traverses all the partitions in the memory and then selects the largest partition among all the partitions, which is a time-consuming process.

# Next Fit Allocation

- Next fit is a modified version of 'first fit'.

It begins as the first fit to find a free partition but when called next time it starts searching from where it left off, not from the beginning.




- This policy makes use of a roving pointer. The pointer moves along the memory chain to search for a next fit.

- This helps in. to avoid the usage of memory always from the head (beginning) of the free block chain.

- **Advantage:**

- Next fit is a very fast searching algorithm and is also comparatively faster than First Fit and Best Fit Memory Management Algorithms.

- **Disadvantage**

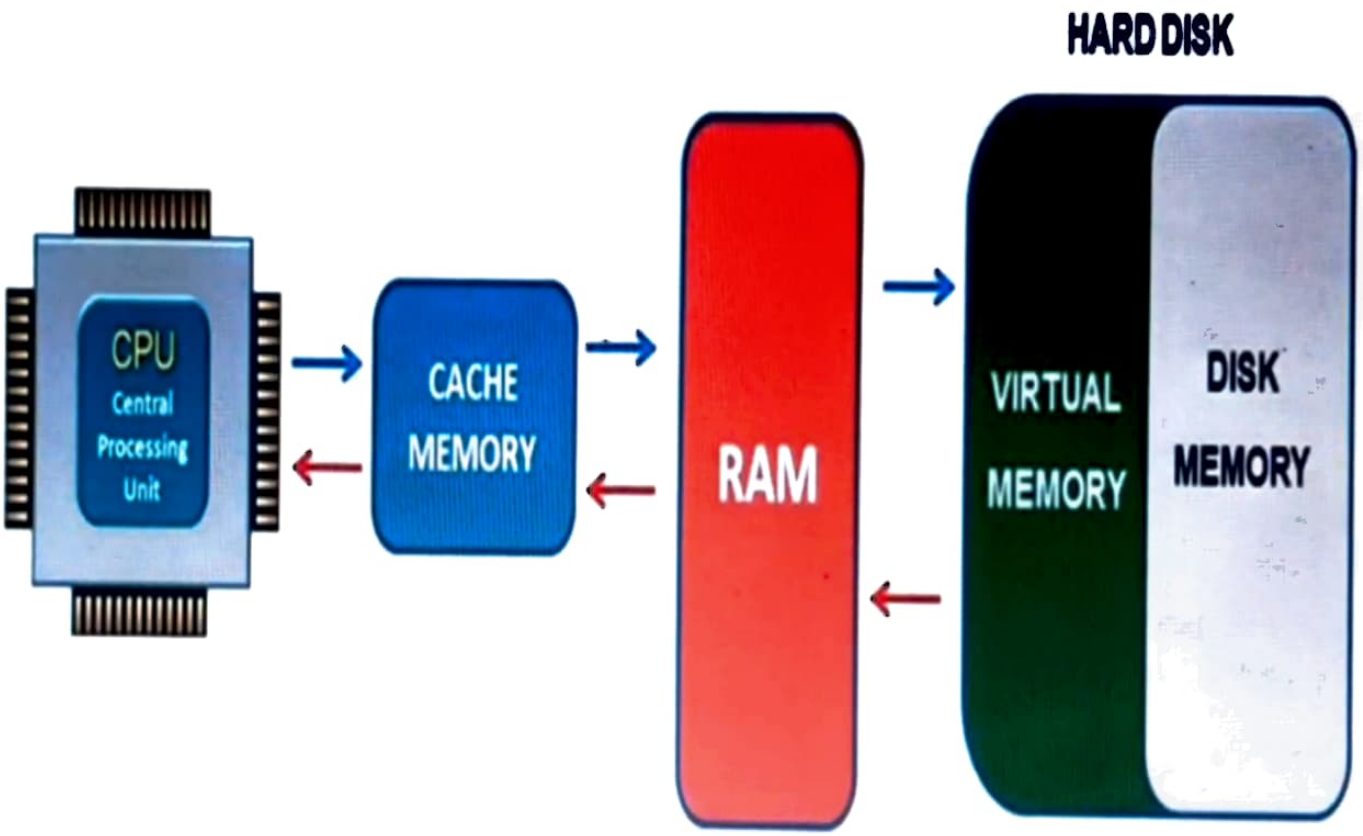
- It may miss the memory block/hole if it doesn`t start searching from the begining and memory block is available at the starting.

# Logical and Physical address map

- **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.

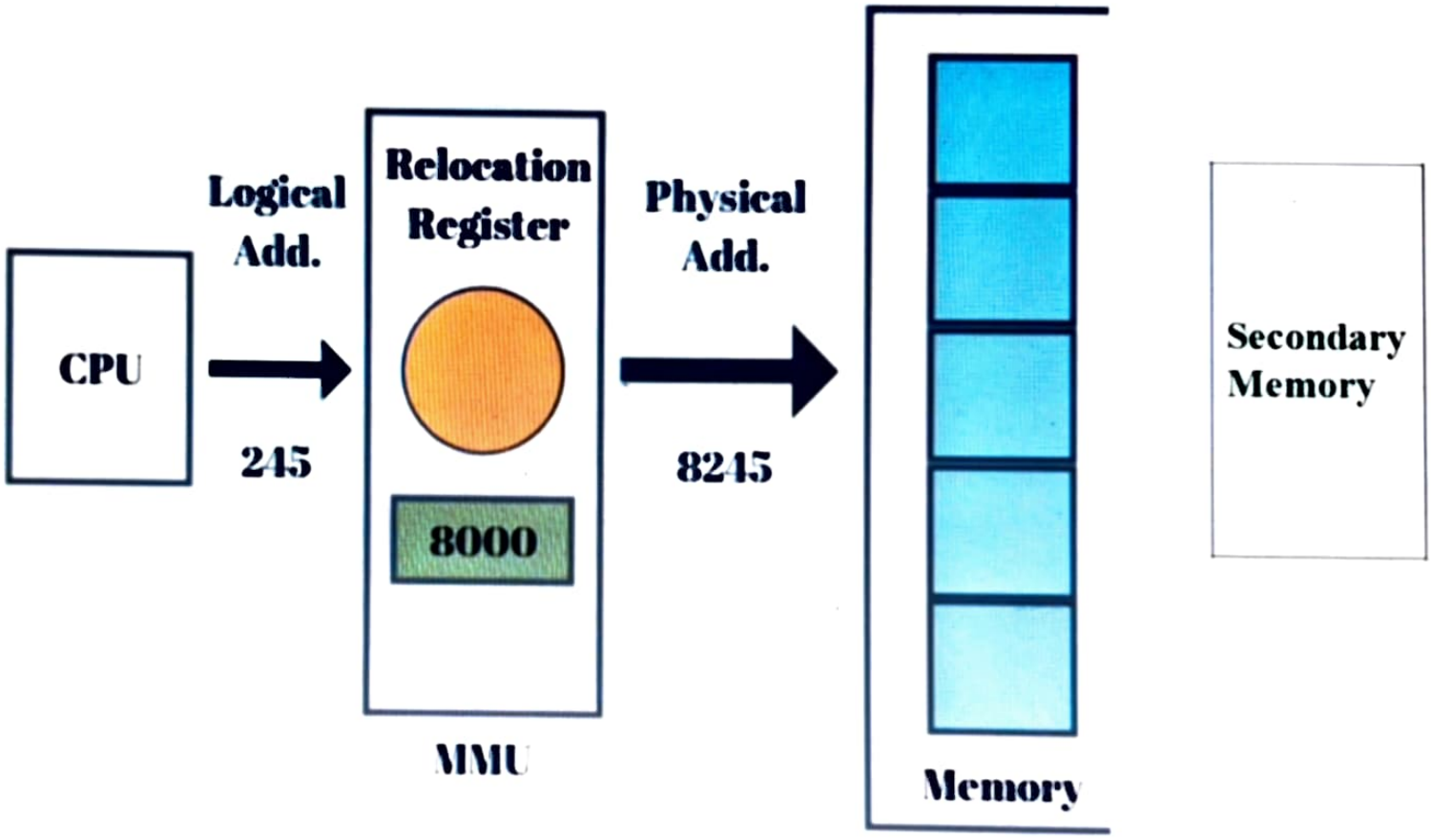
- The hardware device called Memory-Management Unit is used for mapping logical address to its corresponding physical address.


HARD DISK
CPU
Central
Processing
Unit
CACHE
MEMORY
RAM
VIRTUAL
MEMORY
DISK
MEMORY

- **Physical Address** identifies a physical location of required data in a memory. The user never directly deals with the physical address but can access by its corresponding logical address.

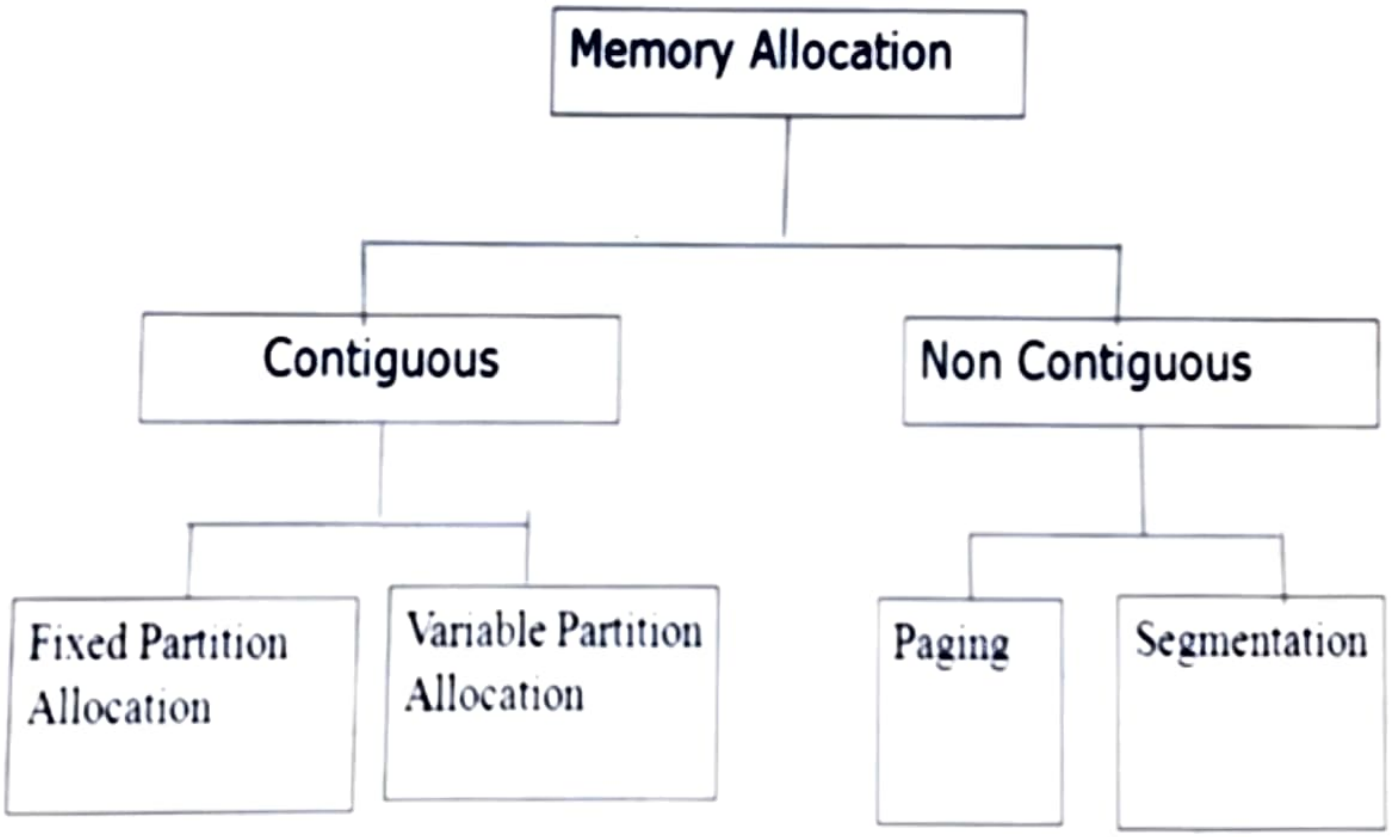
- The user program generates the logical address and thinks that the program is running in this logical address but the program needs physical memory for its execution, therefore, the logical address must be mapped to the physical address by MMU before they are used.

- The term Physical Address Space is used for all physical addresses corresponding to the logical addresses in a Logical address space.


CPU
Logical Add.
245
Relocation Register
8000
MMU
Physical Add.
8245
Memory
Secondary Memory

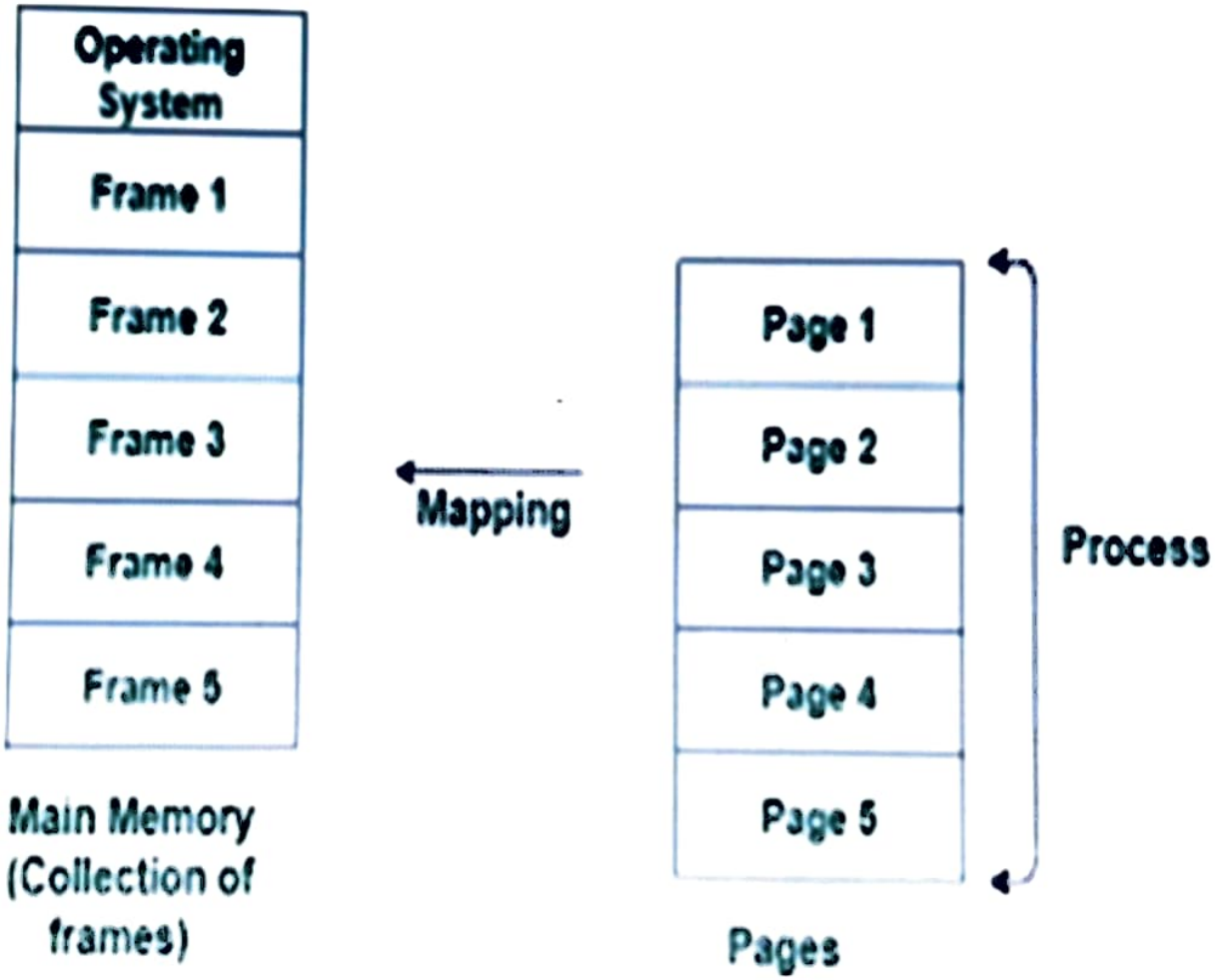
| Basis of comparison | Logical Address | Physical Address |
| --- | --- | --- |
| Basic | Virtually generated by CPU | Exists within the MMU |
| Visibility | Viewable. | Not viewable |
| Address Space | logical address space | physical address space |
| Access | Used to access physical address | Not directly accessed |
| Generation | Generated by the central processing unit. | Computed by the memory management unit. |
| Variation | variable | constant |

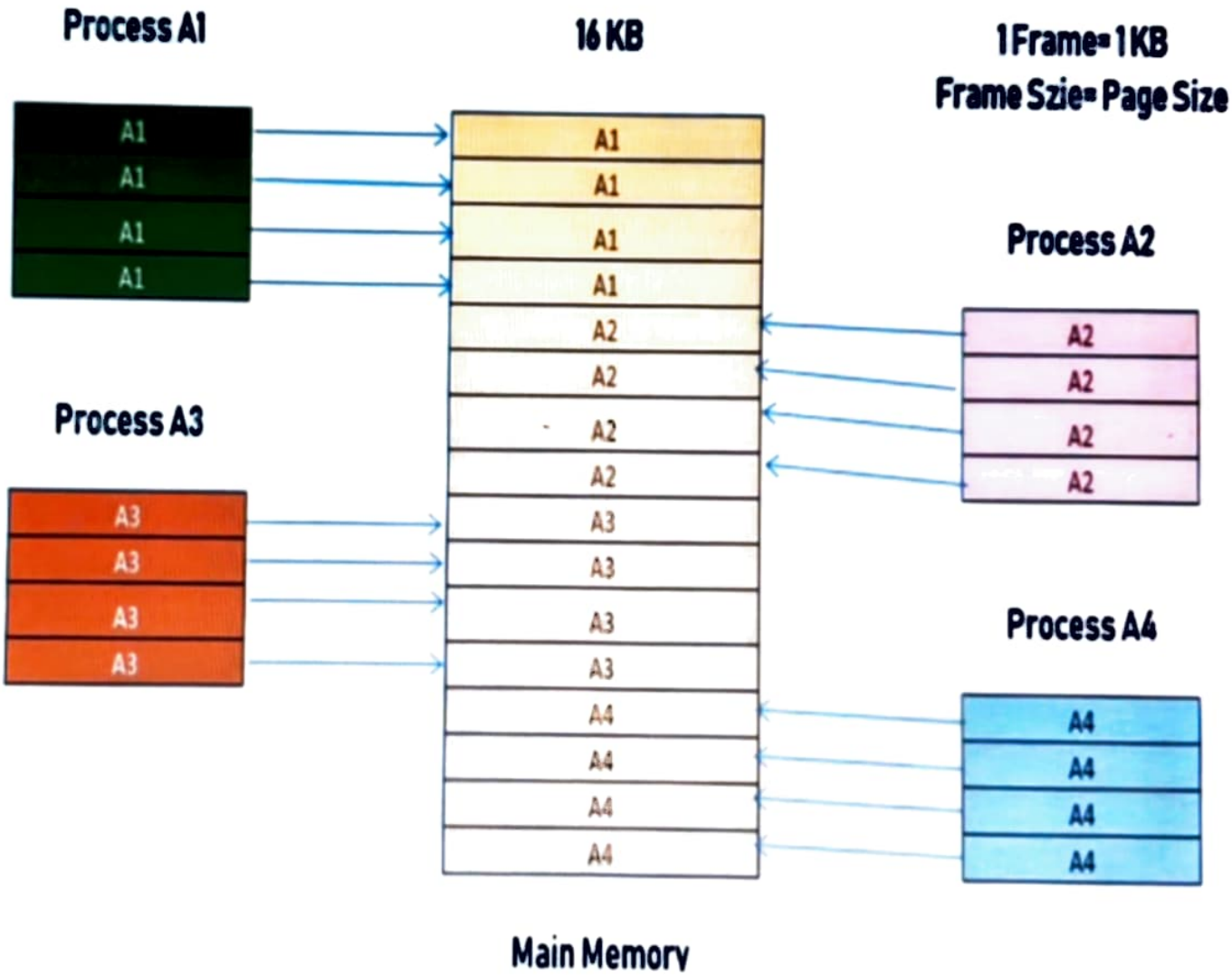
# Paging


Memory Allocation
Contiguous
Non Contiguous
Fixed Partition Allocation
Variable Partition Allocation
Paging
Segmentation

# Paging

- **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.


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


Process A1
16 KB
1 Frame= 1 KB
Frame Szie= Page Size
A1
A1
A1
A1
A1
A1
A1
A1
A2
A2
A2
A2
Process A2
A2
A2
A2
A2
Process A3
A3
A3
A3
A3
A3
A3
A3
A3
Process A4
A4
A4
A4
A4
A4
A4
A4
A4
Main Memory

- 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.

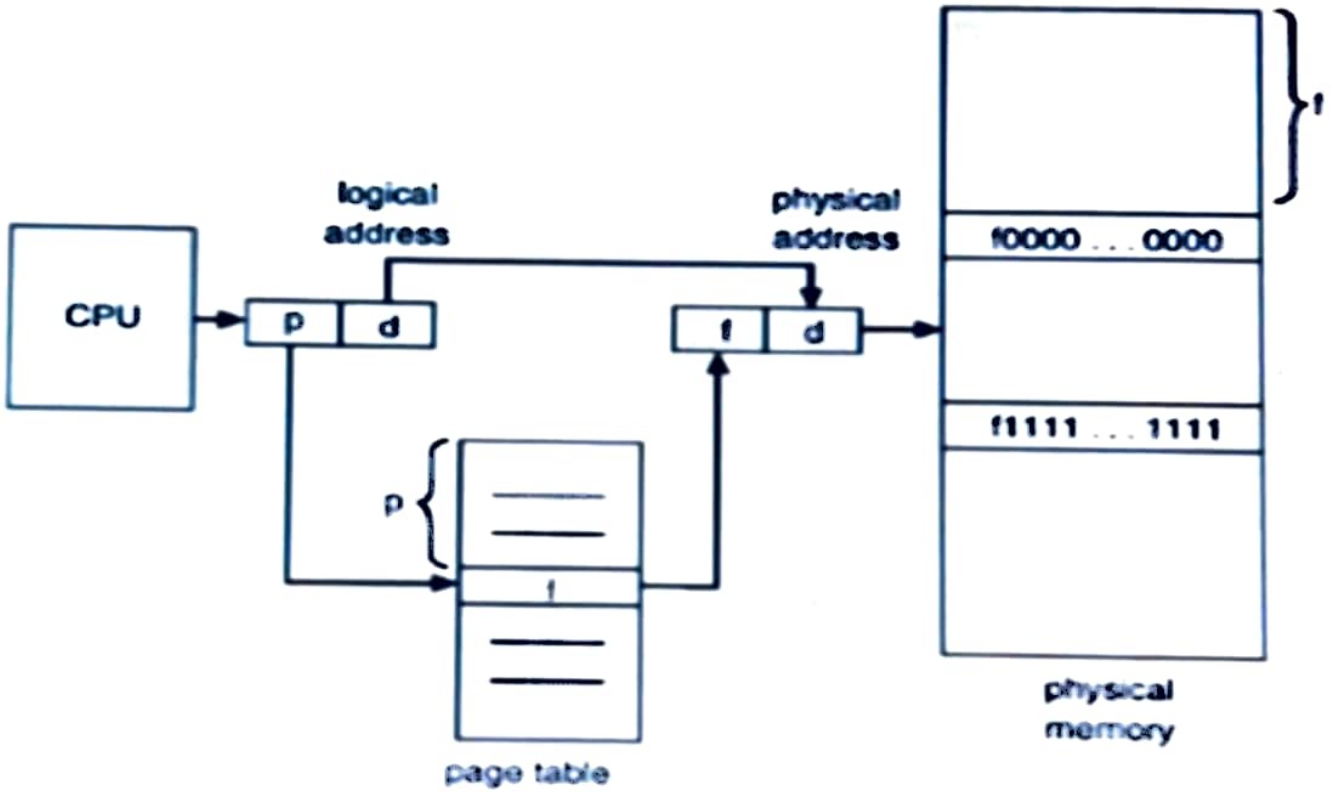



Fig. Address translation scheme

- 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:
- Page Number(p)
- Page Offset (d)
- **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.
- **Page offset** is mainly used to specify the specific word on the page that the CPU wants to read.

# Page Table in OS – For Address Mapping

- 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.

- The physical address consists of two parts:
- Page offset(d)
- Frame Number(f)
- where,
- The Frame number is used to indicate the specific frame where the required page is stored.
- and Page Offset indicates the specific word that has to be read from that page.
- The Page size (like the frame size) is defined with the help of hardware.

# Advantages of Paging

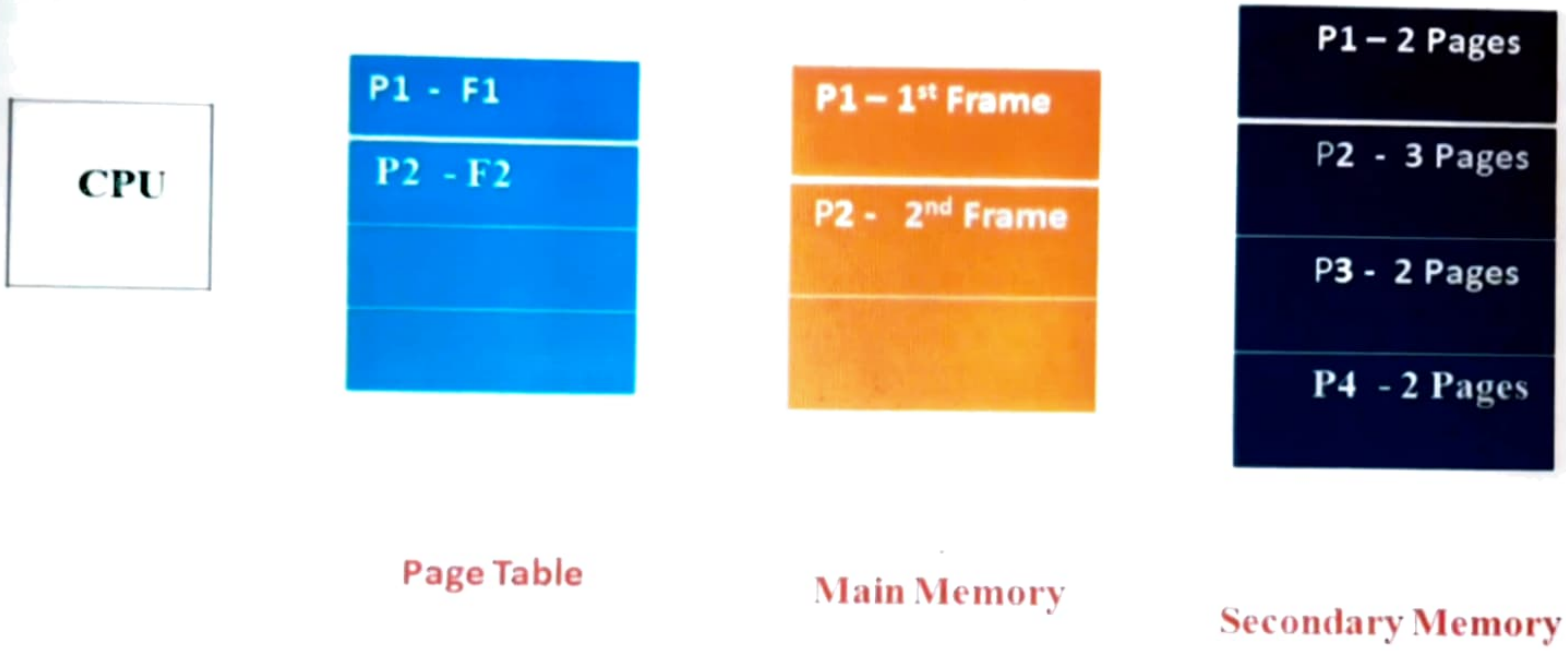
- Easy to use memory management algorithm
- Resolves the problem External Fragmentation
- Swapping is easy between equal-sized pages and page frames.

# Disadvantages of Paging

- May sometimes cause Internal fragmentation.
- Page tables consume additional memory.

# Virtual Memory

- **Virtual Memory** is a storage mechanism which offers user an illusion of having a very big main memory. It is done by treating a part of secondary memory as the main memory.
- In Virtual memory, the user can store processes with a bigger size than the available main memory.
- Therefore, instead of loading one long process in the main memory, the OS loads the various parts of more than one process in the main memory.


CPU
P1 - F1
P2 - F2
P1 – 1st Frame
P2 - 2nd Frame
P1 – 2 Pages
P2 - 3 Pages
P3 - 2 Pages
P4 - 2 Pages
Page Table
Main Memory
Secondary Memory

- Whenever your computer doesn't have space in the physical memory it writes what it needs to remember to the hard disk in a swap file as virtual memory.

- If a computer running Windows needs more memory/RAM, then installed in the system, it uses a small portion of the hard drive for this purpose.

- It is used whenever some pages require to be loaded in the main memory for the execution, and the memory is not available for those many pages.

- instead of preventing pages from entering in the main memory, the OS searches for the RAM space that are minimum used in the recent times or that are not referenced into the secondary memory to make the space for the new pages in the main memory

# Hardware and control structures

- A process may be broken up into a number of pieces and these pieces need not be contiguously located in main memory during the execution. The combination of dynamic run time address translation and the use of a page or segment table permit this.

- With virtual memory based on paging or segmentation, that job is left to the operating system and the hardware. As far as the programmer is concerned, they are dealing with a huge memory associated with disk storage. The operating system automatically loads pieces into main memory as required.

# Locality of Reference

- It refers to a phenomenon in which a computer program tends to access same set of memory locations for a particular time period.

- In other words, **Locality of Reference** refers to the tendency of the computer program to access instructions whose addresses are near one another.

- The property of locality of reference is mainly shown by loops and subroutine calls in a program.

- In case of loops in program control processing unit repeatedly refers to the set of instructions that constitute the loop.

- In case of subroutine calls, every time the set of instructions are fetched from memory.

- References to data items also get localized that means same data item is referenced again and again.

# Page Fault

- A page fault occurs when a program attempts to access a block of memory that is not stored in the physical memory, or RAM.

- The fault notifies the operating system that it must locate the data in virtual memory, then transfer it from the storage device, such as an HDD or SSD, to the system RAM

# Steps for handling page fault

- The memory address requested is first checked, to make sure it was a valid memory request.
- If the reference was invalid, the **process** is terminated. ...
- A free frame is located, possibly from a free-frame list.
- A disk operation is scheduled to bring in the necessary **page** from disk.

- CPU wants to read or fetch the data or instruction. First, it will access the cache memory as it is near to it and provides very fast access.

- If the required data or instruction is found, it will be fetched. This situation is known as a **cache hit**.

- But if the required data or instruction is not found in the cache memory then this situation is known as a cache miss.

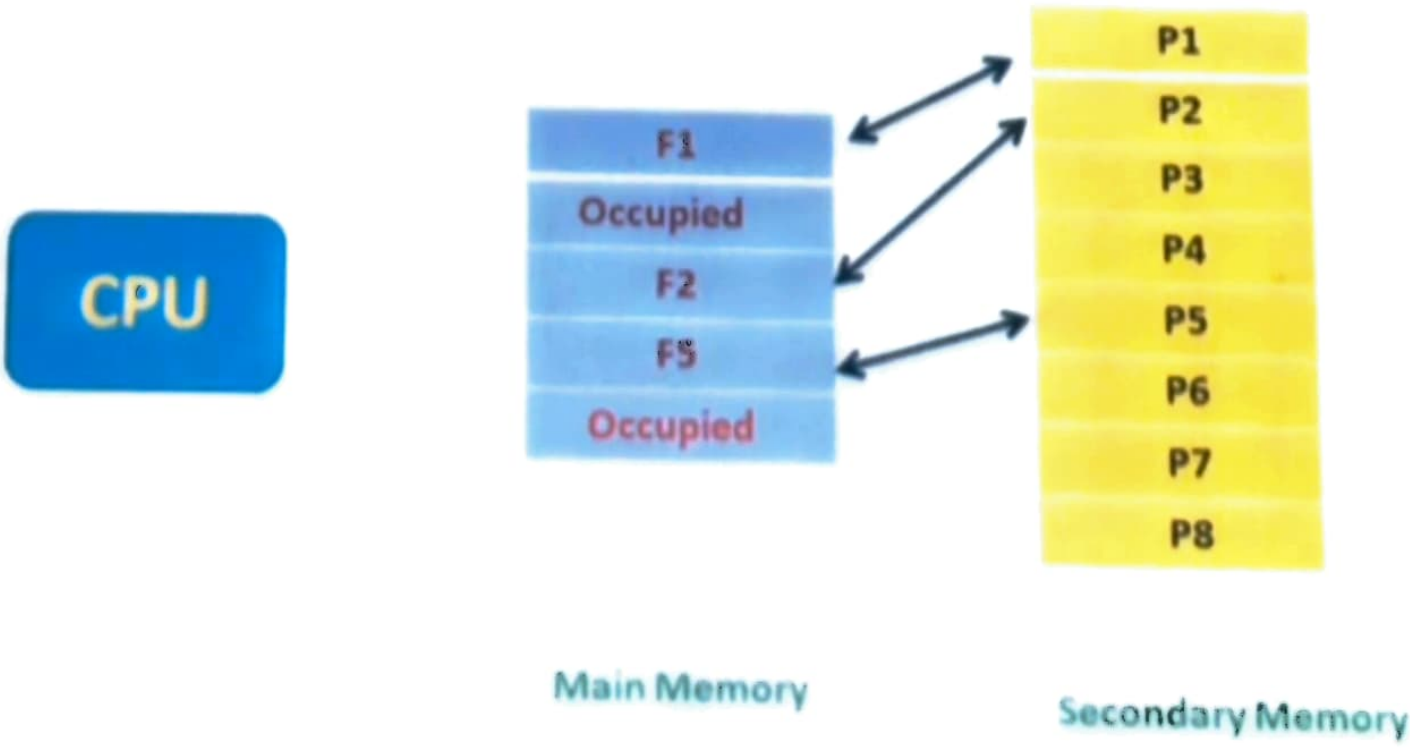
- Now the main memory will be searched for the required data or instruction that was being searched and if found will go through one of the two ways:

# Demand Paging

- According to the concept of Virtual Memory, in order to execute some process, only a part of the process needs to be present in the main memory which means that only a few pages will only be present in the main memory at any time.

- However, deciding, which pages need to be kept in the main memory and which need to be kept in the secondary memory, is going to be difficult because we cannot say in advance that a process will require a particular page at particular time.

- Therefore, to overcome this problem, there is a concept called Demand Paging is introduced. It suggests keeping all pages of the frames in the secondary memory until they are required. In other words, it says that do not load any page in the main memory until it is required.


CPU
F1
Occupied
F2
F5
Occupied
P1
P2
P3
P4
P5
P6
P7
P8
Main Memory
Secondary Memory

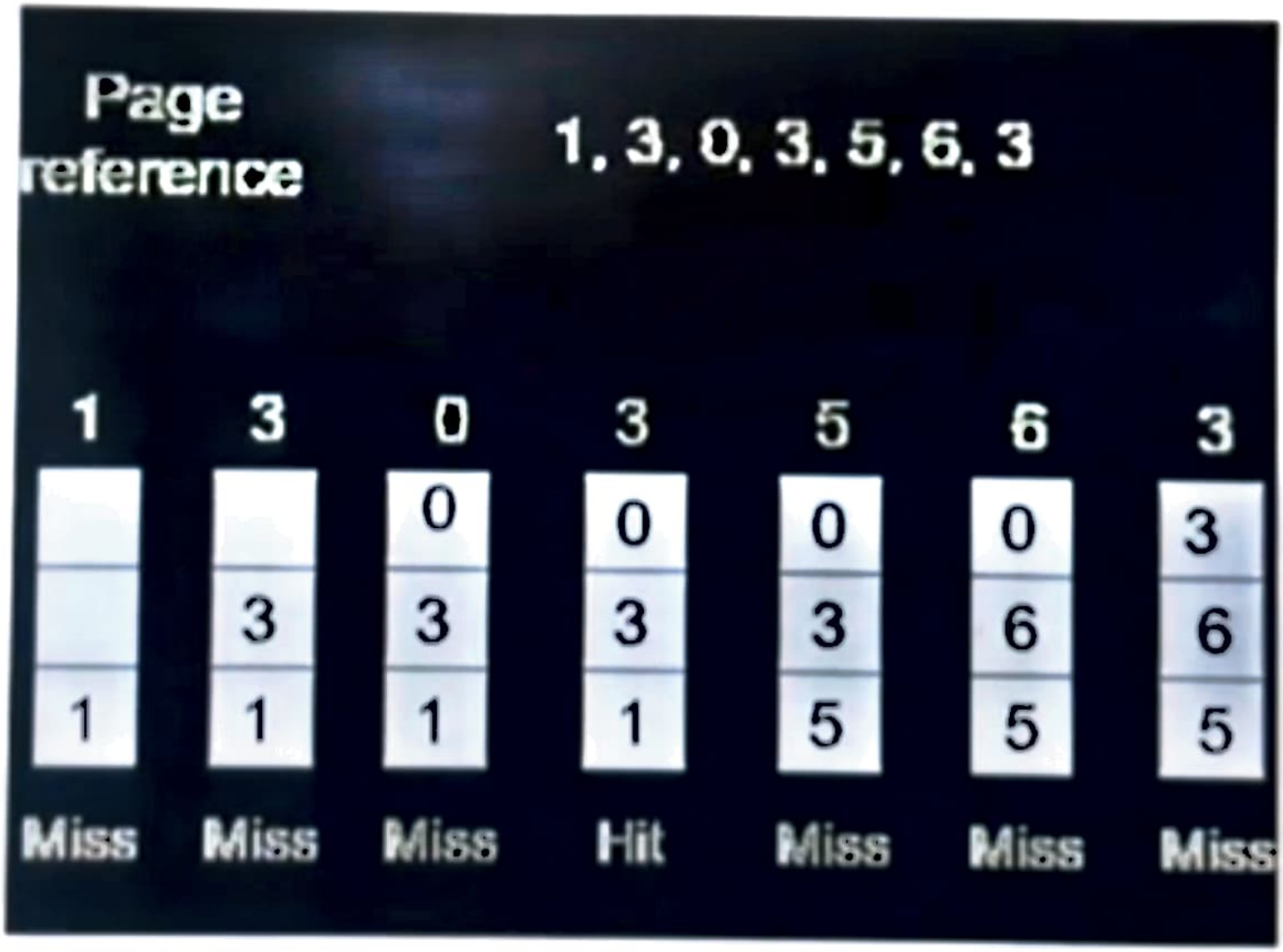
# Page Replacement algorithms

- First In First Out (FIFO)
- Optimal Page replacement
- Least Recently Used(LRU)
- Second Chance(SC)

# First In First Out (FIFO)

This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue.

- When a page needs to be replaced page in the front of the queue is selected for removal.

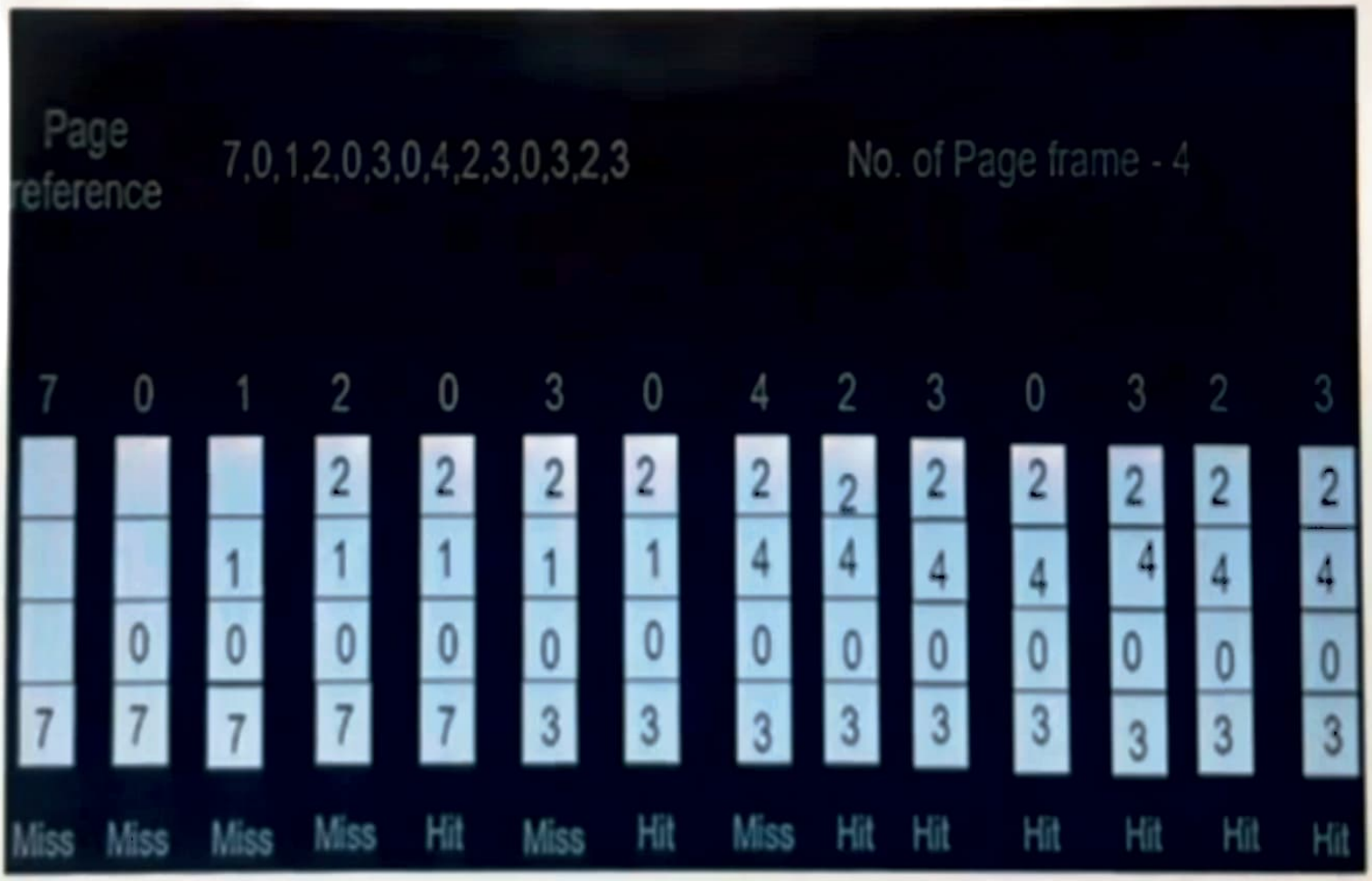

Page reference
1, 3, 0, 3, 5, 6, 3
1
3
0
3
5
6
3
0
3
1
0
3
1
0
3
1
0
6
5
3
6
5
Miss
Miss
Miss
Hit
Miss
Miss
Miss

# Optimal Page replacement

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.

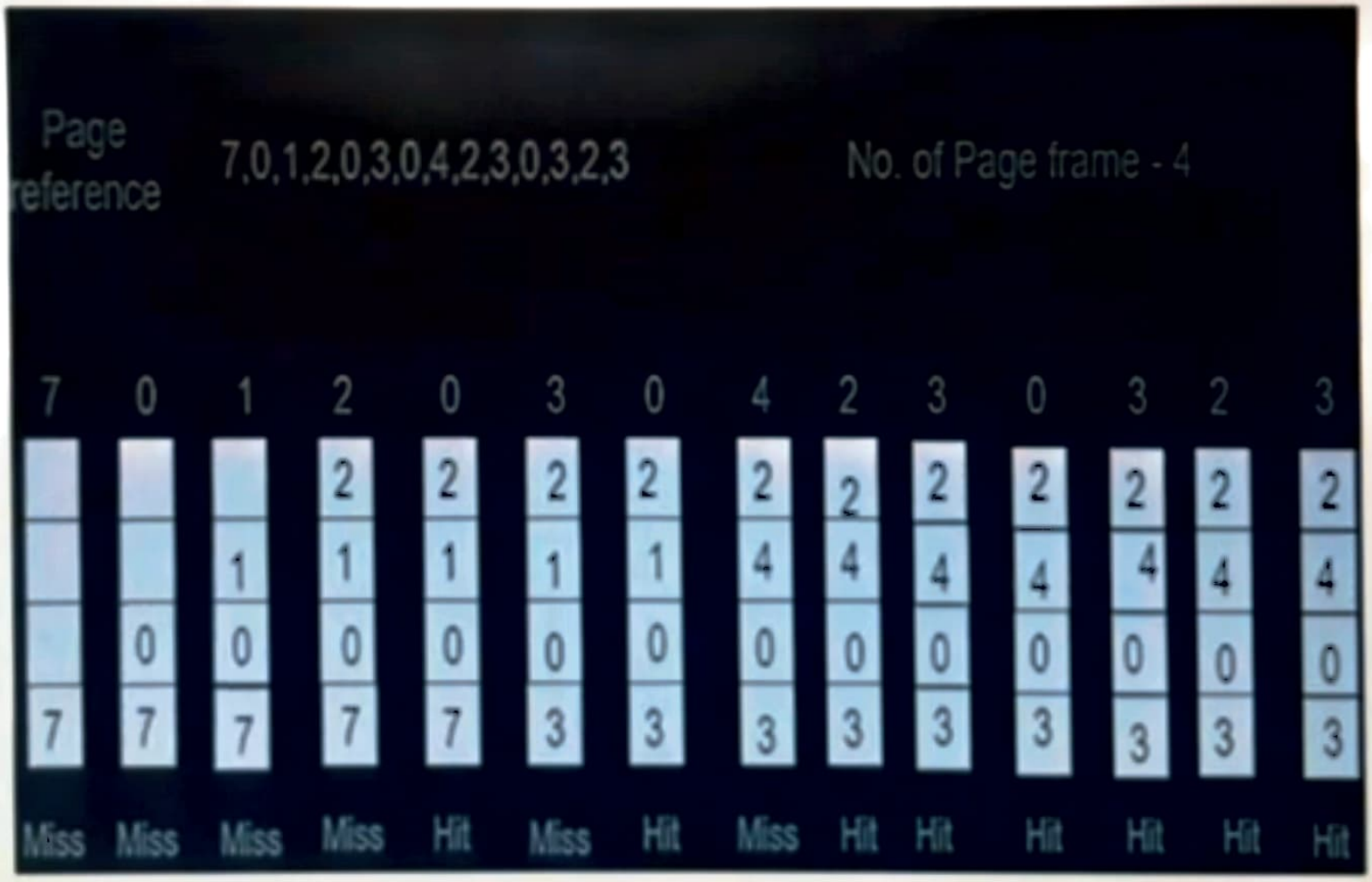
Page reference 7,0,1,2,0,3,0,4,2,3,0,3,2,3

No. of Page frame - 4



# Least Recently Used(LRU)

In this algorithm page will be replaced which is least recently used.


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