Memory Management

Basic memory management

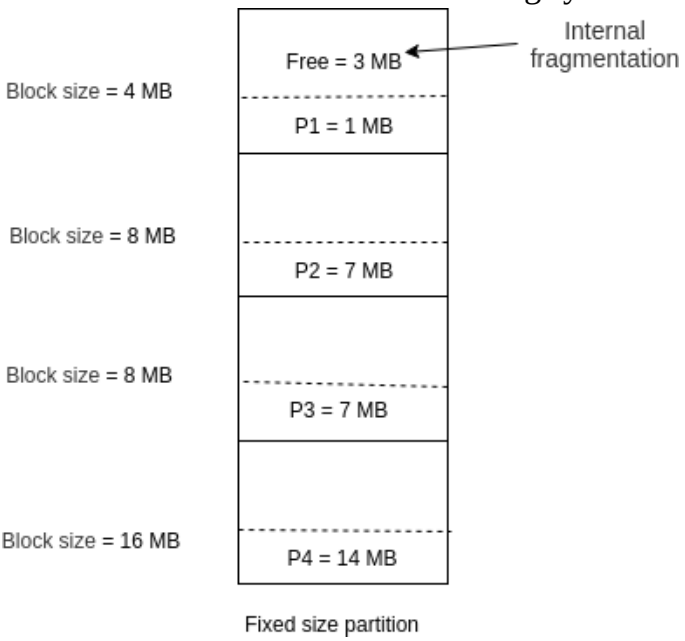
- Memory management is the functionality of an operating system which handles or manages primary memory and moves processes back and forth between main memory and disk during execution.
- Memory management keeps track of each and every memory location, regardless of either it is allocated to some process or it is free.
- It checks how much memory is to be allocated to processes.
- It decides which process will get memory at what time.
- It tracks whenever some memory gets freed or unallocated and correspondingly it updates the status.

There are two Memory Management Techniques: **Contiguous**, and **Non-Contiguous**.

- In Contiguous Technique, executing process must be loaded entirely in main-memory. In contiguous memory allocation each process is contained in a single contiguous block of memory. Memory is divided into several fixed size partitions.
- Each partition contains exactly one process.
- Contiguous Technique can be divided into:
    1. Fixed (or static) partitioning
    2. Variable (or dynamic) partitioning

Fixed Partitioning:

- This is the oldest and simplest technique used to put more than one processes in the main memory.
- In this partitioning, number of partitions (non-overlapping) in RAM are fixed but size of each partition may or may not be same.
- As it is contiguous allocation, hence no spanning is allowed.
- Here partitions are made before execution or during system configure.



Fixed size partition

Advantages of Fixed Partitioning –

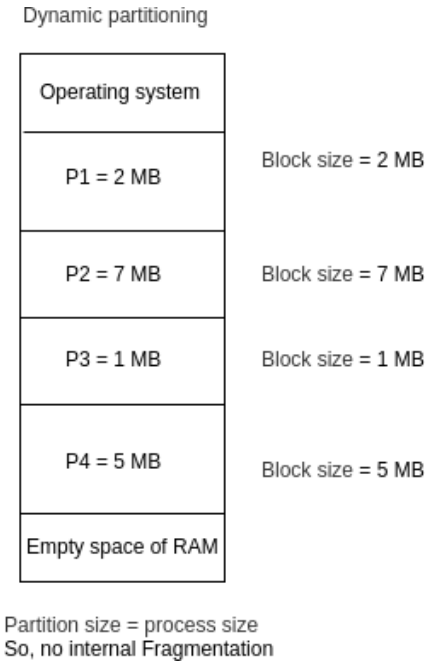
1. Easy to implement:
2. Little OS overhead:

Disadvantages of Fixed Partitioning –

- Internal Fragmentation
- External Fragmentation
- Limit process size
- Limitation on Degree of Multiprogramming

Variable Partitioning

- It is a part of Contiguous allocation technique.
- It is used to alleviate the problem faced by Fixed Partitioning.
- In contrast with fixed partitioning, partitions are not made before the execution or during system configure.
- Various **features** associated with variable Partitioning-
    1. Initially RAM is empty and partitions are made during the run-time according to process's need instead of partitioning during system configure.
    2. The size of partition will be equal to incoming process.
    3. The partition size varies according to the need of the process so that the internal fragmentation can be avoided to ensure efficient utilisation of RAM.
    4. Number of partitions in RAM is not fixed and depends on the number of incoming process and Main Memory's size.



Advantages of Variable Partitioning

1. No Internal Fragmentation
2. No restriction on Degree of Multiprogramming
3. No Limitation on the size of the process

Disadvantages of Variable Partitioning

1. Difficult Implementation
2. External Fragmentation

Free space management techniques in Operating System

- The system keeps tracks of the free disk blocks for allocating space to files when they are created.
- Also, to reuse the space released from deleting the files, free space management becomes crucial.
- The system maintains a free space list which keeps track of the disk blocks that are not allocated to some file or directory.
- The free space list can be implemented mainly as:

1. Bitmap or Bit vector:

- A Bitmap or Bit Vector is series or collection of bits where each bit corresponds to a disk block.
- The bit can take two values: 0 and 1: *0 indicates that the block is allocated* and *1 indicates a free block.*
- The given instance of disk blocks on the disk in *Figure 1* (where green blocks are allocated) can be represented by a bitmap of 16 bits as: **0000111000000110**.

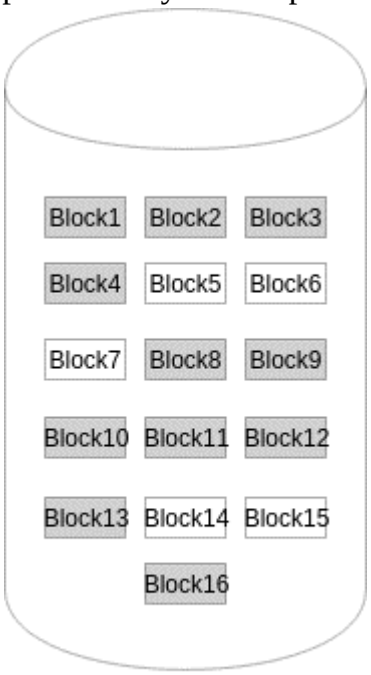


Figure - 1

2. Linked List

In this approach, the free disk blocks are linked together i.e. a free block contains a pointer to the next free block. The block number of the very first disk block is stored at a separate location on disk and is also cached in memory.

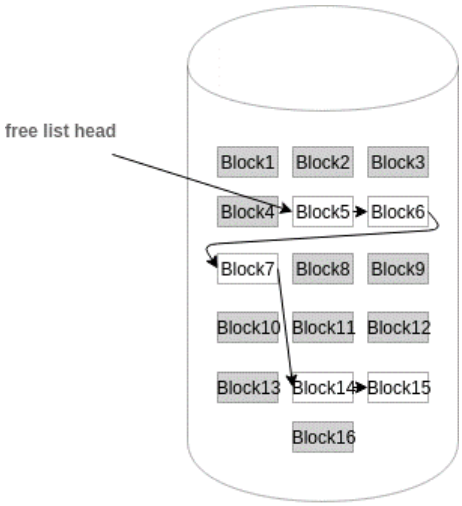


Figure - 2

In *Figure-2*, the free space list head points to Block 5 which points to Block 6, the next free block and so on. The last free block would contain a null pointer indicating the end of free list. A drawback of this method is the I/O required for free space list traversal.

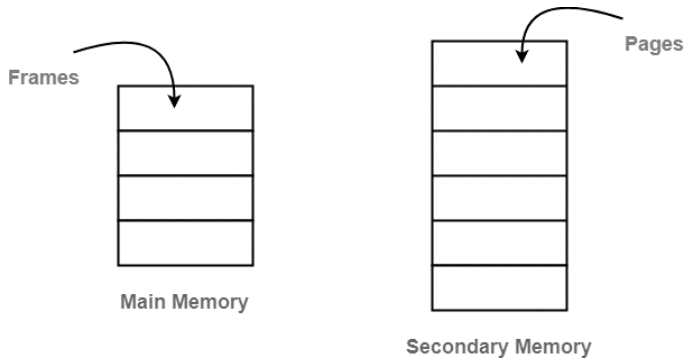
Virtual Memory

- Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of main memory.
- The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program generated addresses are translated automatically to the corresponding machine addresses.
- The size of virtual storage is limited by the addressing scheme of the computer system and amount of secondary memory is available not by the actual number of the main storage locations.
- It is a technique that is implemented using both hardware and software.
- It maps memory addresses used by a program, called virtual addresses, into physical addresses in computer memory.
    1. All memory references within a process are logical addresses that are dynamically translated into physical addresses at run time. This means that a process can be swapped in and out of main memory such that it occupies different places in main memory at different times during the course of execution.
    2. A process may be broken into number of pieces and these pieces need not be continuously located in the main memory during execution. The combination of dynamic run-time address translation and use of page or segment table permits this.

- If these characteristics are present then, it is not necessary that all the pages or segments are present in the main memory during execution.
- This means that the required pages need to be loaded into memory whenever required.
- Virtual memory is implemented using Demand Paging or Demand Segmentation.
- There are two ways in which virtual memory is handled: paging and segmentation.

Paging

- Paging 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.
- Paging is a fixed size partitioning scheme.
- In paging, secondary memory and main memory are divided into equal fixed size partitions.
- The partitions of secondary memory are called as **pages**.
- The partitions of main memory are called as **frames**.



- Each process is divided into parts where size of each part is same as page size.
- The size of the last part may be less than the page size.
- The pages of process are stored in the frames of main memory depending upon their availability.

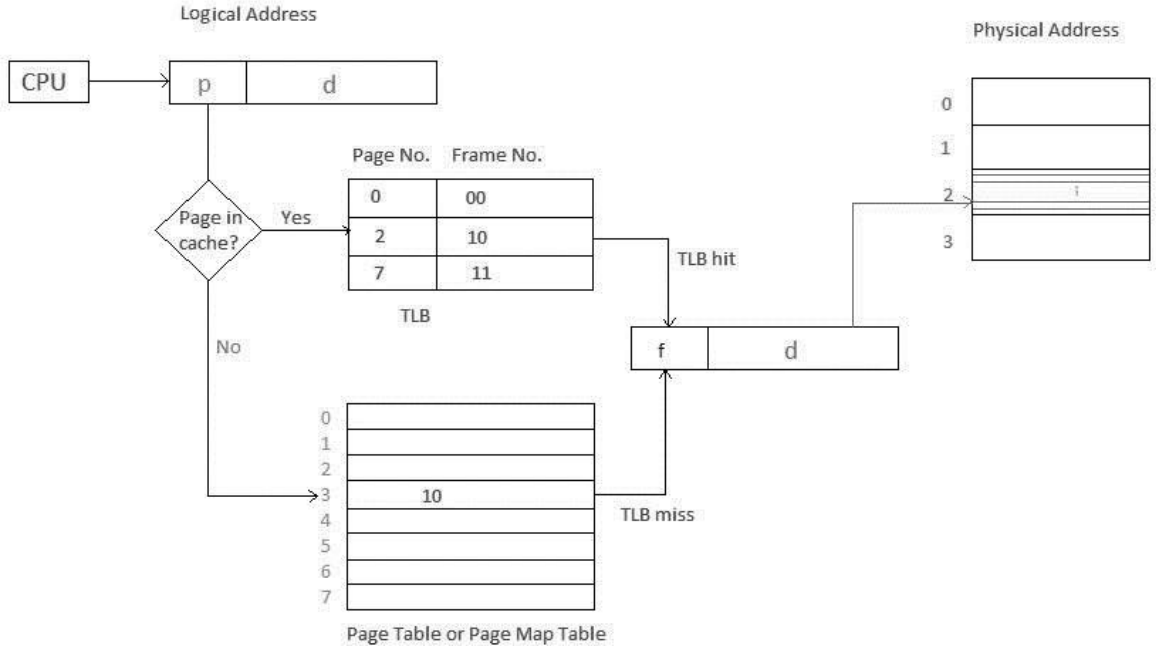
Address generated by CPU is divided into

- **Page number(p):** Number of bits required to represent the pages in Logical Address Space or Page number
- **Page offset(d):** Number of bits required to represent particular word in a page or page size of Logical Address Space or word number of a page or page offset.

Physical Address is divided into

- **Frame number(f):** Number of bits required to represent the frame of Physical Address Space or Frame number.
- **Frame offset(d):** Number of bits required to represent particular word in a frame or frame size of Physical Address Space or word number of a frame or frame offset.

- The hardware implementation of page table can be done by using dedicated registers.
- But the usage of register for the page table is satisfactory only if page table is small.
- If page table contain large number of entries then we can use TLB (translation Look-aside buffer), a special, small, fast look up hardware cache.
  1. The TLB is associative, high speed memory.
  2. Each entry in TLB consists of two parts: a tag and a value.
  3. When this memory is used, then an item is compared with all tags simultaneously. If the item is found, then corresponding value is returned.



Advantages-

The advantages of paging are-

- It allows to store parts of a single process in a non-contiguous fashion.
- It solves the problem of external fragmentation.

Disadvantages-

The disadvantages of paging are-

- It suffers from internal fragmentation.
- There is an overhead of maintaining a page table for each process.
- The time taken to fetch the instruction increases since now two memory accesses are required.

Page Table-

- Page table is a data structure.
- It maps the page number referenced by the CPU to the frame number where that page is stored.

Characteristics-

- Page table is stored in the main memory.

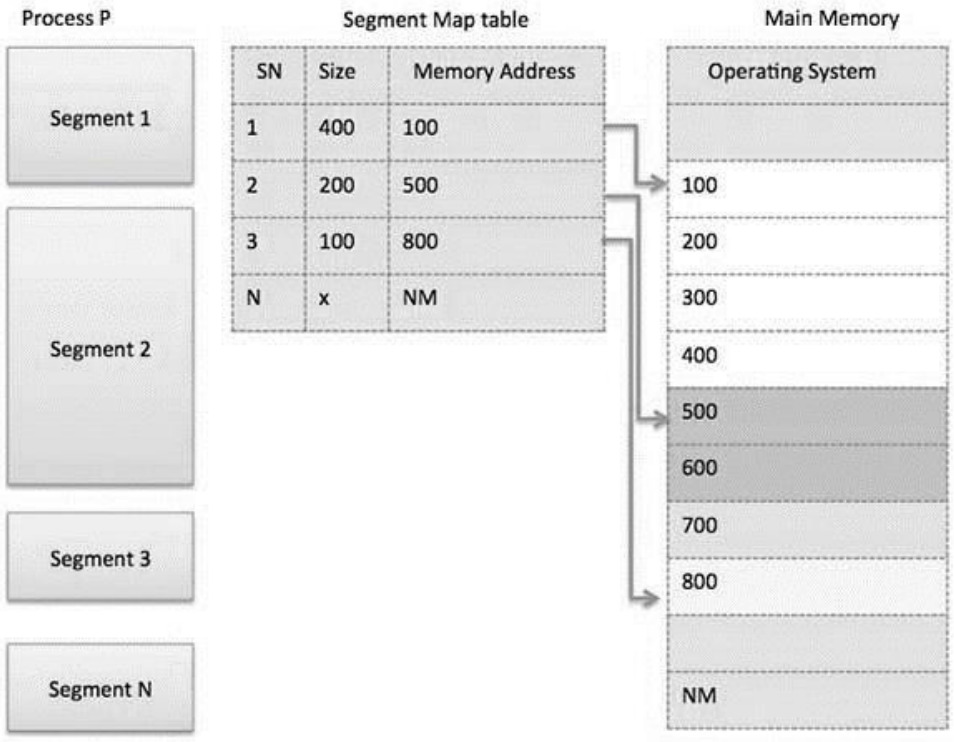
- Number of entries in a page table = Number of pages in which the process is divided.
- Page Table Base Register (PTBR) contains the base address of page table.
- Each process has its own independent page table.
- Page Table Base Register (PTBR) provides the base address of the page table.
- The base address of the page table is added with the page number referenced by the CPU.
- It gives the entry of the page table containing the frame number where the referenced page is stored.

Page Table Entry format:

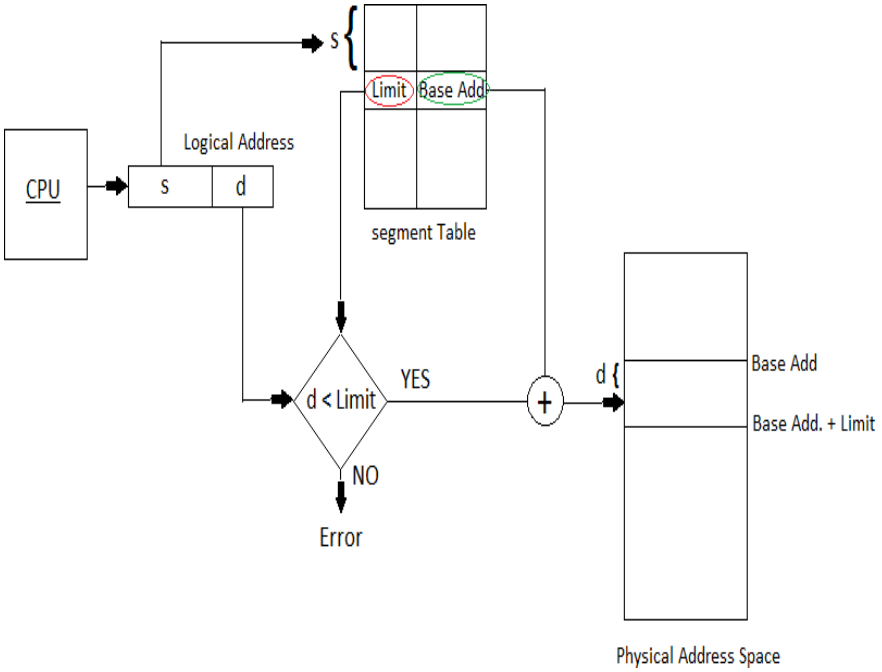
- A page table entry contains several information about the page.
- The information contained in the page table entry varies from operating system to operating system.
- The most important information in a page table entry is frame number, Present / Absent Bit, Protection Bit, Reference Bit, Dirty Bit (Modified bit)

Segmentation

- Segmentation is a memory management technique in which each job is divided into several segments of different sizes, one for each module that contains pieces that perform related functions.
- Each segment is actually a different logical address space of the program.
- When a process is to be executed, its corresponding segmentation are loaded into non-contiguous memory though every segment is loaded into a contiguous block of available memory.
- Segmentation memory management works very similar to paging but here segments are of variable-length where as in paging pages are of fixed size.
- The operating system maintains a segment map table for every process and a list of free memory blocks along with segment numbers, their size and corresponding memory locations in main memory.
- For each segment, the table stores the starting address of the segment and the length of the segment.
- A reference to a memory location includes a value that identifies a segment and an offset.
- **Segment Table –** It maps two-dimensional Logical address into one-dimensional Physical address. It's each table entry has:
    1. **Base Address:** It contains the starting physical address where the segments reside in memory.
    2. **Limit:** It specifies the length of the segment.



Translation of Two Dimensional Logical Address to one dimensional Physical Address.



Address generated by the CPU is divided into:

- **Segment number (s):** Number of bits required to represent the segment.
- **Segment offset (d):** Number of bits required to represent the size of the segment.

Advantages of Segmentation –

- No Internal fragmentation.

- Segment Table consumes less space in comparison to Page table in paging.

Disadvantage of Segmentation –

- As processes are loaded and removed from the memory, the free memory space is broken into little pieces, causing External fragmentation.

Fragmentation

- As processes are loaded and removed from memory, the free memory space is broken into little pieces.
- It happens after sometimes that processes cannot be allocated to memory blocks considering their small size and memory blocks remains unused.
- This problem is known as Fragmentation.

Fragmentation is of two types –

S.N.	Fragmentation & Description
1	**External fragmentation:** Total memory space is enough to satisfy a request or to reside a process in it, but it is not contiguous, so it cannot be used.
2	**Internal fragmentation:** Memory block assigned to process is bigger. Some portion of memory is left unused, as it cannot be used by another process.

Page Fault:

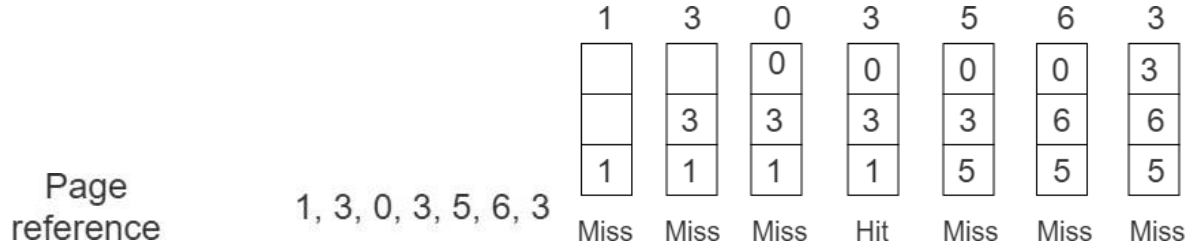
- A page fault occurs when a program attempts to access a block of memory that is not stored in the physical memory, or RAM.
- The fault notifies the operating system that it must locate the data in virtual memory, then transfer it from the storage device, such as an HDD or SSD, to the system RAM.

Page Replacement Algorithms

1. 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.

Example-1 Consider page reference string 1, 3, 0, 3, 5, 6 with 3 page frames. Find number of page faults.

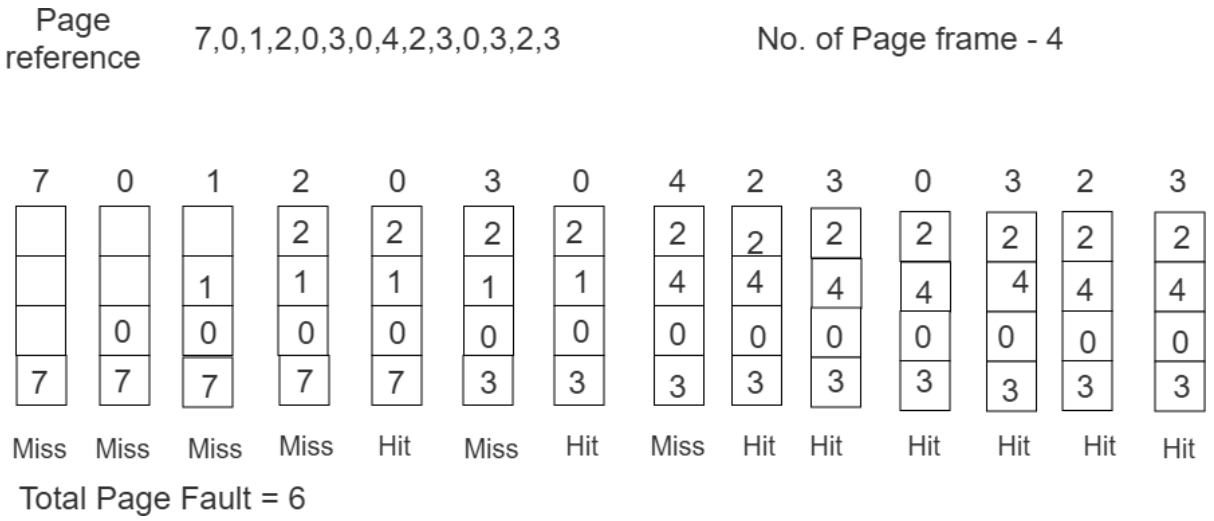


Total Page fault =6

Initially all slots are empty, so when 1, 3, 0 came they are allocated to the empty slots —> **3 Page Faults.**
when 3 comes, it is already in memory so —> **0 Page Faults.**
Then 5 comes, it is not available in memory so it replaces the oldest page slot i.e 1. —>**1 Page Fault.**
6 comes, it is also not available in memory so it replaces the oldest page slot i.e 3 —>**1 Page Fault.**
Finally when 3 come it is not available so it replaces 0 **1 page fault**
Hence Fage Fault Ratio = 6/7 = 0.85

2. Optimal Page replacement

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.
Example-2: Consider the page references 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, with 4-page frame. Find number of page fault.



Hence Fage Fault Ratio = 6/14 = 0.85

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 not used for the longest duration of time in the future.—>**1 Page fault.**
0 is already there so —> **0 Page fault.**.
4 will takes place of 1 —> **1 Page Fault.**

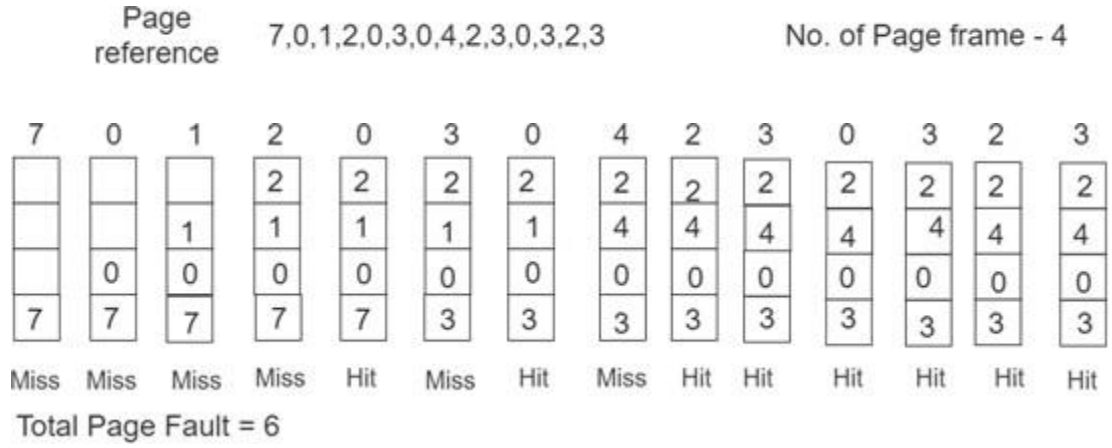
Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

Optimal page replacement is perfect, but not possible in practice as the operating system cannot know future requests. The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analyzed against it.

3. Least Recently Used –

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

Example-3 Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2 with 4 page frames. Find number of page faults.



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 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.

Q. Consider a reference string: 4, 7, 6, 1, 7, 6, 1, 2, 7, 2. the number of frames in the memory is 3. Find out the number of page faults respective to:

1. Optimal Page Replacement Algorithm
2. FIFO Page Replacement Algorithm
3. LRU Page Replacement Algorithm

Optimal Page Replacement Algorithm

Request	4	7	6	1	7	6	1	2	7	2
Frame 3			6	6	6	6	6	2	2	2
Frame 2		7	7	7	7	7	7	7	7	7
Frame 1	4	4	4	1	1	1	1	1	1	1
Miss/Hit	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Hit	Hit

Number of Page Faults in Optimal Page Replacement Algorithm = 5

LRU Page Replacement Algorithm

Request	4	7	6	1	7	6	1	2	7	2
Frame 3			6	6	6	6	6	6	7	7
Frame 2		7	7	7	7	7	7	2	2	2
Frame 1	4	4	4	1	1	1	1	1	1	1
Miss/Hit	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Miss	Hit

Number of Page Faults in LRU = 6

FIFO Page Replacement Algorithm

Request	4	7	6	1	7	6	1	2	7	2
Frame 3			6	6	6	6	6	6	7	7
Frame 2		7	7	7	7	7	7	2	2	2
Frame 1	4	4	4	1	1	1	1	1	1	1
Miss/Hit	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Miss	Hit

Number of Page Faults in FIFO = 6

Q. Given page reference string: 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6. Compare the number of page faults for LRU, FIFO and Optimal page replacement algorithm using 3 and 4 Frame