Memory Management

Jayesh Deep Dubey
Technical Officer
C-DAC Patna

What is Memory Management?

- Memory management is an important aspect of operating systems (OS) that involves allocating and managing memory resources in a computer system.
- The OS is responsible for ensuring that the memory is used efficiently and effectively by different processes running on the system.
- Effective memory management is critical for the efficient operation of an operating system, and various techniques and algorithms are used to optimize memory usage and performance.

Types of Memories

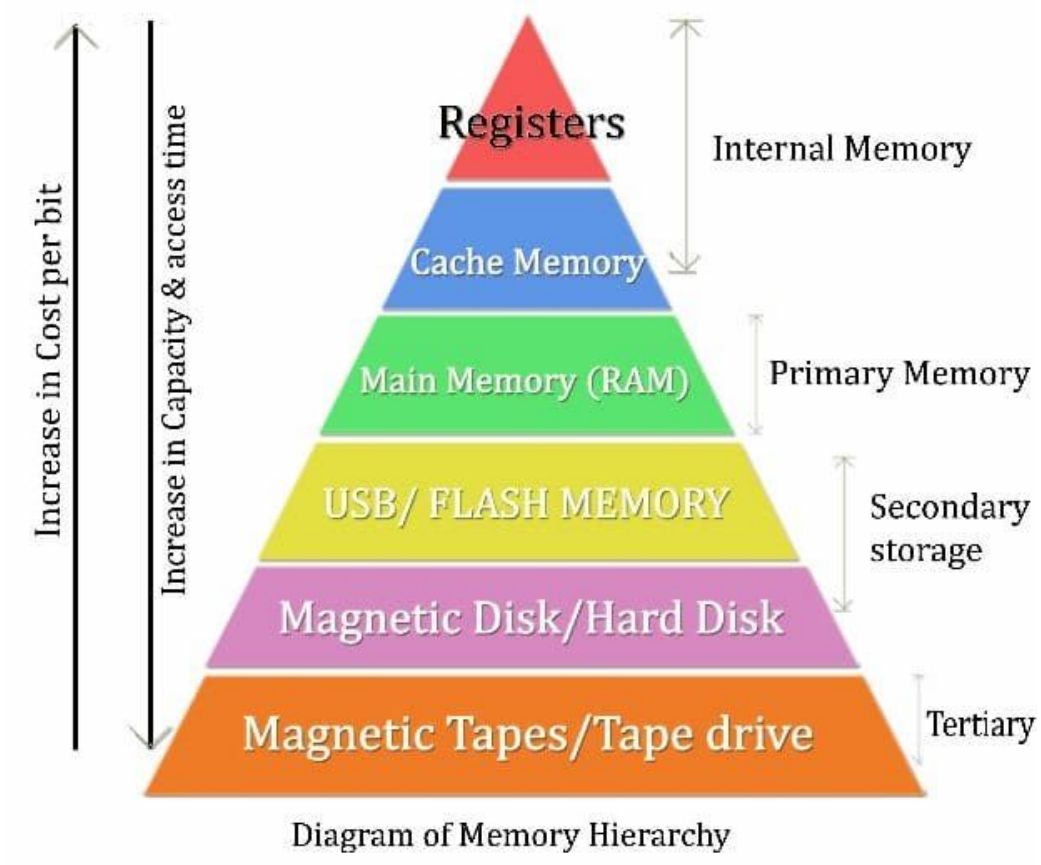


Diagram of Memory Hierarchy

Types of Memories

- **Register**
  - Register is a type of internal memory within a CPU (Central Processing Unit) that stores data and instructions temporarily during the execution of a program. Registers are high-speed memory locations that are built into the CPU, and they are used to hold data that is being processed by the CPU.
  - Registers are very fast compared to other types of memory because they are located within the CPU itself, and they can be accessed much more quickly than other types of memory such as RAM or cache memory

Types of Memories

- **Cache**
  - Cache memory is a type of high-speed memory that is used by the CPU (Central Processing Unit) to store frequently used data and instructions for faster access. It is a small amount of memory that is located on the CPU chip or near the CPU, and it operates at a much faster speed than main memory (RAM).
  - Cache memory is used to reduce the time it takes for the CPU to access data and instructions from memory. When the CPU needs to access data or instructions, it first checks the cache memory. If the data is found in the cache memory, it is accessed much more quickly than if it had to be retrieved from the main memory

Types of Memories

- **RAM**
  - RAM (Random Access Memory) is a type of computer memory that is used to temporarily store data and program instructions that the CPU (Central Processing Unit) needs to access quickly.
  - RAM is volatile memory, which means that it can store data only while power is being supplied to the computer. When the computer is turned off, the data stored in RAM is lost.

Types of Memories

- **Flash Memory**
  - Flash memory is a type of non-volatile computer storage that retains data even when there is no power supply. It is a popular storage medium for portable devices, such as USB flash drives, digital cameras, and mobile phones, as well as for solid-state drives (SSDs) in laptops and desktop computers.
  - Flash memory works by storing data in a series of memory cells, which are arranged in blocks. Each memory cell can be electrically programmed and erased, allowing data to be written, read, and rewritten many times
  - It can read and write data much faster than traditional hard drives.

Types of Memories

- **Magnetic Disk and Hard Disk**
  - Magnetic disk and hard disk are both types of secondary storage devices that use magnetism to store data.
  - Magnetic disks are used for storing data in devices such as floppy disks, magnetic tape, and early hard drives.A hard disk, on the other hand, is a type of magnetic disk that is used for storing data in modern computers.
  - Magnetic disks are used in older types of storage devices such as floppy disks and magnetic tape, while hard disks are the primary storage device in modern computers.

Types of Memories

- **Magnetic Tapes and Tape Drives**
  - They come under tertiary storage and offer much larger storage capacity than other types of storage devices, and they are ideal for long-term storage of large amounts of data that is accessed infrequently.
  - Tape drives typically use a sequential access method, which means that data is stored in a linear fashion on the tape, and it must be read or written in the same order. This makes tape drives slower than other types of storage devices

Memory Allocation Techniques

- Memory allocation is the process of assigning a portion of a computer's memory to a program or process for its use.There are two types of memory allocation technique:
  - Contiguous Allocation
  - Non Contiguous Allocation

Memory Allocation Techniques

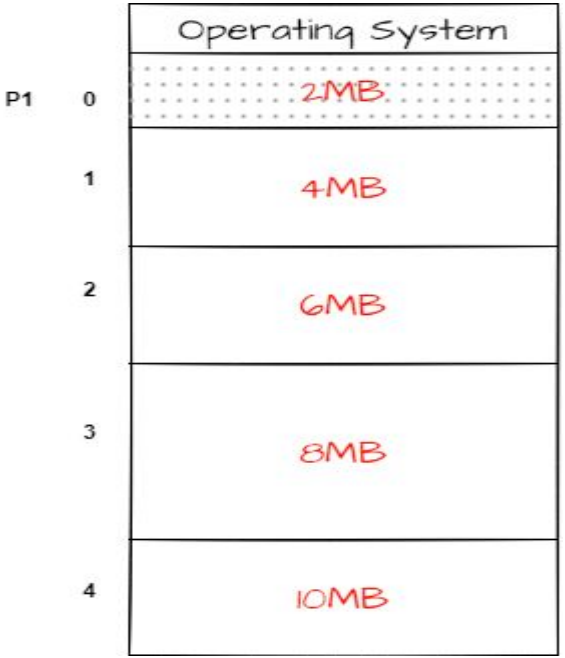
- **Contiguous Allocation**
  - In contiguous memory allocation, a process is allocated a single block of contiguous memory that is not broken into parts. This means that the entire process, including its code, data, and stack segments, is loaded into a contiguous block of memory when it is loaded into memory.
  - Contiguous Memory allocation is further divided into two parts:
    - Fixed Partitioning (Static Partitioning)
    - Variable Partitioning (Dynamic Partitioning)

Memory Allocation Techniques

- **Fixed Partitioning(Static Partitioning)**
  - This is the oldest and simplest technique used to put more than one process in the main memory
  - In this partitioning, the number of partitions in RAM is fixed but the size of each partition may or may not be the same.
  - Partitions are made before execution or during system configuration.
  - This technique suffers from internal and external fragmentation.

Memory Allocation Techniques

- **Fixed Partitioning**

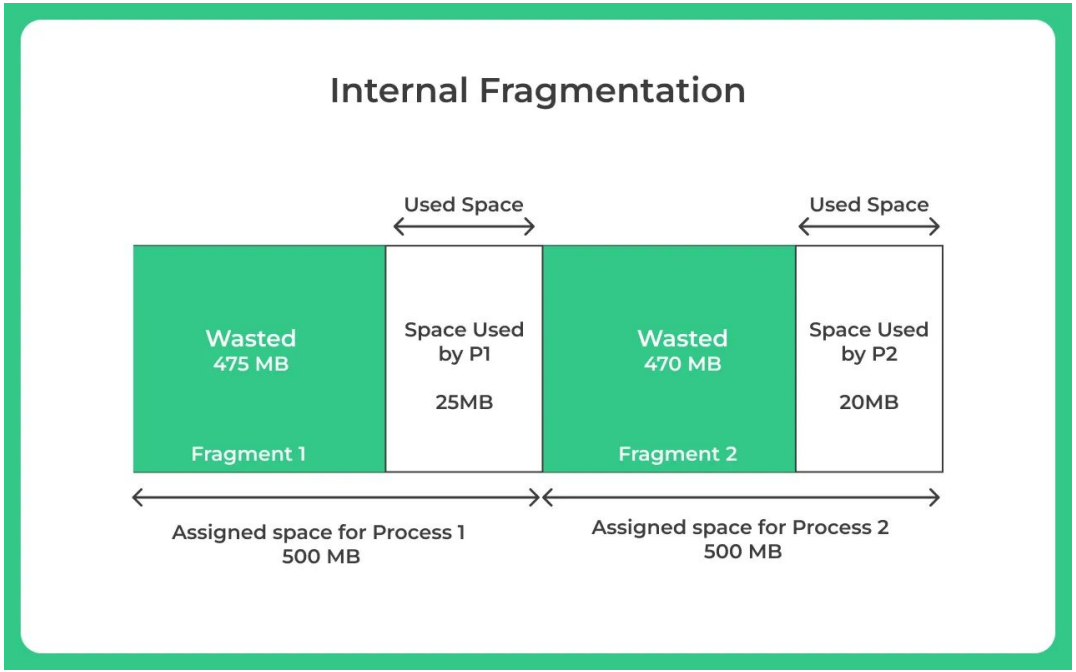


Memory Allocation Techniques

- **Internal and External Fragmentation**
  - **Internal fragmentation** is defined as the difference between memory allocated and the memory space required by a process. Internal fragmentation occurs when the memory allocated to process is larger than the memory required.
  - **External fragmentation** is the unused space that is left between the fragments of non-contiguous memory. These unused spaces are too small to help a new process. The 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.

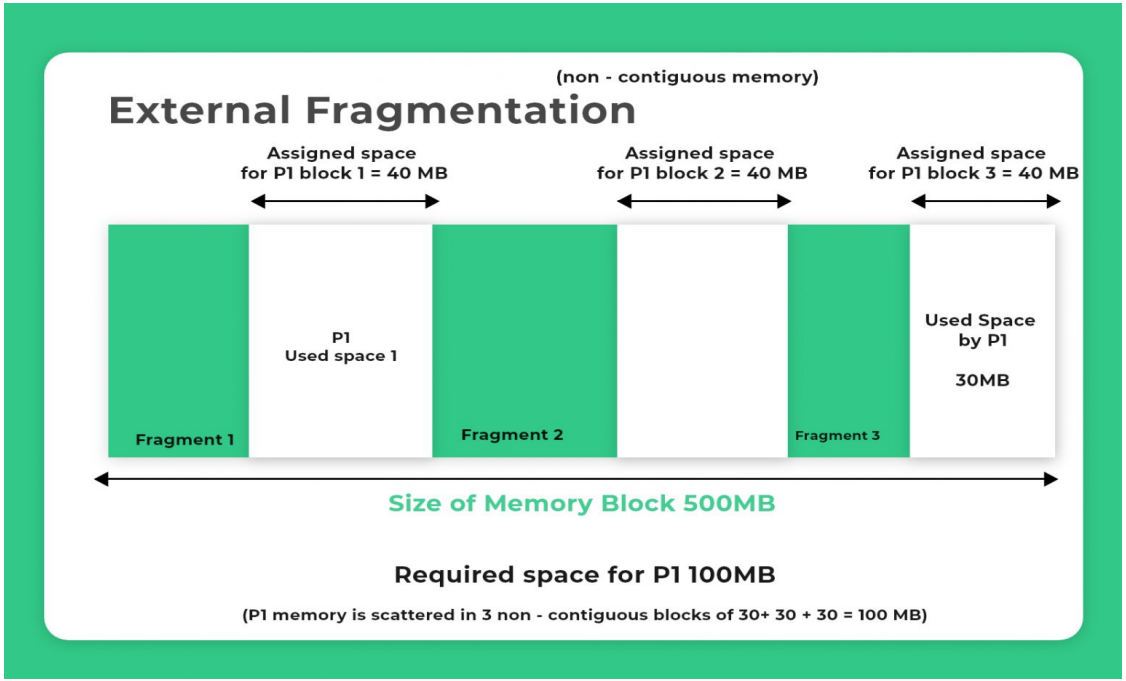
Memory Allocation Techniques

- **Internal and External Fragmentation**



Memory Allocation Techniques

- **Internal and External Fragmentation**

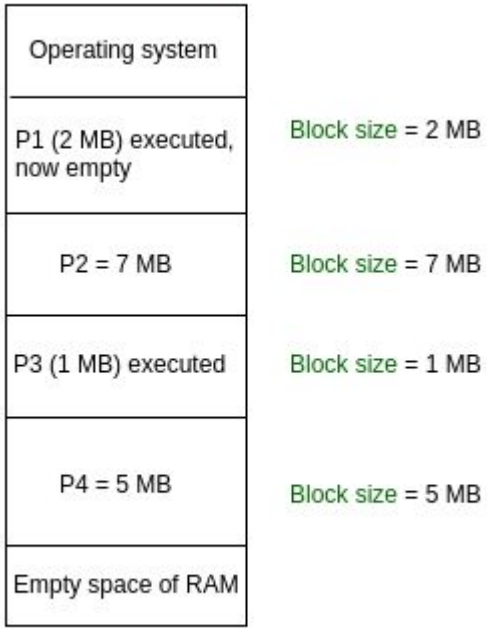


Memory Allocation Techniques

- **Variable Partitioning(Dynamic Partitioning)**
  - In variable partitioning, partitions are not made before the execution or during system configuration.
  - Initially RAM is empty and partitions are made during the run-time according to process’s need.
  - 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
  - This technique does not suffer from internal fragmentation but do suffer from external fragmentation

Memory Allocation Techniques

- **Variable Partitioning (Dynamic Partitioning)**



Memory Allocation Techniques

- **Compaction**
  - Compaction is a technique to collect all the free memory present in form of fragments into one large chunk of free memory, which can be used to run other processes.
  - It does that by moving all the processes towards one end of the memory and all the available free space towards the other end of the memory so that it becomes contiguous.
  - Compaction is used to solve the problem of external fragmentation.
  - It is not easy to do compaction as a huge amount of time is wasted in performing compaction and CPU sits idle for a long time thus reducing the system efficiency.

Memory Allocation Algorithms

- **First Fit:**The first fit algorithm allocates memory to a process based on the first available block of memory that is large enough to accommodate the process.his is the simplest algorithm among the three, as it requires only a linear search through the free memory list.
- **Best Fit:**The best fit algorithm allocates memory to a process based on the smallest block of memory that can accommodate the process. This algorithm searches the entire free memory list for the smallest block of memory that can accommodate the process, and allocates memory from that block

Memory Allocation Algorithms

- **Worst Fit:** The worst fit algorithm allocates memory to a process based on the largest block of memory available. This algorithm searches the entire free memory list for the largest block, and allocates memory from that block.

Memory Allocation Algorithms Example

Example 1: Request from processes are as follows **P1**: 300KB, **P2**: 25KB, **P3**: 125KB and **P4**: 50KB. How will the processes allocated memory using First Fit, Best Fit and Worst Fit? [**Note**: Consider dynamic partitioning]

Memory Allocation Algorithms Examples

- **Example 1**

First Fit

Best Fit

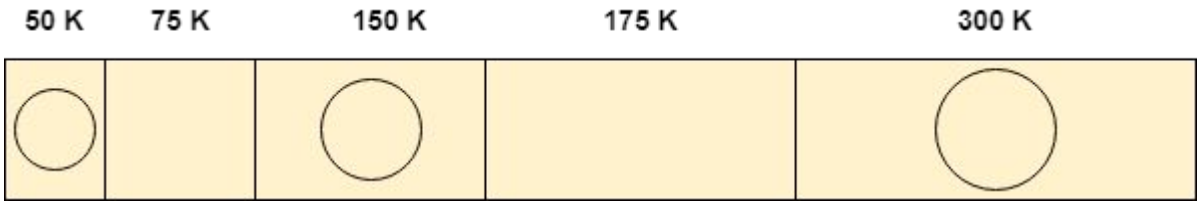
Worst Fit

Memory Allocation Algorithms Question

Process requests are given as;

P1: 25 K, **P2**: 50 K, **P3**: 100 K, **P4**: 75 K

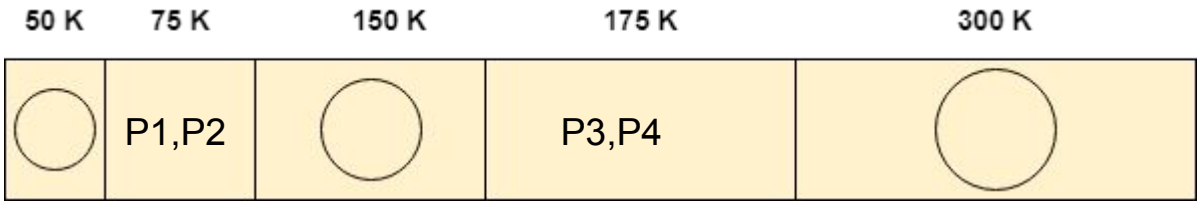
How will the processes be allocated memory using First Fit, Best Fit and Worst Fit? [**Note**: Consider dynamic partitioning].



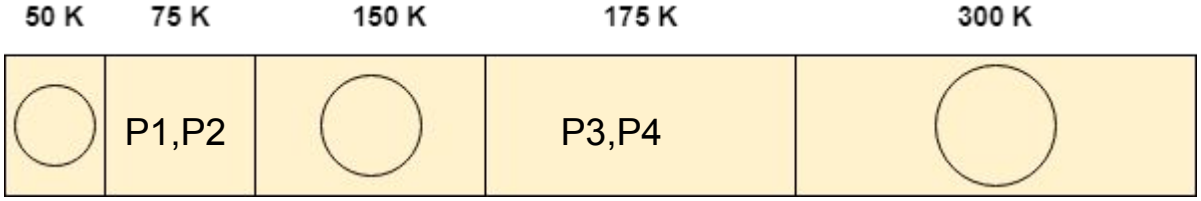
Solution

- **Example 1**

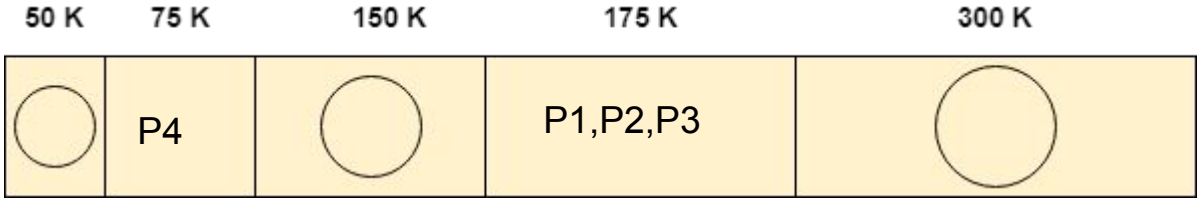
First Fit



Best Fit



Worst Fit



Memory Allocation Techniques

- **Non Contiguous Allocation**
  - Non-contiguous memory allocation is a memory allocation technique used in computer operating systems where memory is not allocated in contiguous blocks. In this technique, memory is allocated to a process in a scattered or non-contiguous manner across the memory space.
  - In simple terms, process is spanned into smaller sections and then allocated memory blocks.
  - Non-contiguous memory allocation can be achieved using techniques such as paging or segmentation

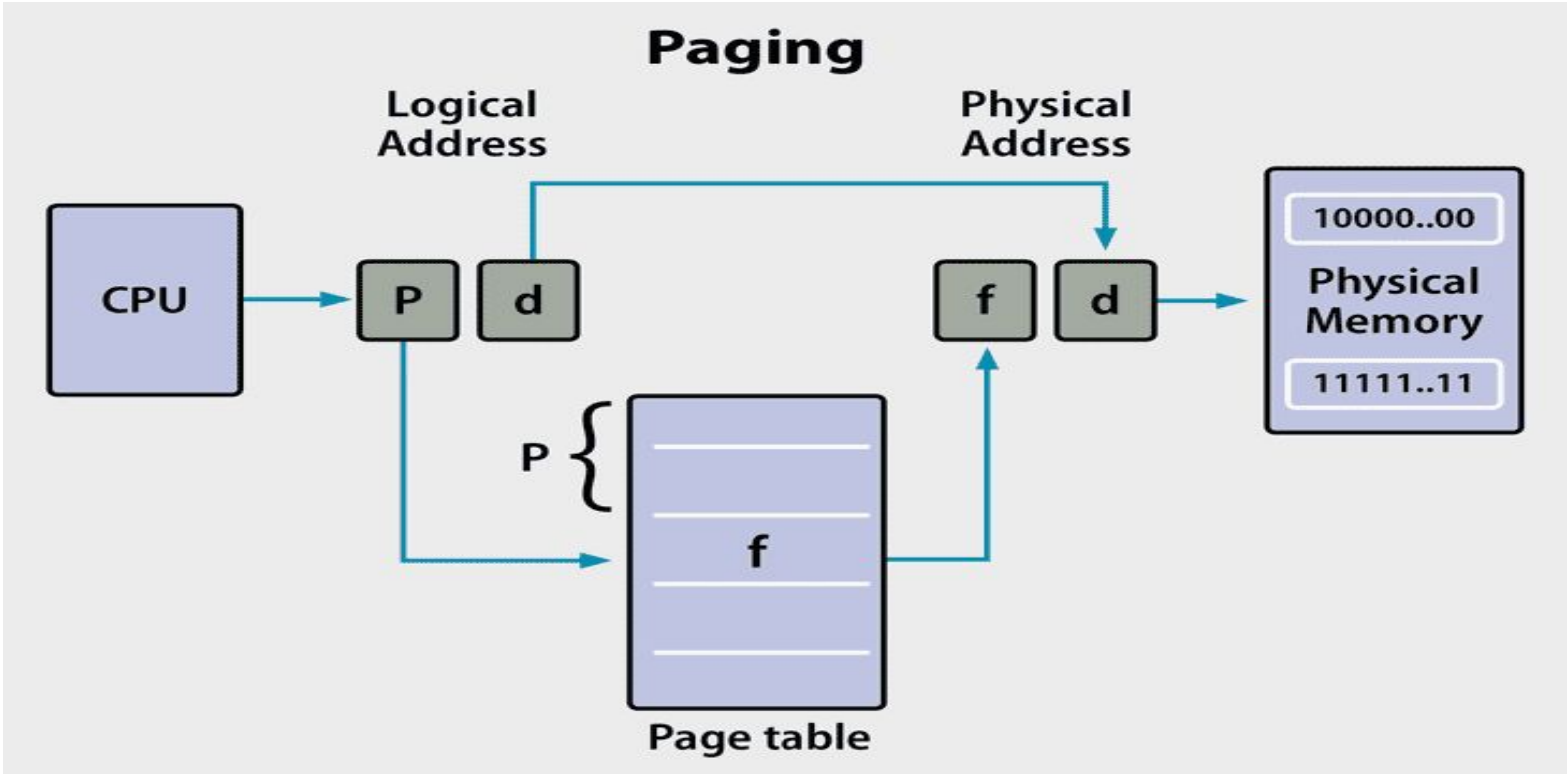
Memory Allocation Techniques

- **Paging**
  - Paging is a memory management technique used in computer operating systems to manage memory allocation(non contagious)
  - In paging, the memory is divided into fixed-size blocks called pages, and a process is divided into fixed-size blocks called frames.
  - Size of page = Size of frame
  - Each frame of a process is assigned to a page of memory, and the pages can be scattered across the memory space.

Memory Allocation Techniques

- **Paging**
  - Address generated by CPU is called Logical address
  - Memory Management Unit converts logical address into physical address, which is the actual location of data on the main memory.

Memory Allocation Techniques (Paging)

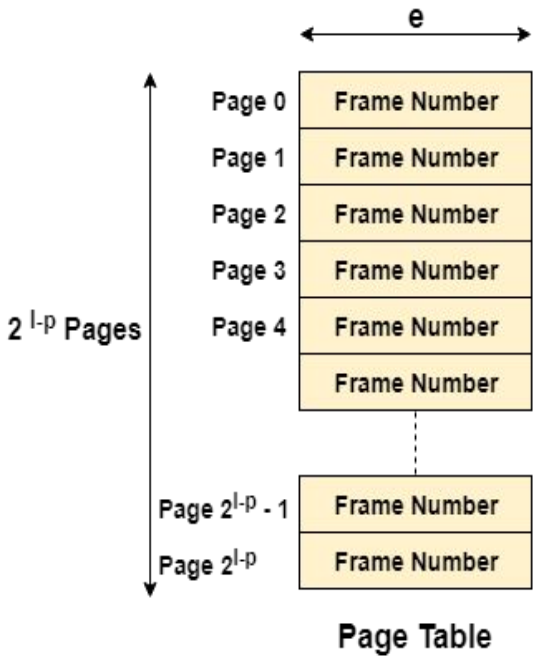


Paging

- **Page Table**
  - A page table is a data structure used by the operating system to keep track of the mapping between logical addresses generated by CPU and physical memory addresses in the computer's RAM.
  - The page table typically contains an entry for each page of the process. Each entry in the page table contains information about the corresponding page of physical memory, such as its location in RAM, whether it is currently in use, and whether it has been modified since it was last loaded into memory.

Paging

- **Page Table**



Paging

- **Page Table Entry**

Paging

- **Page Table Entry**
  - **Valid/Invalid Bit**: A flag that indicates whether the corresponding page is currently in physical memory. If the flag is set, the page is in memory and can be accessed; if the flag is not set, a page fault occurs when the page is accessed.
  - **Frame Number**: The physical address of the page in memory that corresponds to the logical address in the entry.
  - **Protection Bits**: Flags that indicate the permissions for the corresponding page, such as whether the page can be read, written, or executed.

Paging

- **Page Table Entry**
  - **Dirty Bit**: A flag that indicates whether the corresponding page has been modified since it was loaded into memory. This flag is used by the operating system to determine whether the page needs to be written back to disk when it is evicted from memory
  - **Reference Bit**: A flag that is set by the hardware whenever the corresponding page is accessed. This flag is used by the operating system to implement page replacement algorithms, which use the reference bit to determine which pages are frequently used and should be kept in memory

Paging

- **Translation Lookaside Buffer**
  - A Translation Lookaside Buffer (TLB) is a hardware cache used by the CPU to improve the performance of virtual memory address translation.
  - The TLB stores recently used virtual-to-physical address mappings, so that the CPU can quickly access the physical memory location corresponding to a given virtual memory address, without having to perform a time-consuming page table walk each time.

Paging

- **Translation Lookaside Buffer**
  - When a program running on a computer system accesses a virtual memory address, the CPU first consults the TLB to see if the corresponding physical address is already stored in the cache. If the address is found in the TLB, the CPU can immediately retrieve the data from physical memory.
  - If the address is not found in the TLB, the CPU must perform a page table walk to look up the corresponding physical address, and store the result in the TLB for future reference.

Memory Allocation Techniques

- **Segmentation**
  - Segmentation is a memory management technique used by operating systems to manage virtual memory.
  - Unlike paging, where memory is divided into fixed-size pages, segmentation divides memory into variable-sized segments that correspond to logical units of a program, such as functions, arrays, and data structures.
  - Each segment is identified by a segment number or a segment name, and contains a contiguous block of memory addresses that can be used to store data or code.

Memory Allocation Techniques

- **Segmentation**
  - When a program running on a computer system needs to access a particular segment, it uses the segment number or name to retrieve the starting address of the segment from a segment table stored in memory
  - The segment table contains information about each segment, such as its starting address, size, and access permissions.

Segmentation

- **Segment Table**

Virtual Memory

- Virtual memory is a computer system's ability to use secondary storage (such as hard disk drives) to extend the available memory beyond the physical RAM installed in the system.
- It is a memory management technique that allows a computer to run more applications or processes than it could with just the physical memory
- In virtual memory, the operating system divides the virtual address space (the range of memory addresses used by a program) into smaller units called pages.
- Each page is a fixed size and corresponds to a block of physical memory or storage on disk

Virtual Memory

- When a program running on a computer system tries to access a page that is not currently loaded in physical memory, the operating system retrieves the required page from disk and swaps out another page that is currently in physical memory.
- Virtual memory is an important component of modern operating systems, as it allows programs to run efficiently and effectively, even when the physical memory is limited.

Demand Paging

- Demand Paging is a technique in which a page is usually brought into the main memory only when it is needed or demanded by the CPU. Initially, only those pages are loaded that are required by the process immediately. Those pages that are never accessed are thus never loaded into the physical memory.
- Advantages of Demand Paging:
  - Memory can be put to better use
  - By using demand paging, we can run programs that are larger than physical memory

Virtual Memory

- **Page Fault**
  - A page fault is a type of exception that occurs when a program running on a computer system attempts to access a page of memory that is not currently loaded in physical memory (RAM).
  - A page fault indicates that the required page must be loaded from secondary storage (such as a hard disk) into physical memory before the program can continue executing.
  - When a page fault occurs, the operating system's memory manager is responsible for resolving the exception

Page Replacement Algorithms

1. **FIFO (First In First Out)**

   This algorithm removes the oldest page in memory first. It works by maintaining a queue of pages, with the oldest page at the front of the queue. When a new page needs to be loaded and memory is full, the page at the front of the queue is removed and the new page is loaded into memory.

2. **LRU (Least Recently Used)**

   This algorithm removes the least recently used page from memory first. It works by maintaining a list of pages and updating the list each time a page is accessed. When a new page needs to be loaded and memory is full, the page at the end of the list (i.e., the least recently used page) is removed and the new page is loaded into memory.

Page Replacement Algorithms

3. **Optimal Page Replacement**

   This algorithm removes the page that will not be used for the longest period of time in the future. It works by simulating all possible page replacements and choosing the page that results in the longest time until the next reference to the page. While this algorithm is theoretically optimal, it requires knowledge of future page references, which is generally not available in practice.

4. **MRU (Most Recently Used)**

   The MRU algorithm works by maintaining a list of pages and updating the list each time a page is accessed. When a new page needs to be loaded and memory is full, the page that was most recently accessed (i.e., the page at the front of the list) is removed and the new page is loaded into memory.

FIFO Question 1

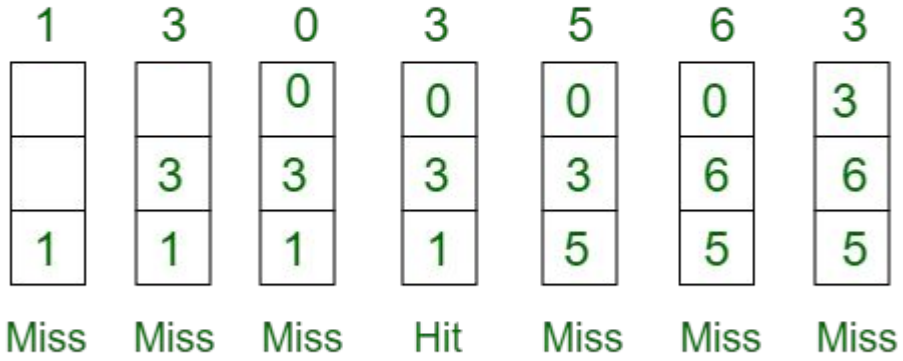
A system uses 3 page frames for storing process pages in main memory. It uses the FIFO page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

1,3,0,3,5,6,3

FIFO Question 1 Solution

A system uses 3 page frames for storing process pages in main memory. It uses the FIFO page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

1,3,0,3,5,6,3



Total Page Fault = 6

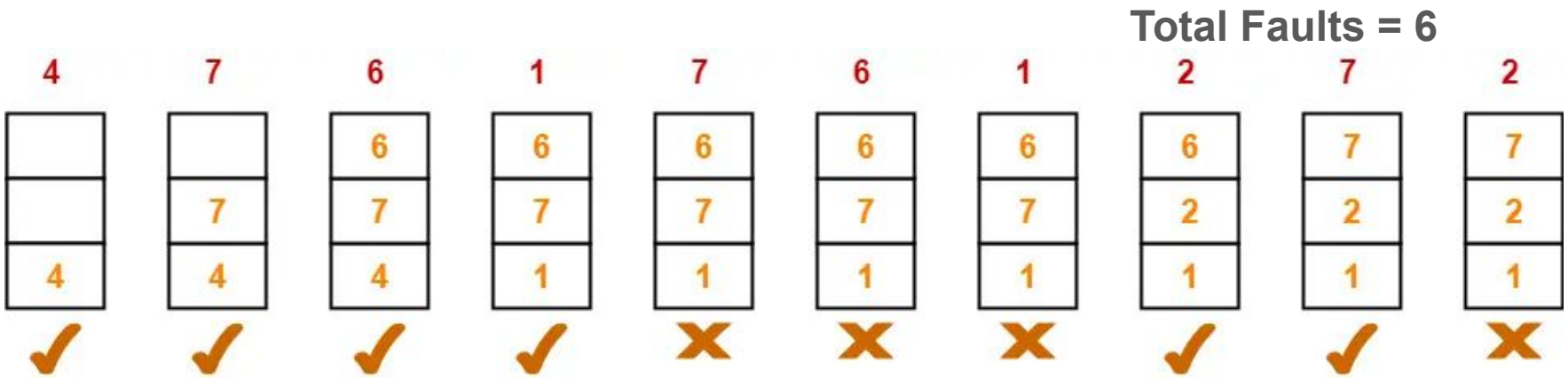
FIFO Question 2

A system uses 3 page frames for storing process pages in main memory. It uses the FIFO page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

4,7,6,1,7,6,1,2,7,2

FIFO Question 2 Solution

A system uses 3 page frames for storing process pages in main memory. It uses the FIFO page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below- **4,7,6,1,7,6,1,2,7,2**



LRU Question 1

A system uses 3 page frames for storing process pages in main memory. It uses the Least Recently Used (LRU) page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

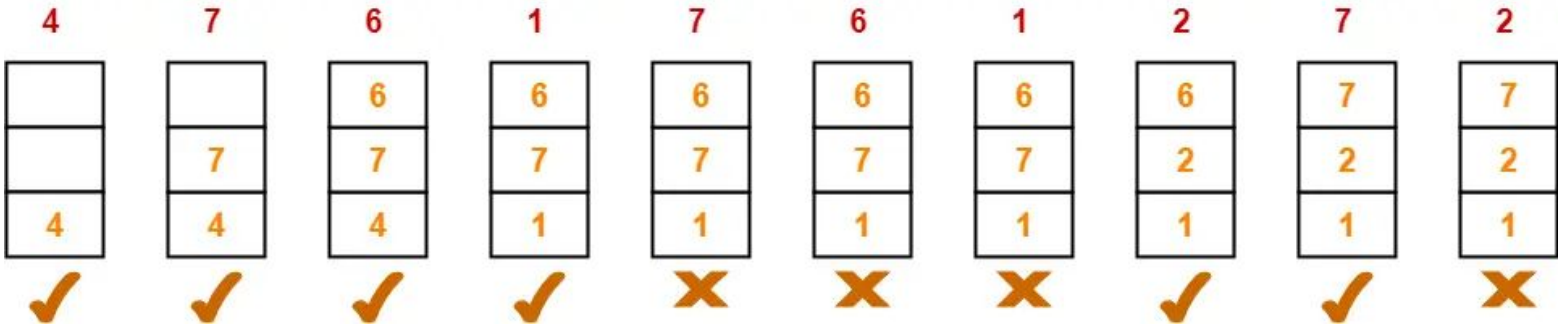
4 ,7, 6, 1, 7, 6, 1, 2, 7, 2

LRU Question 1 Solution

A system uses 3 page frames for storing process pages in main memory. It uses the Least Recently Used (LRU) page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

4 ,7, 6, 1, 7, 6, 1, 2, 7, 2

Total faults = 6



LRU Question 2

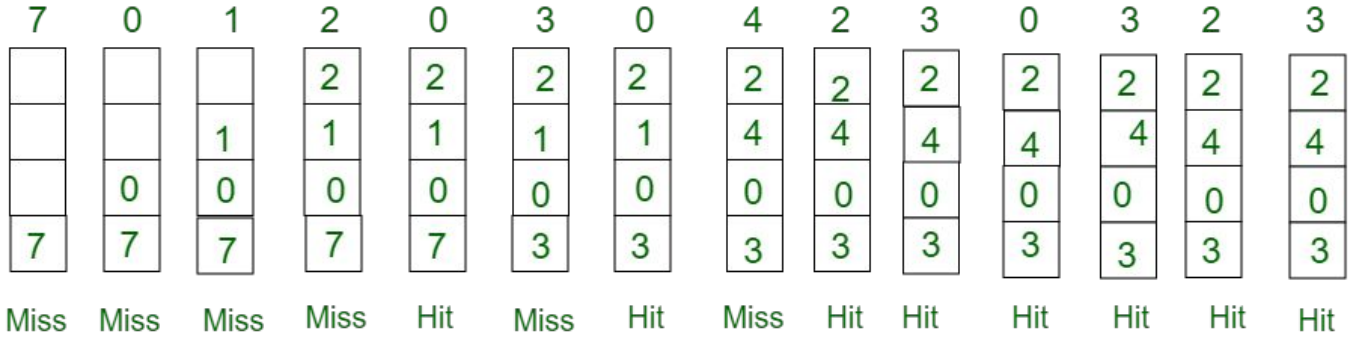
A system uses 4 page frames for storing process pages in main memory. It uses the Least Recently Used (LRU) page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

7,0,1,2,0,3,0,4,2,3,0,3,2,3

LRU Question 2 Solution

A system uses 4 page frames for storing process pages in main memory. It uses the Least Recently Used (LRU) page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

7,0,1,2,0,3,0,4,2,3,0,3,2,3



Total Page Fault = 6

Optimal Question 1

A system uses 3 page frames for storing process pages in main memory. It uses the Optimal page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

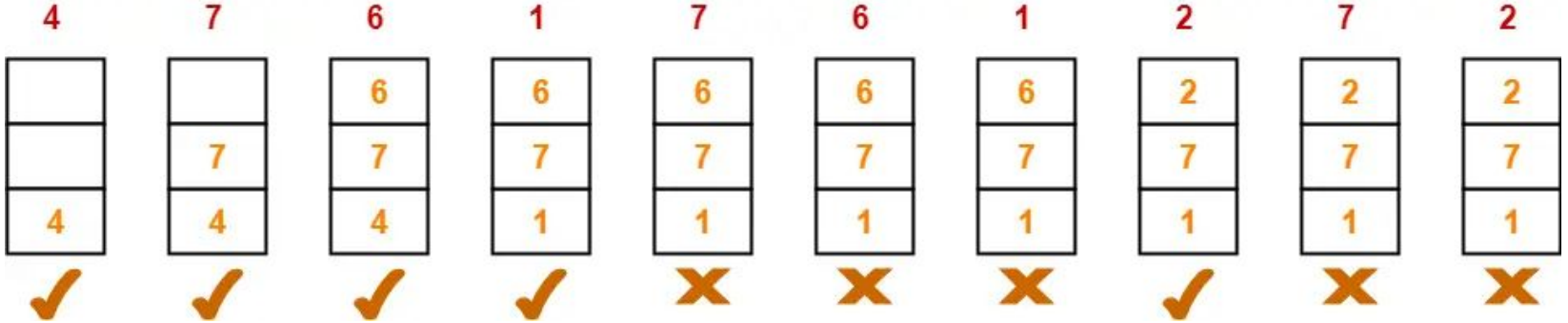
4 , 7, 6, 1, 7, 6, 1, 2, 7, 2

Optimal Question 1 Solution

A system uses 3 page frames for storing process pages in main memory. It uses the Optimal page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

4 , 7, 6, 1, 7, 6, 1, 2, 7, 2

Total faults = 5



Optimal Question 2

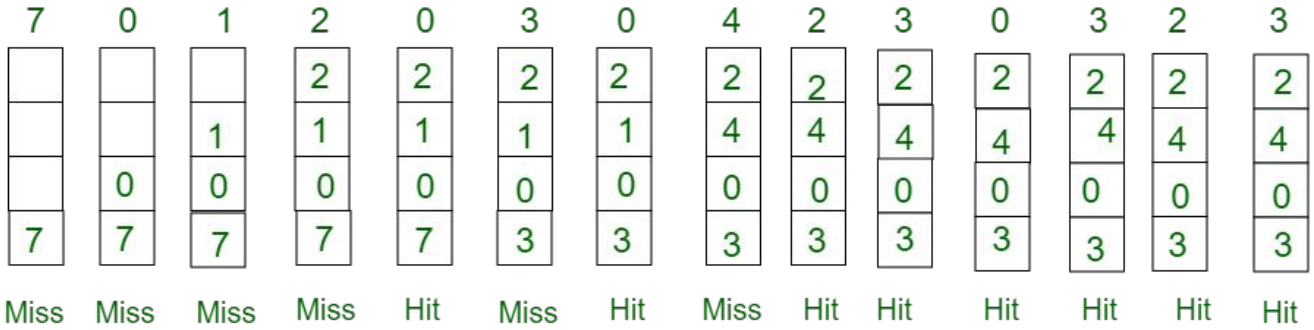
A system uses 4 page frames for storing process pages in main memory. It uses the Optimal page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

7,0,1,2,0,3,0,4,2,3,0,3,2,3

Optimal Question 2 Solution

A system uses 4 page frames for storing process pages in main memory. It uses the Optimal page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

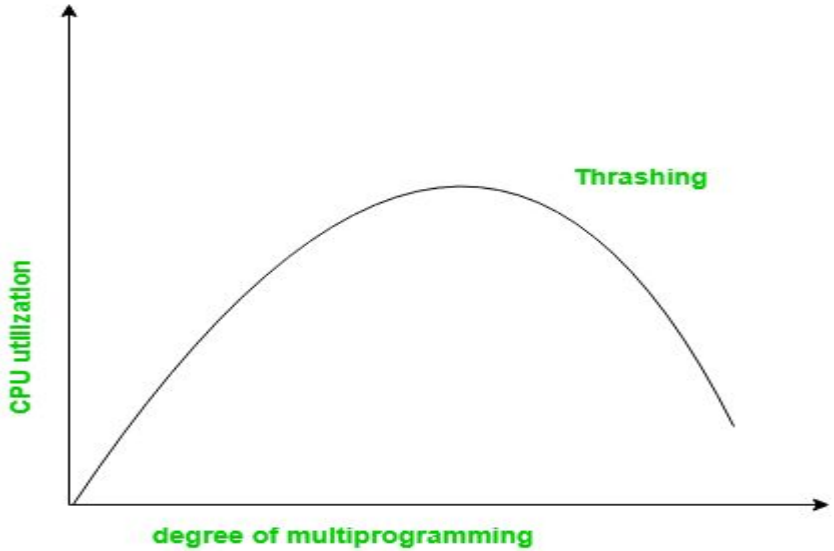
7,0,1,2,0,3,0,4,2,3,0,3,2,3



Total Page Fault = 6

Thrashing

- Thrashing is a condition or a situation when the system is spending a major portion of its time servicing the page faults, but the actual processing done is very negligible. As a result CPU utilization is going to be reduced or negligible



THANK YOU