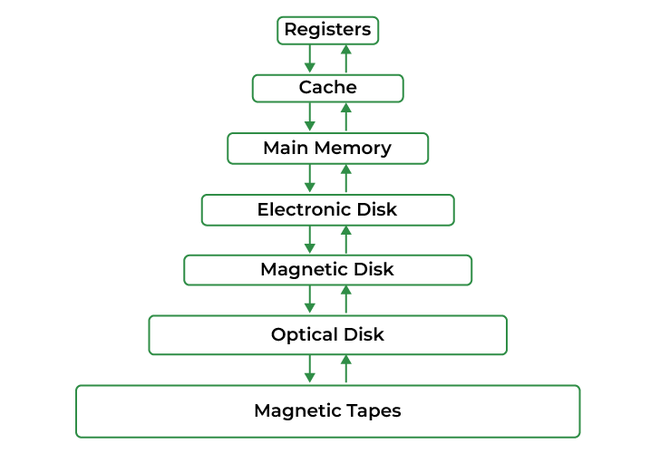
**UNIT-4:**

1. **Memory management**

* The term memory can be defined as a collection of data in a specific format. It is used to store instructions and process data.
* The memory comprises a large array or group of words or bytes, each with its own location.
* To achieve a degree of multiprogramming and proper utilization of memory, memory management is important. Many memory management methods exist, reflecting various approaches, and the effectiveness of each algorithm depends on the situation.
* Main memory is the place where programs and information are kept when the processor is effectively utilizing them.
* Main memory is also known as [RAM (Random Access Memory)](https://www.geeksforgeeks.org/random-access-memory-ram/). This memory is volatile. RAM loses its data when a power interruption occurs.



### Registers

Registers are small, high-speed memory units located in the CPU. They are used to store the most frequently used data and instructions. [Registers](https://www.geeksforgeeks.org/different-classes-of-cpu-registers/) have the fastest access time and the smallest storage capacity, typically ranging from 16 to 64 bits.

### Cache Memory

Cache memory is a small, fast memory unit located close to the CPU. It stores frequently used data and instructions that have been recently accessed from the main memory. [Cache memory](https://www.geeksforgeeks.org/cache-memory-in-computer-organization/) is designed to minimize the time it takes to access data by providing the CPU with quick access to frequently used data.

### ****Main Memory****

Main memory, also known as [RAM](https://www.geeksforgeeks.org/random-access-memory-ram/) (Random Access Memory), is the primary memory of a computer system. It has a larger storage capacity than cache memory, but it is slower. [Main memory](https://www.geeksforgeeks.org/memory-management-in-operating-system/) is used to store data and instructions that are currently in use by the CPU.

Types of Main Memory

* **Static RAM:** Static RAM stores the binary information in flip flops and information remains valid until power is supplied. [Static RAM](https://www.geeksforgeeks.org/difference-between-sram-and-dram/) has a faster access time and is used in implementing cache memory.
* **Dynamic RAM:** It stores the binary information as a charge on the capacitor. It requires refreshing circuitry to maintain the charge on the capacitors after a few milliseconds. It contains more memory cells per unit area as compared to SRAM.

### Secondary Storage

Secondary storage, such as [hard disk drives (HDD) and solid-state drives (SSD)](https://www.geeksforgeeks.org/difference-between-hard-disk-drive-hdd-and-solid-state-drive-ssd/), is a non-volatile memory unit that has a larger storage capacity than main memory. It is used to store data and instructions that are not currently in use by the CPU. Secondary storage has the slowest access time and is typically the least expensive type of memory in the memory hierarchy.

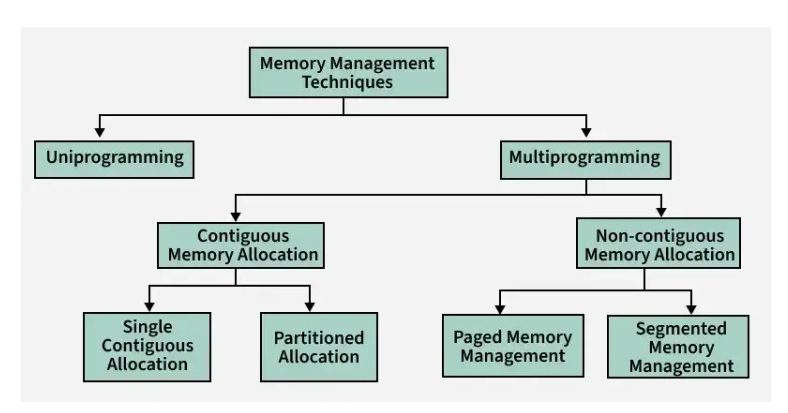
### Magnetic Disk

Magnetic Disks are simply circular plates that are fabricated with either a metal or a plastic or a magnetized material. The [Magnetic disks](https://www.geeksforgeeks.org/magnetic-disk-memory/) work at a high speed inside the computer and these are frequently used.

### Magnetic Tape

Magnetic Tape is simply a magnetic recording device that is covered with a plastic film. [Magnetic Tape](https://www.geeksforgeeks.org/magnetic-tape-memory/) is generally used for the backup of data. In the case of a magnetic tape, the access time for a computer is a little slower and therefore, it requires some amount of time for accessing the strip.

* The task of subdividing the memory among different processes is called Memory Management. Memory management is a method in the operating system to manage operations between main memory and disk during process execution. The main aim of memory management is to achieve efficient utilization of memory.
* **Why Memory Management is Required?**
* Allocate and de-allocate memory before and after process execution.
* To keep track of used memory space by processes.
* To minimize [fragmentation](https://www.geeksforgeeks.org/difference-between-internal-and-external-fragmentation/)issues.
* To proper utilization of main memory.
* To maintain data integrity while executing of process.
* **Logical and Physical Address Space**
* Logical Address Space: An address generated by the CPU is known as a “Logical Address”. It is also known as a Virtual address. Logical address space can be defined as the size of the process. A logical address can be changed.
* Physical Address Space: An address seen by the memory unit (i.e. the one loaded into the memory address register of the memory) is commonly known as a “Physical Address”. A Physical address is also known as a Real address. The set of all physical addresses corresponding to these logical addresses is known as Physical address space. A [physical address](https://www.geeksforgeeks.org/logical-and-physical-address-in-operating-system/) is computed by MMU. The run-time mapping from virtual to physical addresses is done by a hardware device Memory Management Unit(MMU). The physical address always remains constant.
* **Static and Dynamic Loading**
* Loading a process into the main memory is done by a loader.
* There are two different types of loading :
  1. Static Loading: Static Loading is basically loading the entire program into a fixed address. It requires more memory space.
  2. Dynamic Loading: The entire program and all data of a process must be in physical memory for the process to execute. So, the size of a process is limited to the size of [physical memory.](https://www.geeksforgeeks.org/logical-and-physical-address-in-operating-system/)
* **Static and Dynamic Linking**
* Static Linking: In [static linking](https://www.geeksforgeeks.org/static-and-dynamic-linking-in-operating-systems/), the linker combines all necessary program modules into a single executable program. So there is no runtime dependency. Some operating systems support only static linking, in which system language libraries are treated like any other object module.
* Dynamic Linking: The basic concept of dynamic linking is similar to dynamic loading. In [dynamic linking](https://www.geeksforgeeks.org/static-and-dynamic-linking-in-operating-systems/), "Stub" is included for each appropriate library routine reference. A stub is a small piece of code. When the stub is executed, it checks whether the needed routine is already in memory or not. If not available then the program loads the routine into memory.
* **Memory Management Techniques**



* Two primary approaches:

## ****Uniprogramming****

* One program at a time.
* Only one process is loaded in memory at a time.
* Simple memory management.
* No sharing or protection is needed.
* Not efficient for CPU usage — the CPU may remain idle during I/O.

1. **Multiprogramming**

* Multiple programs in memory, managed concurrently.
* Multiple processes are loaded in memory to **increase CPU utilization.**
* Requires **process isolation**, protection, and scheduling.
* Memory can be allocated in two ways:

### **Contiguous Memory Allocation**

* Contiguous memory allocation is a memory allocation strategy. As the name implies, we utilize this technique to assign contiguous blocks of memory to each task. Thus, whenever a process asks to access the main memory, we allocate a continuous segment from the empty region to the process based on its size.
* In this technique, memory is allotted in a continuous way to the processes.
* **Contiguous Memory Management has two types:**
* **Fixed (or Static) Partition**
* **Variable (or Dynamic) Partitioning**

**Fixed Partition Scheme**

* In the [fixed partition scheme](https://www.geeksforgeeks.org/fixed-or-static-partitioning-in-operating-system), memory is divided into fixed number of partitions.
* Fixed means number of partitions are fixed in the memory. In the fixed partition, in every partition only one process will be accommodated.
* Degree of multi-programming is restricted by number of partitions in the memory. Maximum size of the process is restricted by maximum size of the partition. Every partition is associated with the limit registers.
* Disadvantages Fix partition scheme
* Maximum process size should always be less than equal to maximum partition size.
* The degree of multiprogramming is directly proportional to the number of partitions.
* If a process of 19kb wants to allocate and we have free space which is not continuous we are not able to allocate the space.
* [Internal Fragmentation](https://www.geeksforgeeks.org/difference-between-internal-and-external-fragmentation) is found in fixed partition scheme. To overcome the problem of internal fragmentation, instead of fixed partition scheme, variable partition scheme is used.

**Variable Partition Scheme**

* In the [variable partition scheme](https://www.geeksforgeeks.org/variable-or-dynamic-partitioning-in-operating-system), initially memory will be single continuous free block. Whenever the request by the process arrives, accordingly partition will be made in the memory. If the smaller processes keep on coming then the larger partitions will be made into smaller partitions.
* In variable partition schema initially, the memory will be full contiguous free block
* Memory divided into partitions according to the process size where process size will vary.
* One partition is allocated to each active partition.
* [External Fragmentation](https://www.geeksforgeeks.org/difference-between-internal-and-external-fragmentation) is found in variable partition scheme.
* Advantages of Variable Partition Scheme
* Portion size = process size
* There is no internal fragmentation (which is the drawback of fixed partition schema).
* Degree of [multiprogramming](https://www.geeksforgeeks.org/multiprogramming-in-operating-system) varies and is directly proportional to a number of processes.
* Disadvantage Variable Partition Scheme
* External fragmentation is still there.
* External Fragmentation in Variable Partition Scheme
* Solution of External Fragmentation
* To overcome the problem of external fragmentation, following techniques are used :

Compaction

Moving all the processes toward the top or towards the bottom to make free available memory in a single continuous place is called compaction. [Compaction](https://www.geeksforgeeks.org/compaction-in-operating-system) is undesirable to implement because it interrupts all the running processes in the memory. It also consumes CPU time (overhead).

### 🔸 B. **Non-Contiguous Memory Allocation**

* Non-contiguous allocation, also known as dynamic or linked allocation, is a memory allocation technique used in operating systems to allocate memory to processes that do not require a contiguous block of memory.
* In this technique, each process is allocated a series of non-contiguous blocks of memory that can be located anywhere in the physical memory.
* Non-contiguous allocation involves the use of pointers to link the non-contiguous memory blocks allocated to a process. These pointers are used by the operating system to keep track of the memory blocks allocated to the process and to locate them during the execution of the process.
* [Paging](https://www.geeksforgeeks.org/operating-system-paging/) and [Segmentation](https://www.geeksforgeeks.org/operating-systems-segmentation/) are the two ways that allow a process’s physical address space to be non-contiguous.
  1. **Paging:**
* In paging, each process consists of fixed-size components called pages.
* The size of a page is defined by the hardware of a computer, and the demarcation of pages is implicit in it. The memory can accommodate an integral number of pages. It is partitioned into memory areas that have the same size as a page, and each of these memory areas is considered separately for allocation to a page. This way, any free memory area is exactly the same size as a page, so [external fragmentation](https://www.geeksforgeeks.org/external-fragmentation-in-os/) does not arise in the system. Internal fragmentation can arise because the last page of a process is allocated a page-size memory area even if it is smaller than a page in size.
* Why Paging is Important?

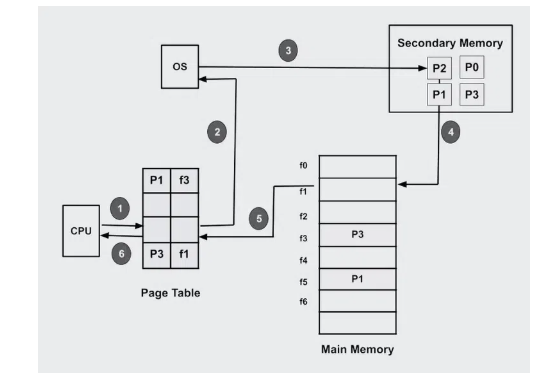
In paging, the Operating system needs to maintain the table which is called the [Page Table](https://www.geeksforgeeks.org/page-table-entries-in-page-table/) for each process which contains the base address of each block that is acquired by the process in memory space. In non-contiguous memory allocation, different parts of a process are allocated to different places in Main Memory. Spanning is allowed which is not possible in other techniques like Dynamic or Static Contiguous memory allocation. That's why paging is needed to ensure effective memory allocation. Paging is done to remove External Fragmentation.

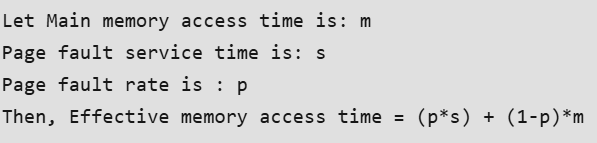
* How does paging work?
  1. **Segmentation**
* In segmentation, a programmer identifies components called segments in a process.
* A segment is a logical entity in a program, e.g., a set of functions, data structures, or objects. Segmentation facilitates the sharing of code, data, and program modules processes. However, segments have different sizes, so the [kernel](https://www.geeksforgeeks.org/kernel-in-operating-system/) has to use memory reuse techniques such as first-fit or best-fit allocation. Consequently, external fragmentation can arise.
* How Does segmentation Work?

1. **Virtual memory concepts:**

* 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.
* The main objective of virtual memory is to support multiprogramming.
* The main advantage that virtual memory provides is, a running process does not need to be entirely in memory. Programs can be larger than the available physical memory. Virtual Memory provides an abstraction of main memory, eliminating concerns about storage limitations.
* A memory hierarchy, consisting of a computer system's memory and a disk, enables a process to operate with only some portions of its address space in RAM to allow more processes to be in memory.
* **How Virtual Memory Works?**
* Virtual memory is implemented using Demand Paging or Demand Segmentation.
* **Types of Virtual Memory**
* **There are two main types of virtual memory:**
  1. **Paging**
  2. **Segmentation**

**Paging**

* Paging divides memory into small fixed-size blocks called pages. When the computer runs out of RAM, pages that aren't currently in use are moved to the hard drive, into an area called a swap file.
* The swap file acts as an extension of RAM. When a page is needed again, it is swapped back into RAM, a process known as page swapping. This ensures that the operating system (OS) and applications have enough memory to run.
* Demand Paging:
* The process of loading the page into memory on demand (whenever a page fault occurs) is known as [demand paging](https://www.geeksforgeeks.org/what-is-demand-paging-in-operating-system/).
* The term "page miss" or "[page fault](https://www.geeksforgeeks.org/page-fault-handling-in-operating-system/)" refers to a situation where a referenced page is not found in the main memory.
* When a program tries to access a page, or fixed-size block of memory, that isn't currently loaded in physical memory (RAM), an exception known as a page fault happens. Before enabling the program to access a page that is required, the operating system must bring it into memory from secondary storage (such a hard drive) in order to handle a page fault.
* Common Algorithms Used for Demand Paging in OS
  1. FIFO (First-In-First-Out):
  2. LRU (Least Recently Used):
  3. LFU (Least Frequently Used):
  4. MRU (Most Recently Used):
* The process includes the following steps are as follows:
* If the CPU tries to refer to a page that is currently not available in the main memory, it generates an interrupt indicating a memory access fault.
* The OS puts the interrupted process in a blocking state. For the execution to proceed the OS must bring the required page into the memory.
* The OS will search for the required page in the logical address space.
* The required page will be brought from logical address space to physical address space. The page replacement algorithms are used for the decision-making of replacing the page in physical address space.
* The page table will be updated accordingly.
* The signal will be sent to the CPU to continue the program execution and it will place the process back into the ready state.
* Hence whenever a page fault occurs these steps are followed by the operating system and the required page is brought into memory.
* 
* **What is Page Fault Service Time?**
* The time taken to service the page fault is called page fault service time.
* The page fault service time includes the time taken to perform all the above six steps.



**Advantages of Demand Paging**

* **Efficient use of physical memory**: Query paging allows for more efficient use because only the necessary pages are loaded into memory at any given time.
* **Support for larger programs:** Programs can be larger than the physical memory available on the system because only the necessary pages will be loaded into memory.
* **Faster program start:** Because only part of a program is initially loaded into memory, programs can start faster than if the entire program were loaded at once.
* **Reduce memory usage:**Query paging can help reduce the amount of memory a program needs, which can improve system performance by reducing the amount of disk I/O required.

**Disadvantages of Demand Paging**

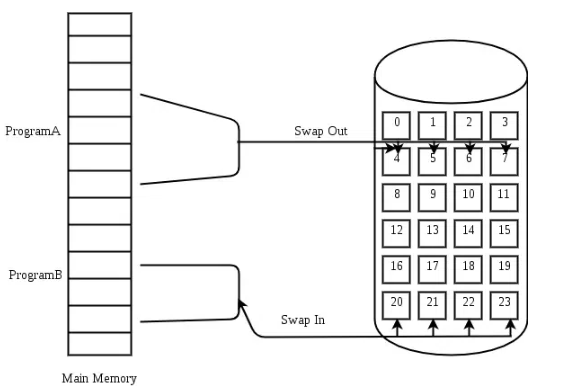
* **Page Fault Overload:**The process of [swapping](https://www.geeksforgeeks.org/swapping-in-operating-system/) pages between memory and disk can cause a performance overhead, especially if the program frequently accesses pages that are not currently in memory.
* **Degraded Performance:**If a program frequently accesses pages that are not currently in memory, the system spends a lot of time swapping out pages, which degrades performance.
* **Fragmentation:**Query paging can cause physical memory [fragmentation](https://www.geeksforgeeks.org/what-is-fragmentation-in-operating-system/), degrading system performance over time.
* **Complexity:** Implementing query paging in an operating system can be complex, requiring complex algorithms and [data structures](https://www.geeksforgeeks.org/introduction-to-data-structures/) to manage page tables and swap space.

**Segmentation**

* Segmentation divides virtual memory into segments of different sizes.
* Segments that aren't currently needed can be moved to the hard drive. The system uses a segment table to keep track of each segment's status, including whether it's in memory, if it's been modified, and its physical address.
* Segments are mapped into a process's address space only when needed.
* Numericals:

**Swapping**

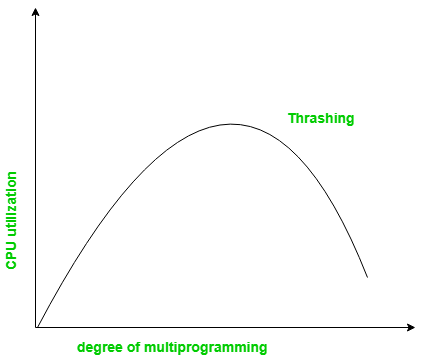
* Swapping is a process out means removing all of its pages from memory, or marking them so that they will be removed by the normal page replacement process.
* Suspending a process ensures that it is not runnable while it is swapped out. At some later time, the system swaps back the process from the secondary storage to the main memory. When a process is busy [swapping](https://www.geeksforgeeks.org/swapping-in-operating-system/) pages in and out then this situation is called thrashing.



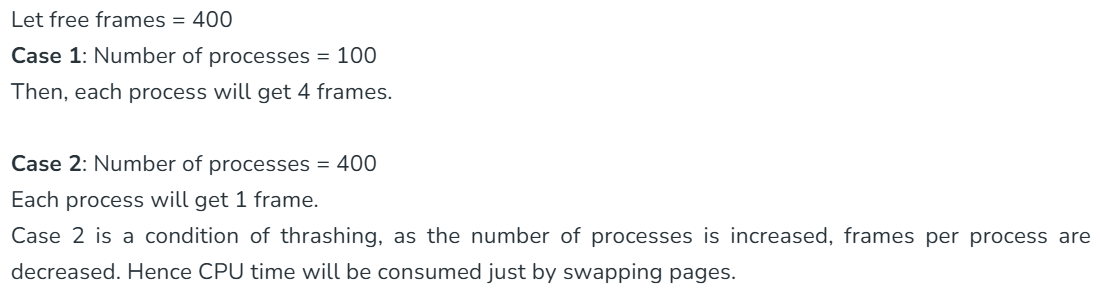
* Numericals:

**Thrashing:**

* When a process is busy [swapping](https://www.geeksforgeeks.org/swapping-in-operating-system/) pages in and out then this situation is called thrashing.
* **Thrashing** is a process that occurs when the system spends a major portion of time transferring shared data block blocks from one node to another in comparison with the time spent on doing the useful work of executing the application process. If thrashing is not handled carefully it degrades system performance considerably.



* Causes of Thrashing
  1. **High Degree of Multiprogramming**
* If the number of processes keeps on increasing in the memory then the number of frames allocated to each process will be decreased. So, fewer frames will be available for each process. Due to this, a [page fault](https://www.geeksforgeeks.org/page-fault-handling-in-operating-system/) will occur more frequently and more CPU time will be wasted in just swapping in and out of pages and the utilization will keep on decreasing.
* For example:



### ****Lacks of Frames****

* If a process has fewer frames then fewer pages of that process will be able to reside in memory and hence more frequent swapping in and out will be required. This may lead to thrashing.
* Hence a sufficient amount of frames must be allocated to each process in order to prevent thrashing.
* Recovery of Thrashing:
* Do not allow the system to go into thrashing by instructing the long-term scheduler not to bring the processes into memory after the threshold.
* If the system is already thrashing then instruct the[mid-term scheduler](https://www.geeksforgeeks.org/medium-term-scheduler-in-operating-system/) to suspend some of the processes so that we can recover the system from thrashing.
* Performance in Virtual Memory
* Let p be the page fault rate( 0 <= p <= 1).
* if p = 0 no page faults
* if p =1, every reference is a fault.

Effective access time (EAT) = (1-p)\* Memory Access Time + p \* Page fault time.  
Page fault time = page fault overhead + swap out + swap in +restart overhead

The performance of a virtual memory management system depends on the total number of page faults, which depend on "[paging policies](https://www.geeksforgeeks.org/paging-in-operating-system/)" and "[frame allocation](https://www.geeksforgeeks.org/operating-system-allocation-frames/)"

* Frame Allocation

    A number of frames allocated to each process in either static or dynamic.

* Static Allocation: The number of frame allocations to a process is fixed.
* Dynamic Allocation:  The number of frames allocated to a process changes.
* Paging Policies
* Fetch Policy: It decides when a page should be loaded into memory.
* Replacement Policy: It decides which page in memory should be replaced.
* Placement Policy: It decides where in memory should a page be loaded.
* Applications of Virtual memory:
* Increased Effective Memory: One major practical application of virtual memory is, virtual memory enables a computer to have more memory than the physical memory using the disk space. This allows for the running of larger applications and numerous programs at one time while not necessarily needing an equivalent amount of [DRAM](https://www.geeksforgeeks.org/dram-full-form/).
* Memory Isolation: Virtual memory allocates a unique address space to each process and that also plays a role in process segmentation. Such separation increases safety and reliability based on the fact that one process cannot interact with and or modify another’s memory space through a mistake, or even a deliberate act of vandalism.
* Efficient Memory Management: Virtual memory also helps in better utilization of the physical memories through methods that include paging and segmentation. It can transfer some of the memory pages that are not frequently used to disk allowing RAM to be used by active processes when required in a way that assists in efficient use of memory as well as system performance.
* Simplified Program Development: For case of programmers, they don’t have to consider physical memory available in a system in case of having virtual memory. They can program ‘as if’ there is one big block of memory and this makes the programming easier and more efficient in delivering more complex applications.
* How to Manage Virtual Memory?

1. Adjust the Page File Size

* Automatic Management: All contemporary operating systems including Windows contain the auto-configuration option for the size of the empirical page file. But depending on the size of the RAM, they are set automatically, although the user can manually adjust the page file size if required.
* Manual Configuration: For tuned up users, the setting of the custom size can sometimes boost up the performance of the system. The initial size is usually advised to be set to the minimum value of 1. To set the size of the swap space equal to 5 times the amount of physical RAM and the maximum size 3 times the physical RAM.

1. Place the Page File on a Fast Drive

* SSD Placement: If this is feasible, the page file should be stored in the SSD instead of the [HDD](https://www.geeksforgeeks.org/hard-disk-drive-hdd-secondary-memory/) as a storage device. It has better read and write times, and the virtual memory may prove beneficial in an SSD.
* Separate Drive: Regarding systems having multiple drives involved, the page file needs to be placed on a different drive than the OS and that shall in turn improve its performance.

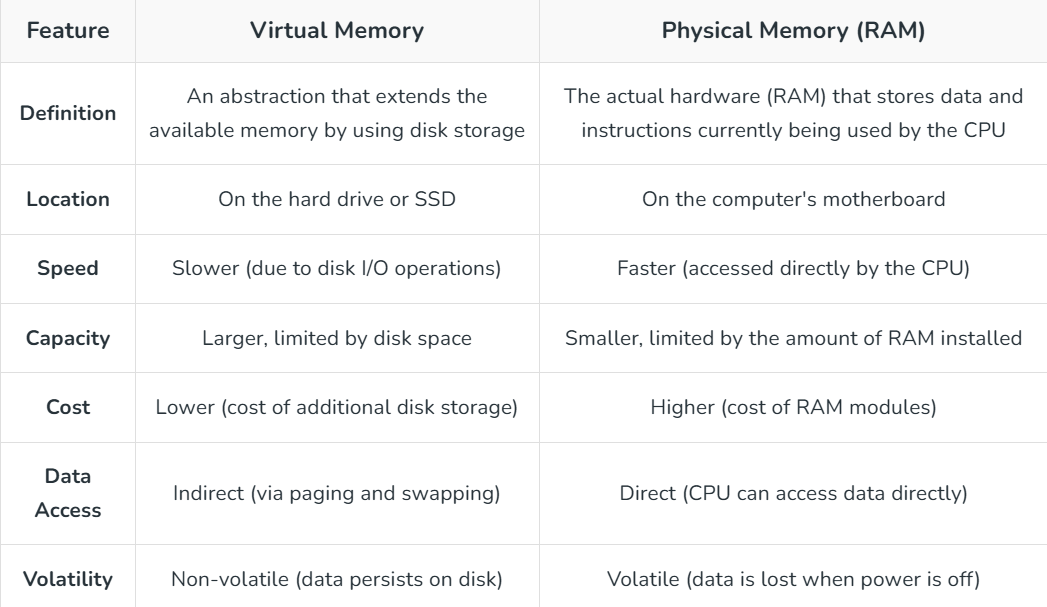
1. Monitor and Optimize Usage

* Performance Monitoring: Employ the software tools used in monitoring the performance of the system in tracking the amounts of virtual memory. High page file usage may signify that there is a lack of physical RAM or that virtual memory needs a change of settings or addition in physical RAM.
* Regular Maintenance: Make sure there is no toolbar or other application running in the background, take time and uninstall all the tool bars to free virtual memory.

1. Disable Virtual Memory for SSD

* Sufficient RAM: If for instance your system has a big physical memory, for example 16GB and above then it would be advised to freeze the page file in order to minimize [SSD](https://www.geeksforgeeks.org/introduction-to-solid-state-drive-ssd/) usage. But it should be done, in my opinion, carefully and only if the additional signals that one decides to feed into his applications should not likely use all the available RAM.

1. Optimize System Settings

* System Configuration: Change some general properties of the system concerning virtual memory efficiency. This also involves enabling additional control options in Windows such as adjusting additional system setting option on the operating system, or using other options in different operating systems such as [Linux](https://www.geeksforgeeks.org/introduction-to-linux-operating-system/) that provides different tools and commands to help in adjusting how virtual memory is utilized.
* Regular Updates: Ensure that your drivers are run in their newest version because new releases contain some enhancements and issues regarding memory management.
* Benefits of Using Virtual Memory:
* Many processes maintained in the main memory.
* A process larger than the main memory can be executed because of demand [paging](https://www.geeksforgeeks.org/paging-in-operating-system/). The OS itself loads pages of a process in the main memory as required.
* It allows greater multiprogramming levels by using less of the available (primary) memory for each process.
* It has twice the capacity for addresses as main memory.
* It makes it possible to run more applications at once.
* Users are spared from having to add memory modules when RAM space runs out, and applications are liberated from shared memory management.
* When only a portion of a program is required for execution, speed has increased.
* Memory isolation has increased security.
* It makes it possible for several larger applications to run at once.
* Memory allocation is comparatively cheap.
* It doesn't require outside fragmentation.
* It is efficient to manage logical partition workloads using the [CPU](https://www.geeksforgeeks.org/difference-between-cpu-and-gpu/).
* Automatic data movement is possible.
* Limitation of Virtual Memory:
* It can slow down the system performance, as data needs to be constantly transferred between the physical memory and the hard disk.
* It can increase the risk of data loss or corruption, as data can be lost if the hard disk fails or if there is a power outage while data is being transferred to or from the hard disk.
* It can increase the complexity of the memory management system, as the operating system needs to manage both physical and virtual memory.
* 

1. **Page replacement:**

* Page replacement algorithms are techniques used in operating systems to manage memory efficiently when the physical 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.
* If no page frame is free, the [virtual memory](https://www.geeksforgeeks.org/virtual-memory-in-operating-system/) manager performs a page replacement operation to replace one of the pages existing in memory with the page whose reference caused the page fault.
* It is performed as follows: The virtual memory manager uses a page replacement algorithm to select one of the pages currently in memory for replacement, accesses the page table entry of the selected page to mark it as “not present” in memory, and initiates a page-out operation for it if the modified bit of its page table entry indicates that it is a dirty page.
* **Common Page Replacement Techniques**
* **First In First Out (FIFO)**
* **Optimal Page replacement**
* **Least Recently Used (LRU)**
* **Most Recently Used (MRU)**

**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, 3 with 3-page frames. Find the number of page faults using FIFO Page Replacement Algorithm.**

**Optimal Page Replacement**

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

**Least Recently Used**

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

**Most Recently Used (MRU)**

* In this algorithm, page will be replaced which has been used recently.
* Belady's anomaly can occur in this algorithm.
* Belady's Anomaly is a phenomenon in operating systems where increasing the number of page frames in memory leads to an increase in the number of page faults for certain page replacement algorithms.
* Normally, as more page frames are available, the operating system has more flexibility to keep the necessary pages in memory, which should reduce the number of page fault
* This phenomenon is commonly experienced in the following page replacement algorithms:
  1. First in first out (FIFO)
  2. Second chance algorithm
  3. Random page replacement algorithm
* A page fault occurs when a page is not found in the memory and needs to be loaded from the disk. If a page fault occurs and all memory frames have been already allocated, then the replacement of a page in memory is required at the request of a new page. This is referred to as demand-paging.
* The choice of which page to replace is specified by page replacement algorithms. The commonly used page replacement algorithms are FIFO, LRU, optimal page replacement algorithms, etc.
* **Reason for Belady's Anomaly**

Belady's Anomaly happens because algorithms like FIFO replace pages based on their arrival order, without considering how often or soon they will be used again. Adding more frames can change the replacement order, causing frequently used pages to be evicted earlier and increasing page faults. This occurs due to the lack of awareness of future page references and the non-optimal nature of such algorithms.

* How Can Belady's Anomaly Be Removed?

A stack-based approach can be used to get rid of Belady's Algorithm. These are some examples of such algorithms:

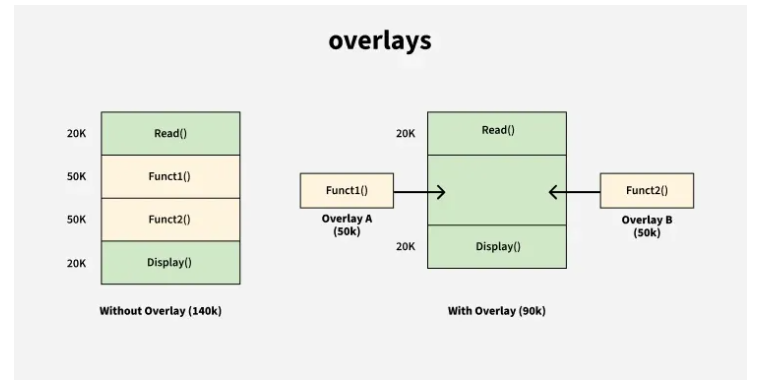
* Optimal Page Replacement Algorithm
* Least Recently Used Algorithm (LRU)

These algorithms are based on the idea that if a page is inactive for a long time, it is not being utilised frequently. Therefore, it would be best to forget about this page. This allows for improvised memory management and the abolition of Belady's anomaly.

* Features of Belady's Anomaly
  1. Page fault rate: Page fault rate is the number of page faults that occur during the execution of a process. Belady's Anomaly occurs when the page fault rate increases as the number of page frames allocated to a process increases.
  2. Page replacement algorithm: Belady's Anomaly is specific to some page replacement algorithms, including the First-In-First-Out (FIFO) algorithm and the Second-Chance algorithm.
  3. System workload: Belady's Anomaly can occur when the system workload changes. Specifically, it can happen when the number of page references in the workload increases.
  4. Impact on performance: Belady's Anomaly can significantly impact system performance, as it can result in a higher number of page faults and slower overall system performance. It can also make it challenging to choose an optimal number of page frames for a process.
* Advantages of Belady’s algorithm:
  1. Better insight into algorithm behaviour: Belady's Anomaly can provide insight into how a page replacement algorithm works and how it can behave in different scenarios. This can be helpful in designing and optimizing algorithms for specific use cases.
  2. Improved algorithm performance: In some cases, increasing the number of frames allocated to a process can actually improve algorithm performance, even if it results in more page faults. This is because having more frames can reduce the frequency of page replacement, which can improve overall performance.
* Disadvantages of Belady’s Anomaly:
  1. Poor predictability: Belady's Anomaly can make it difficult to predict how an algorithm will perform with different configurations of frames and pages, which can lead to unpredictable performance and system instability.
  2. Increased overhead: In some cases, increasing the number of frames allocated to a process can result in increased overhead and resource usage, which can negatively impact system performance.
  3. Unintuitive behaviour: Belady's Anomaly can result in unintuitive behaviour, where increasing the number of frames allocated to a process results in more page faults, which can be confusing for users and system administrators.
  4. Difficulty in optimization: Belady's Anomaly can make it difficult to optimize page replacement algorithms for specific use cases, as the behaviour of the algorithm can be unpredictable and inconsistent.
* **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.**

1. **Overlays in memory management:**

* Overlays are a memory management technique used to efficiently manage limited memory by loading only necessary parts of a program into memory at a time. This allows larger programs to run smoothly, even when the available memory is smaller than the program's size.
* The basic idea behind overlays is to load only the portion of a program that is needed for the current task. The unused parts of the program are stored on disk or other storage and are loaded into memory as needed. Once a part of the program finishes executing, it is unloaded from memory, freeing up space for the next part to be loaded. This method allows programs to be larger than the available physical memory, without running into memory constraints.
* **How Overlays Work:**



* **Overlays Driver:**

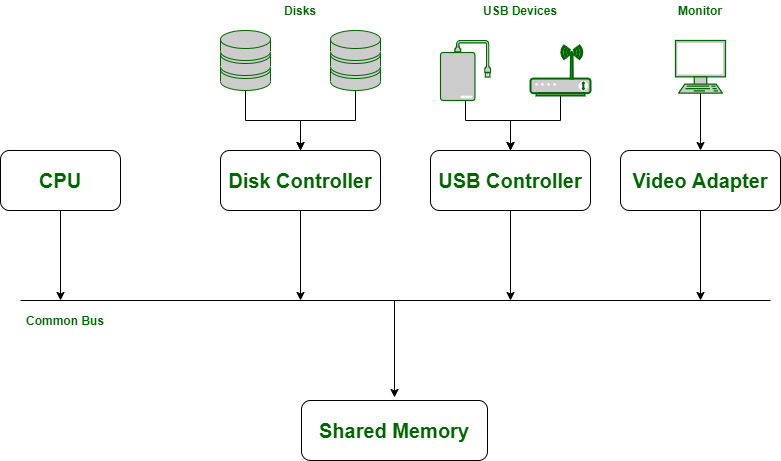
The overlays driver is the user's responsibility. The operating system does not provide an automatic mechanism for swapping the different parts of the program in and out of memory. The user must manually manage the overlay process.

* The user must load the required part (either Pass 1 or Pass 2) into memory.
* After one pass completes, the user must unload that pass from memory and load the next pass.
* The overlays driver facilitates this by managing the swapping of code in and out of memory.
* **Advantages of using overlays include:**
  1. Increased memory utilization: Overlays allow multiple programs to share the same physical memory space, increasing memory utilization and reducing the need for additional memory.
  2. Reduced load time: Only the necessary parts of a program are loaded into memory, reducing load time and increasing performance.
  3. Improved reliability: Overlays reduce the risk of memory overflow, which can cause crashes or data loss.
  4. Reduce memory requirement
  5. Reduce time requirement
* **Disadvantages of using overlays include:**
  1. Complexity: Overlays can be complex to implement and manage, especially for large programs.
  2. Performance overhead: The process of loading and unloading overlays can result in increased CPU and disk usage, which can slow down performance.
  3. Compatibility issues: Overlays may not work on all hardware and software configurations, making it difficult to ensure compatibility across different systems.
  4. Overlap map must be specified by programmer
  5. Programmer must know memory requirement

1. **I/O management:**

* Input/Output (I/O) operations are how a computer communicates with external devices such as hard drives, keyboards, printers, and network interfaces.
* Since I/O devices are much slower than the CPU, efficient management of I/O requests is crucial. This is where I/O Scheduling Algorithms come in.
* These algorithms determine the order in which I/O requests are handled to improve system performance, reduce wait times, and ensure fairness among processes.
* **Common algorithms include FCFS (First-Come-First-Serve), SSTF (Shortest Seek Time First), and SCAN.**
* **How I/O Operations Are Performed**

1. I/O Request Initiation: When a user or program requests an I/O operation (like opening a file), the OS communicates with the device driver to handle the request.
2. I/O Traffic Controller: The I/O Traffic Controller keeps track of the status of all devices, control units, and communication channels. It ensures that devices are ready and available to handle the request.
3. I/O Scheduler: The I/O Scheduler determines the order in which I/O requests are processed. It manages access to devices based on priority, fairness, and availability to optimize system performance.
4. I/O Device Handler: The I/O Device Handler manages device interrupts and oversees the data transfer. It ensures that data is transferred between the device and memory (or CPU) properly.
5. Completion and Notification: Once the I/O operation is complete, the OS informs the program or user that the task is finished.



* **I/O Scheduling Algorithms( done in notebook)**
  1. **FCFS [First come first server].**
  2. **SSTF [Shortest seek time first].**
  3. **SCAN**
  4. **Look**

**N-Step Scan**

**C-SCAN**

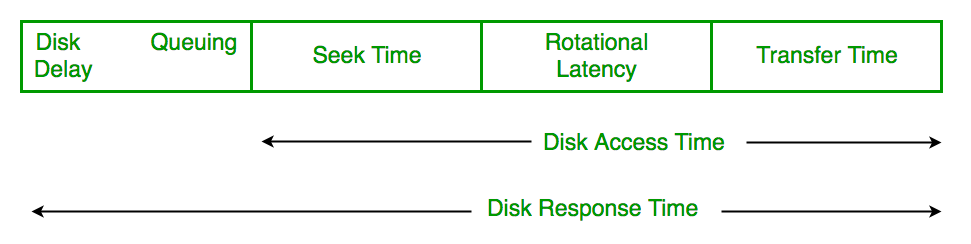
**C-LOOK**

1. **Disk scheduling:**

* Disk scheduling algorithms are crucial in managing how data is read from and written to a computer's hard disk.
* These algorithms help determine the order in which disk read and write requests are processed, significantly impacting the speed and efficiency of data access.
* Disk scheduling is a technique operating systems use to manage the order in which disk I/O (input/output) requests are processed. Disk scheduling is also known as I/O Scheduling.
* The main goals of disk scheduling are to optimize the performance of disk operations, reduce the time it takes to access data and improve overall system efficiency.
* **Importance of Disk Scheduling in Operating System**
  1. Multiple I/O requests may arrive by different processes and only one I/O request can be served at a time by the disk controller. Thus other I/O requests need to wait in the waiting queue and need to be scheduled.
  2. Two or more requests may be far from each other so this can result in greater disk arm movement.
  3. Hard drives are one of the slowest parts of the computer system and thus need to be accessed in an efficient manner.
* **Key Terms Associated with Disk Scheduling**
  1. Seek Time: Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or written. So the disk scheduling algorithm that gives a minimum average seek time is better.
  2. Rotational Latency: Rotational Latency is the time taken by the desired sector of the disk to rotate into a position so that it can access the read/write heads. So the disk scheduling algorithm that gives minimum rotational latency is better.
  3. Transfer Time: Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and the number of bytes to be transferred.
  4. Disk Access Time:

Disk Access Time = Seek Time + Rotational Latency + Transfer Time

Total Seek Time = Total head Movement \* Seek Time



* 1. Disk Response Time: Response Time is the average time spent by a request waiting to perform its I/O operation. The average Response time is the response time of all requests. Variance Response Time is the measure of how individual requests are serviced with respect to average response time. So the disk scheduling algorithm that gives minimum variance response time is better.
* **Goal of Disk Scheduling Algorithms**
  1. Minimize Seek Time
  2. Maximize Throughput
  3. Minimize Latency
  4. Fairness
  5. Efficiency in Resource Utilization
* **Disk Scheduling Algorithms (done in notebook)**
  1. FCFS (First Come First Serve)
  2. SSTF (Shortest Seek Time First)
  3. SCAN
  4. C-SCAN
  5. LOOK
  6. C-LOOK
  7. RSS (Random Scheduling)
  8. LIFO (Last-In First-Out)
  9. N-STEP SCAN
  10. F-SCAN

1. **File system organization**

* A computer file is defined as a medium used for saving and managing data in the computer system. The data stored in the computer system is completely in digital format, although there can be various types of files that help us to store the data.
* File systems are a crucial part of any operating system, providing a structured way to store, organize, and manage data on storage devices such as hard drives, SSDs, and USB drives.
* A file system is a method an operating system uses to store, organize, and manage files and directories on a storage device.
* **Some common types of file systems include:**
  1. **FAT (File Allocation Table):** An older file system used by older versions of Windows and other operating systems.
  2. **NTFS (New Technology File System):** A modern file system used by Windows. It supports features such as file and folder permissions, compression, and encryption.
  3. **ext (Extended File System):** A file system commonly used on [Linux](https://www.geeksforgeeks.org/beginners-guide-to-linux-system-administration/)and [Unix](https://www.geeksforgeeks.org/linux-tutorial/)-based operating systems.
  4. **HFS (Hierarchical File System):** A file system used by macOS.
  5. **APFS (Apple File System):** A new file system introduced by Apple for their Macs and iOS devices.
* A file is a collection of related information that is recorded on secondary storage. Or file is a collection of logically related entities. From the user’s perspective, a file is the smallest allotment of logical secondary storage.
* **The name  of the file is divided into two parts as shown below:**
* Name
* Extension, separated by a period.

## Issues Handled By File System

1. We've seen a variety of data structures where the file could be kept. The file system's job is to keep the files organized in the best way possible.
2. A free space is created on the hard drive whenever a file is deleted from it. To reallocate them to other files, many of these spaces may need to be recovered.
3. Choosing where to store the files on the [hard disc](https://www.geeksforgeeks.org/hard-disk-drive-hdd-secondary-memory/) is the main issue with files one block may or may not be used to store a file. It may be kept in the disk's non-contiguous blocks. We must keep track of all the blocks where the files are partially located.

### Files Attributes And Their Operations

### 

### File Types and Their Contents:

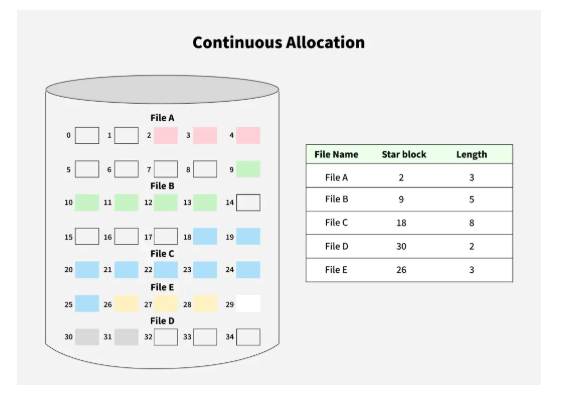
### ****--****

### ****File Allocation Methods****

* 1. Continuous Allocation
  2. Linked Allocation(Non-contiguous allocation)
  3. Indexed Allocation

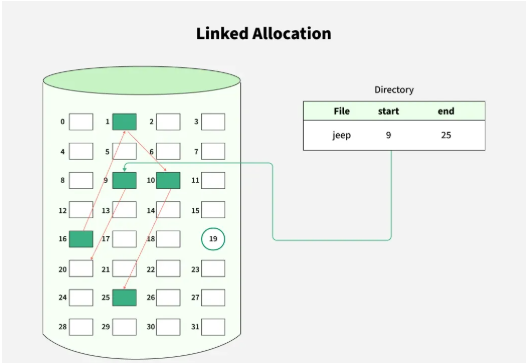
## ****Continuous Allocation****

* A single continuous set of blocks is allocated to a file at the time of file creation. Thus, this is a pre-allocation strategy, using variable size portions.
* The file allocation table needs just a single entry for each file, showing the starting block and the length of the file. This method is best from the point of view of the individual sequential file.
* Multiple blocks can be read in at a time to improve I/O performance for sequential processing. It is also easy to retrieve a single block.
* For example, if a file starts at block b, and the ith block of the file is wanted, its location on secondary storage is simply b+i-1.

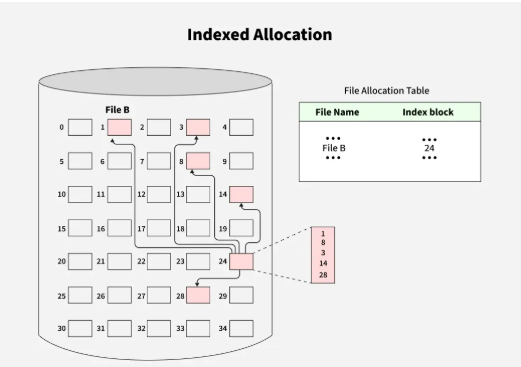


* **Disadvantages of Continuous Allocation**
* External fragmentation will occur, making it difficult to find contiguous blocks of space of sufficient length. A compaction algorithm will be necessary to free up additional space on the disk.
* Also, with pre-allocation, it is necessary to declare the size of the file at the time of creation.

### ****Linked Allocation(Non-Contiguous Allocation)****

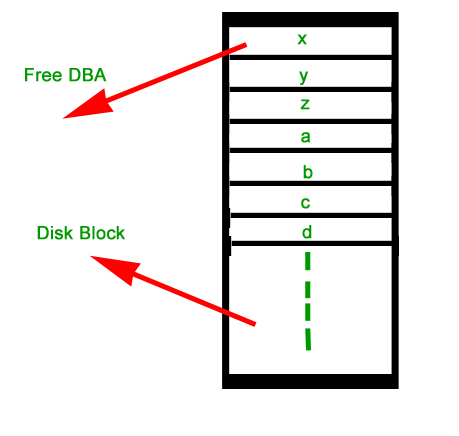
* Allocation is on an individual block basis. Each block contains a pointer to the next block in the chain. Again the file table needs just a single entry for each file, showing the starting block and the length of the file.
* Although pre-allocation is possible, it is more common simply to allocate blocks as needed. Any free block can be added to the chain. The blocks need not be continuous. An increase in file size is always possible if a free disk block is available. There is no external [fragmentation](https://www.geeksforgeeks.org/what-is-fragmentation-in-operating-system/)because only one block at a time is needed but there can be internal fragmentation but it exists only in the last disk block of the file.
* 
* **Disadvantage Linked Allocation(Non-contiguous allocation)**
* [Internal fragmentation](https://www.geeksforgeeks.org/internal-fragmentation-in-os/) exists in the last disk block of the file.
* There is an overhead of maintaining the pointer in every disk block.
* If the pointer of any disk block is lost, the file will be truncated.
* It supports only the sequential access of files.

### ****Indexed Allocation****

* It addresses many of the problems of contiguous and chained allocation. In this case, the file allocation table contains a separate one-level index for each file: The index has one entry for each block allocated to the file.
* The allocation may be on the basis of fixed-size blocks or variable-sized blocks. Allocation by blocks eliminates [external fragmentation](https://www.geeksforgeeks.org/external-fragmentation-in-os/), whereas allocation by variable-size blocks improves locality.
* This allocation technique supports both sequential and direct access to the file and thus is the most popular form of file allocation.
* 

## ****Disk Free Space Management****

* Just as the space that is allocated to files must be managed, so the space that is not currently allocated to any file must be managed.
* To perform any of the file allocation techniques, it is necessary to know what blocks on the disk are available. Thus we need a disk allocation table in addition to a file allocation table.
* The following are the approaches used for free space management.
* **Bit Tables**: This method uses a vector containing one bit for each block on the disk. Each entry for a 0 corresponds to a free block and each 1 corresponds to a block in use.   
  For example 00011010111100110001   
  In this vector every bit corresponds to a particular block and 0 implies that that particular block is free and 1 implies that the block is already occupied. A bit table has the advantage that it is relatively easy to find one or a contiguous group of free blocks. Thus, a bit table works well with any of the [file allocation methods](https://www.geeksforgeeks.org/file-allocation-methods/). Another advantage is that it is as small as possible.
* **Free Block List**: In this method, each block is assigned a number sequentially and the list of the numbers of all free blocks is maintained in a reserved block of the disk.



## ****Advantages of File System:****

## ****Organization:**** A file system allows files to be organized into directories and subdirectories, making it easier to manage and locate files.

## ****Data Protection:**** File systems often include features such as file and folder permissions, [backup and restore](https://www.geeksforgeeks.org/backup-and-restore/), and error detection and correction, to protect data from loss or corruption.

## ****Improved Performance:****A well-designed file system can improve the performance of reading and writing data by organizing it efficiently on disk.

## ****Disadvantages of File System:****

## ****Compatibility Issues:**** Different file systems may not be compatible with each other, making it difficult to transfer data between different operating systems.

## ****Disk Space Overhead:**** File systems may use some disk space to store metadata and other overhead information, reducing the amount of space available for user data.

## ****Vulnerability:**** File systems can be vulnerable to data corruption, [malware](https://www.geeksforgeeks.org/malware-and-its-types/), and other security threats, which can compromise the stability and security of the system.