Ms. Vandana Sharma

BCA 306

UNIT-5

INFORMATION MANAGEMENT

SYLLABUS: File Concept, Access Methods, Directory Structure, Device Management: Disk Structure, Disk Scheduling Algorithms

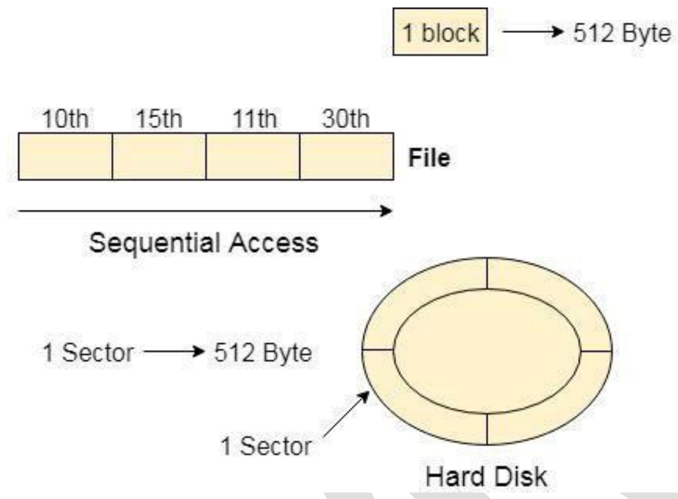
File system is the part of the operating system which is responsible for file management.It provides a mechanism to store the data and access to the file contents including data and programs. Some Operating systems treats everything as a file for example Ubuntu.
A file is a collection of correlated information which is recorded on secondary or non-volatile storage like magnetic disks, optical disks, and tapes. It is a method of data collection that is used as a medium for giving input and receiving output from that program.
In general, a file is a sequence of bits, bytes, or records whose meaning is defined by thefile creator and user. Every File has a logical location where they are located for storage and retrieval. The File system takes care of the following issues

- **File Structure**
- We have seen various data structures in which the file can be stored. The task ofthe file system is to maintain an optimal file structure.
- **Recovering Free space**
- Whenever a file gets deleted from the hard disk, there is a free space created inthe disk. There can be many such spaces which need to be recovered in order to reallocate them to other files.
- **Disk space assignment to the files**
- The major concern about the file is deciding where to store the files on the harddisk. There are various disks scheduling algorithm.
- **Tracking data location**
- A File may or may not be stored within only one block. It can be stored in the non contiguous blocks on the disk. We need to keep track of all the blocks on which thepart of the files reside.

File Access Methods in Operating System
When a file is used, information is read and accessed into computer memory and there are several ways to access this information of the file. Some systems provide only one access method for files. Other systems, such as those of IBM, support many access methods, and choosing the right one for a particular application is a major design problem.
There are three ways to access a file into a computer system: Sequential-Access, Direct Access, Index sequential Method.

1- Sequential Access –

It is the simplest access method. Information in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editor and compiler usually access the file in this fashion. Read and write make up the bulk of the operation on a file. A read operation *-read next-* read the next position of the file and automatically advance a file pointer, which keeps track I/O location. Similarly, for the write *write next* append to the end of the file and advance to the newly written material.



Most of the operating systems access the file sequentially. In other words, we can saythat most of the files need to be accessed sequentially by the operating system. In sequential access, the OS read the file word by word. A pointer is maintained which initially points to the base address of the file. If the user wants to read first word of the file then the pointer provides that word to the user and increases its value by 1 word. This process continues till the end of the file.

Key points:

1. Data is accessed one record right after another record in an order.
   When we use read command, it move ahead pointer by one
2. When we use write command, it will allocate memory and move the pointer tothe end of the file
3. Such a method is reasonable for tape.

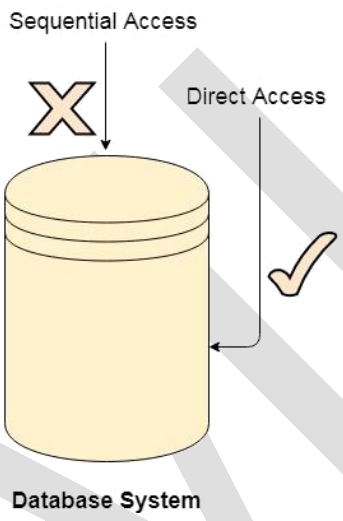
2- Direct Access –

Another method is *direct access method* also known as *relative access method*. A filed- length logical record that allows the program to read and write record rapidly. in no particular order. The direct access is based on the disk model of a file since disk allows random accessto any file block. For direct access, the file is viewed as a numbered sequence of block or record. Thus, we may read block 14 then block 59 and then we can write block 17. There is no restriction on the order of reading and writing for a direct access file.

A block number provided by the user to the operating system is normally a *relative blocknumber*, the

first relative block of the file is 0 and then 1 and so on.
The Direct Access is mostly required in the case of database systems. In most of the cases, we need filtered information from the database. The sequential access can be veryslow and inefficient in such cases.

Suppose every block of the storage stores 4 records and we know that the record weneeded is stored in 10th block. In that case, the sequential access will not beimplemented because it will traverse all the blocks in order to access the needed record. Direct access will give the required result despite of the fact that the operating system hasto perform some complex tasks such as determining the desired block number. However,that is generally implemented in database applications.



3- Index sequential method –

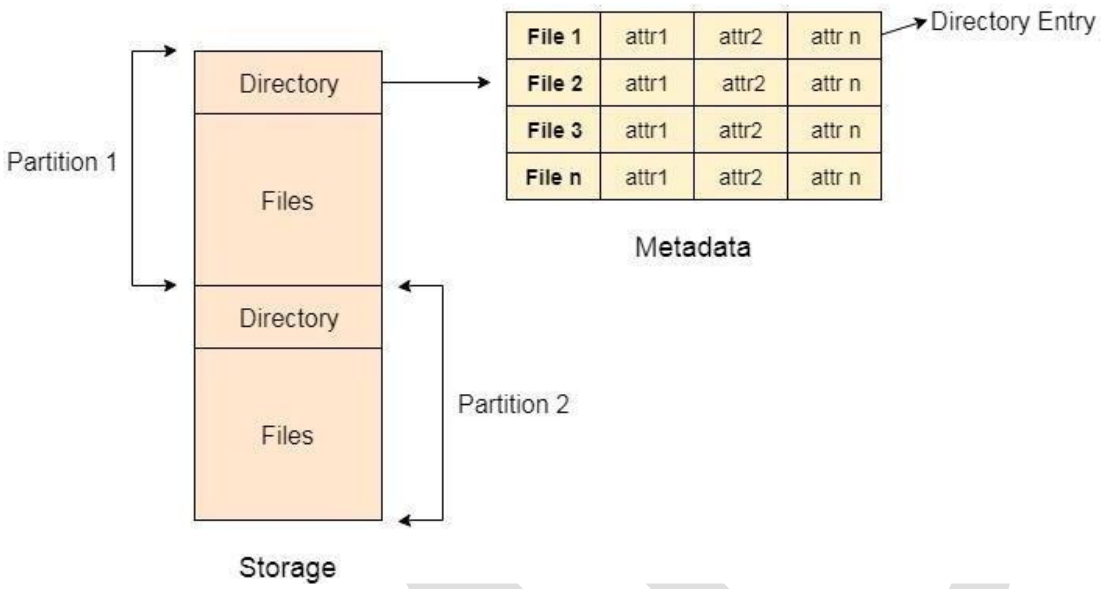
It is the other method of accessing a file which is built on the top of the direct access method. These methods construct an index for the file. The index, like an index in theback of a book, contains the pointer to the various blocks. To find a record in the file, we first search the index and then by the help of pointer we access the file directly. Provides solutions to problems of contiguous and linked allocation. A index block is created having all pointers to files. Each file has its own index block which stores the addresses of disk spaceoccupied by the file. Directory contains the addresses of index blocks of files.

Key points:

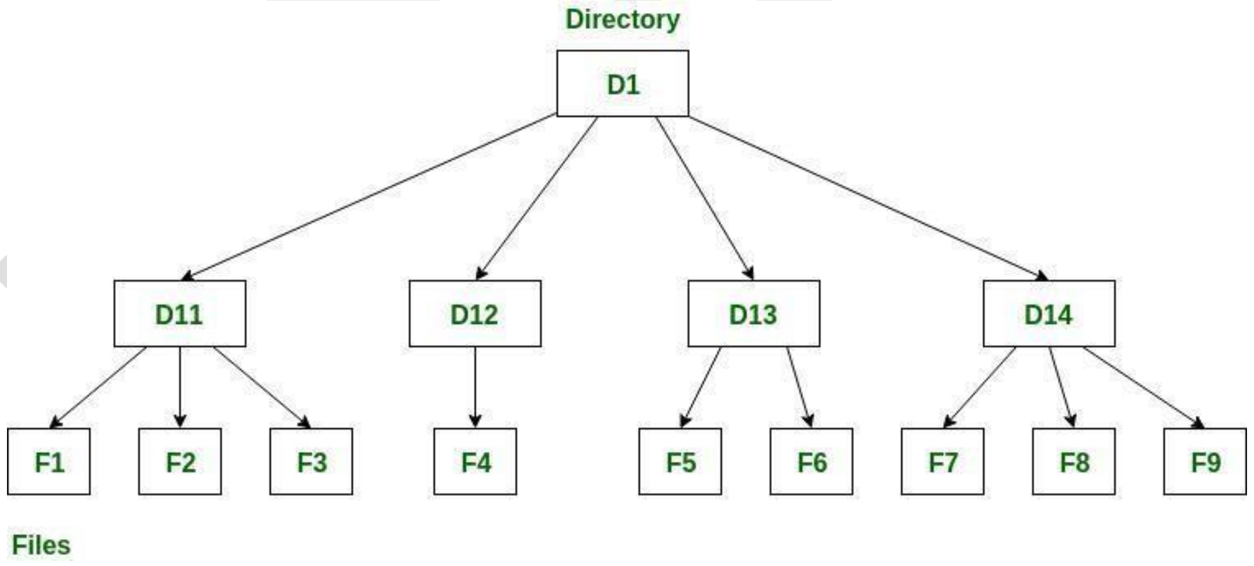
1 It is built on top of Sequential access.
2. It controls the pointer by using index.

Directory Structure:-

Directory can be defined as the listing of the related files on the disk. The directory may store some or the entire file attributes. To get the benefit of different file systems on the different operating systems, A hard disk can be divided into the number of partitions of different sizes. The partitions are also called volumes or mini disks. Each partition must have at least one directory in which, all the files of the partition can belisted. A directory entry is maintained for each file in the directory which stores all the information related to that file.



A directory can be viewed as a file which contains the Meta data of the bunch of files. A **directory** is a container that is used to contain folders and file. It organizes files and folders into a hierarchical manner.

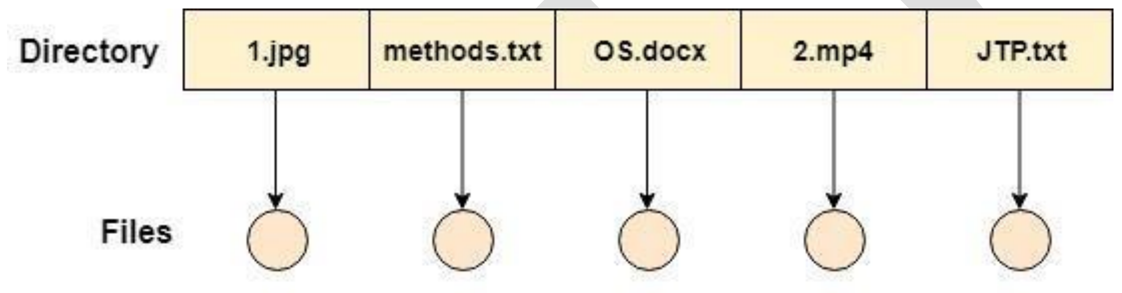


Every Directory supports a number of common operations on the file:

1. File Creation
2. Search for the file
3. File deletion
4. Renaming the file
5. Traversing Files
6. Listing of files

1. Single Level Directory

The simplest method is to have one big list of all the files on the disk. The entire system will contain only one directory which is supposed to mention all the files present in the file system. The directory contains one entry per each file present on the file system.



Single Level Directory

This type of directories can be used for a simple system.

Advantages

1. Implementation is very simple.
2. If the sizes of the files are very small then the searching becomes faster.
3. File creation, searching, deletion is very simple since we have only one directory.

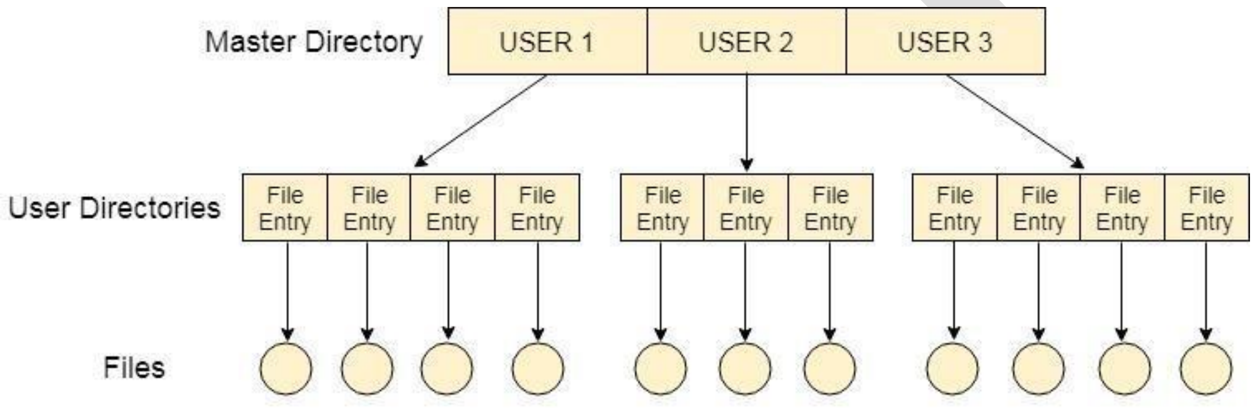
Disadvantages

1. We cannot have two files with the same name.
2. The directory may be very big therefore searching for a file may take so much time.

3. Protection cannot be implemented for multiple users.
4. There are no ways to group same kind of files.
5. Choosing the unique name for every file is a bit complex and limits the number of files in the system because most of the Operating System limits the number of characters used to construct the file name.

2. **Two Level Directory**

In two level directory systems, we can create a separate directory for each user. There is one master directory which contains separate directories dedicated to each user. For eachuser, there is a different directory present at the second level, containing group of user's file. The system doesn't let a user to enter in the other user's directory without permission.



Two Level Directory

Characteristics of two level directory system

1. Each files has a path name as ***/User-name/directory-name/***
2. Different users can have the same file name.
3. Searching becomes more efficient as only one user's list needs to be traversed.
4. The same kind of files cannot be grouped into a single directory for a particular user.

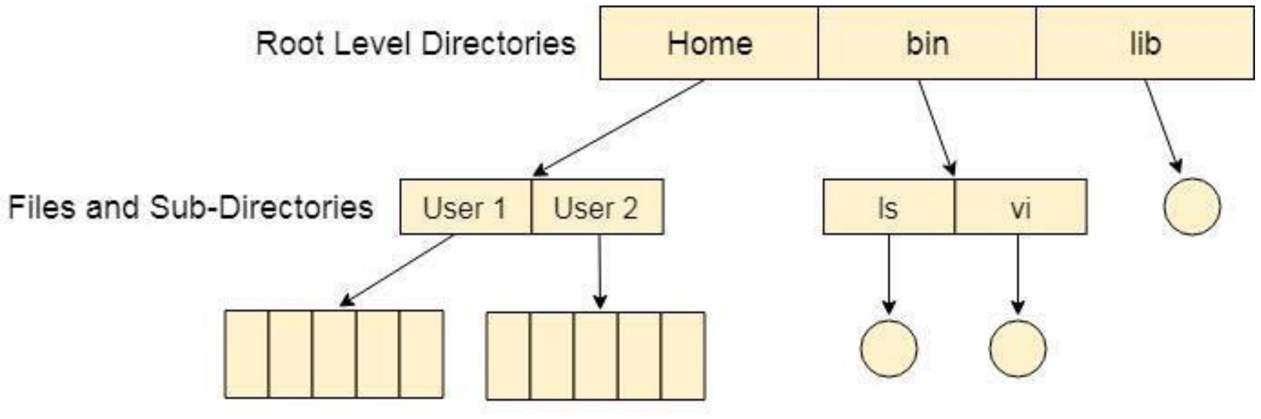
Every Operating System maintains a variable as **PWD** which contains the present directory name (present user name) so that the searching can be done appropriately.

3. **Tree Structured Directory**

In Tree structured directory system, any directory entry can either be a file or sub directory. Tree structured directory system overcomes the drawbacks of two level directory system. The similar kind of files can now be grouped in one directory.

Each user has its own directory and it cannot enter in the other user's directory. However, the user has the permission to read the root's data but he cannot write or modify this.Only administrator of the system has the complete access of root directory. Searching is more efficient in this directory structure. The concept of current working directory is used. A file can be accessed by two types of path, either relative or absolute.

Absolute path is the path of the file with respect to the root directory of the system while relative path is the path with respect to the current working directory of the system. In tree structured directory systems, the user is given the privilege to create the files as well as directories.



The Structured Directory System

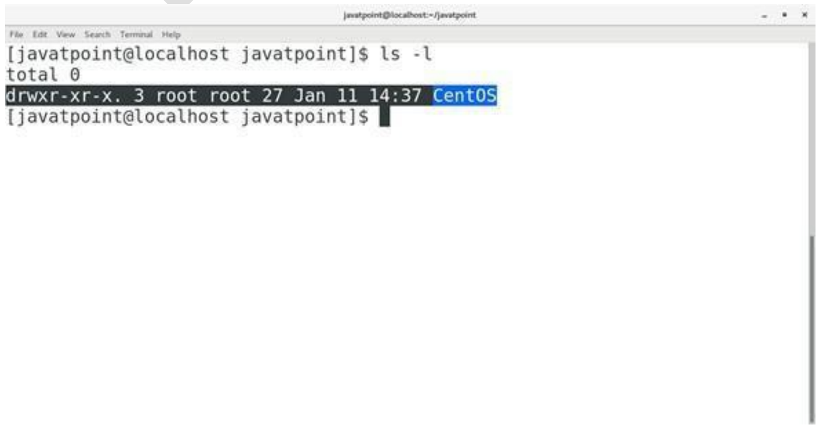
Permissions on the file and directory

A tree structured directory system may consist of various levels therefore there is a set of permissions assigned to each file and directory.

The permissions are **R W X** which are regarding reading, writing and the execution of the files or directory. The permissions are assigned to three types of users: owner, group and others.

There is a identification bit which differentiate between directory and file. For a directory, it is **d** and for a file, it is dot **(.)**

The following snapshot shows the permissions assigned to a file in a Linux based system. Initial bit **d** represents that it is a directory.


javatpoint@localhost:~/javatpoint
File Edit View Search Terminal Help

```
[javatpoint@localhost javatpoint]$ ls -l
total 0
drwxr-xr-x. 3 root root 27 Jan 11 14:37 CentOS
[javatpoint@localhost javatpoint]$
```

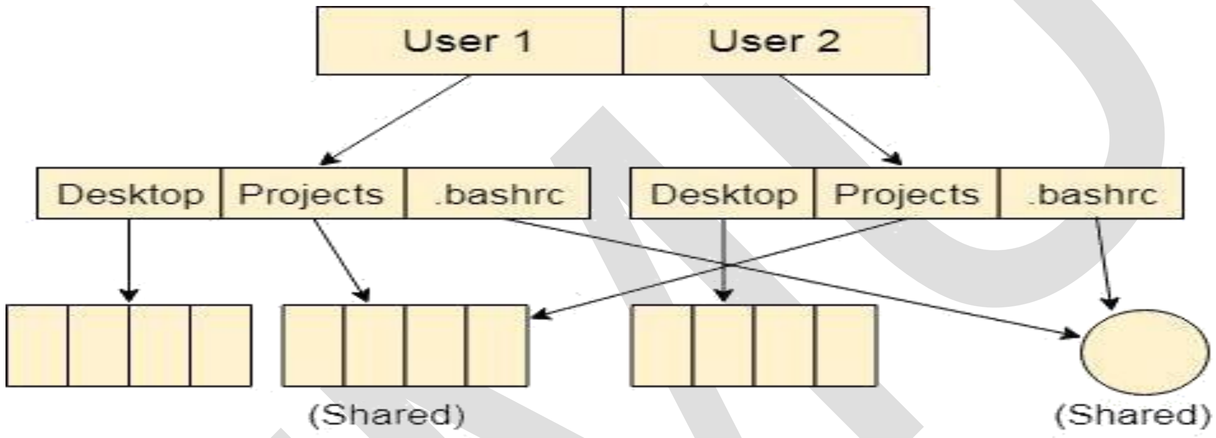
4- Acyclic-Graph Structured Directories

The tree structured directory system doesn't allow the same file to exist in multiple directories therefore sharing is major concern in tree structured directory system. We can provide sharing by making the directory an acyclic graph. In this system, two or more directory entry can point to the same file or sub directory. That file or sub directory is shared between the two directory entries.

These kinds of directory graphs can be made using links or aliases. We can have multiple paths for a same file. Links can either be symbolic (logical) or hard link (physical).

If a file gets deleted in acyclic graph structured directory system, then

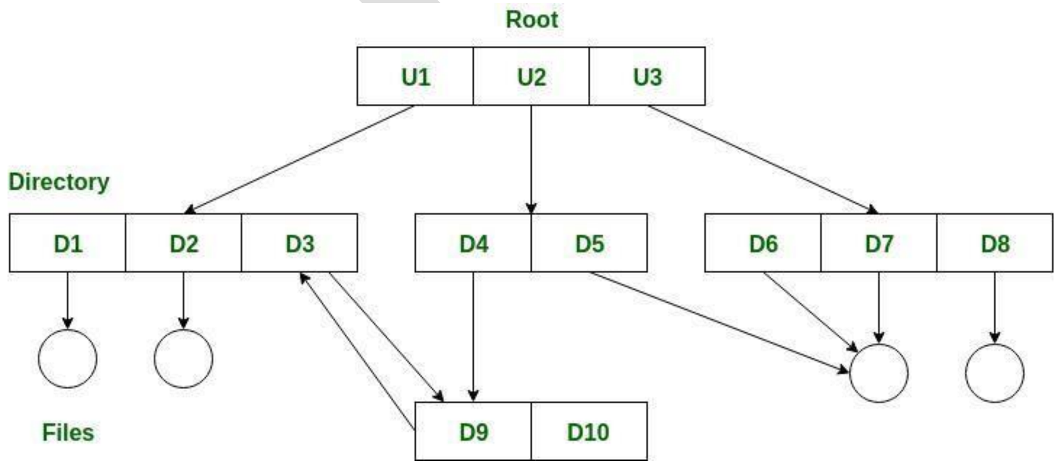
1- In the case of soft link, the file just gets deleted and we are left with a dangling pointer.
2- In the case of hard link, the actual file will be deleted only if all the references to it gets deleted.



Acyclic-Graph Structured Directory System

5- General graph directory structure –

In general graph directory structure, cycles are allowed within a directory structure where multiple directories can be derived from more than one parent directory. The main problem with this kind of directory structure is to calculate total size or space that has been taken by the files and directories.



Advantages:

1. It allows cycles.
2. It is more flexible than other directories structure.

Disadvantages:

1. It is more costly than others.
2. It needs garbage collection.

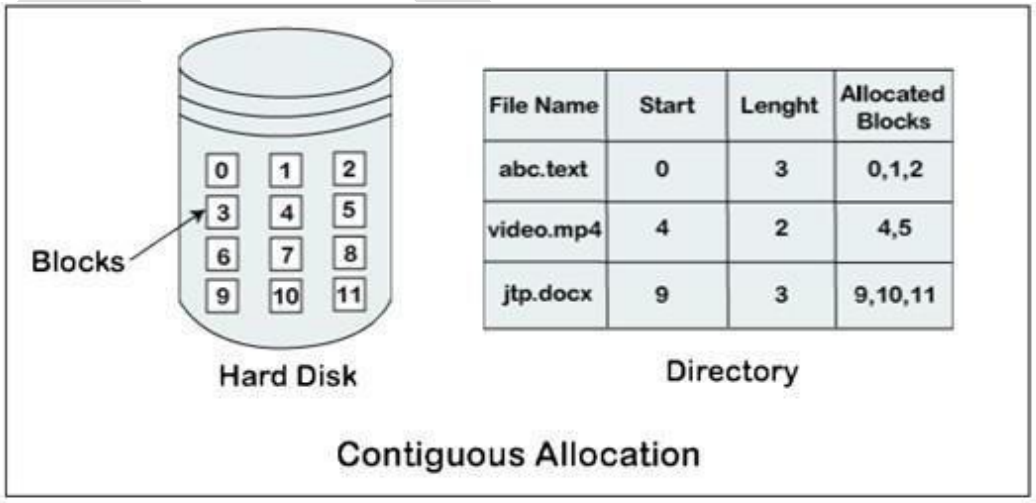
Structures of Directory in OperatingSystem

Allocation Method

The allocation method defines how the files are stored in the disk blocks. The direct access nature of the disks gives us the flexibility to implement the files. In many cases, different files or many files are stored on the same disk. The main problem that occurs in the operating system is that how we allocate the spaces to these files so that the utilization of disk is efficient and the quick access to the file is possible. There are mainly three methods of file allocation in the disk. Each method has its advantages and disadvantages. Mainly a system uses one method for all files within the system.

1- Contiguous Allocation: – Contiguous allocation is one of the most used methods for allocation. Contiguous allocation means we allocate the block in such a manner, so that in the hard disk, all the blocks get the contiguous physical block.

We can see in the below figure that in the directory, we have three files. In the table, we have mentioned the starting block and the length of all the files. We can see in the table that for each file, we allocate a contiguous block.

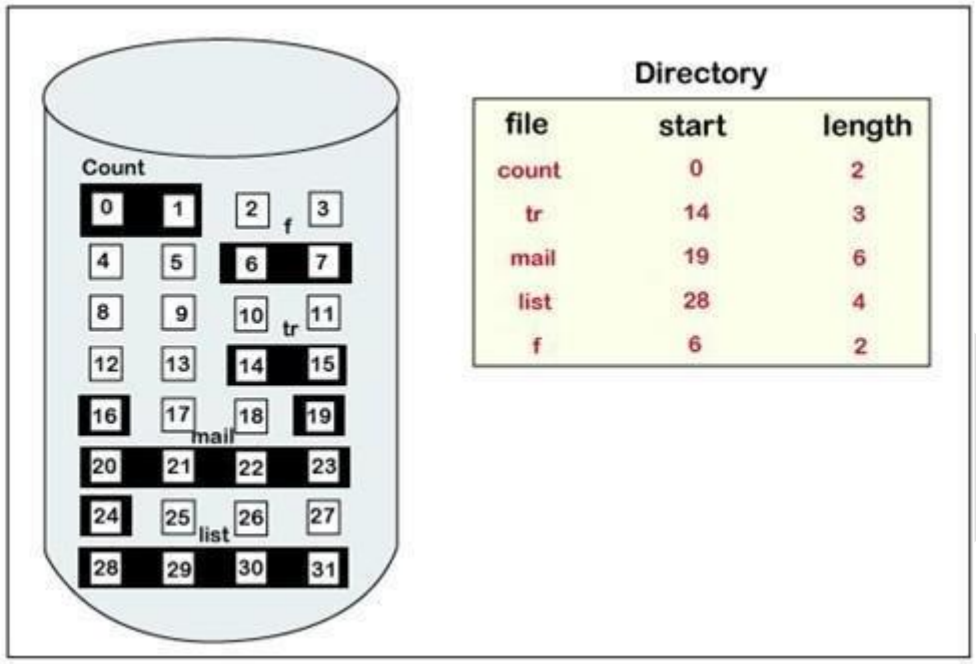


File Name	Start	Lenght	Allocated Blocks
abc.text	0	3	0,1,2
video.mp4	4	2	4,5
jtp.docx	9	3	9,10,11

Contiguous Allocation

Example of contiguous allocation

We can see in the given diagram, that there is a file. The name of the file is 'mail.' The file starts from the 19th block and the length of the file is 6. So, the file occupies 6 blocks in a contiguous manner. Thus, it will hold blocks 19, 20, 21, 22, 23, 24.



file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Advantages of Contiguous Allocation

1. The advantages of contiguous allocation are:
2. The contiguous allocation method gives excellent read performance.
3. Contiguous allocation is easy to implement.
4. The contiguous allocation method supports both types of file accessmethods that are sequential access and direct access.
5. The Contiguous allocation method is fast because, in this method number ofseeks is less due to the contiguous allocation of file blocks.

Disadvantages of Contiguous allocation

The disadvantages of contiguous allocation method are:

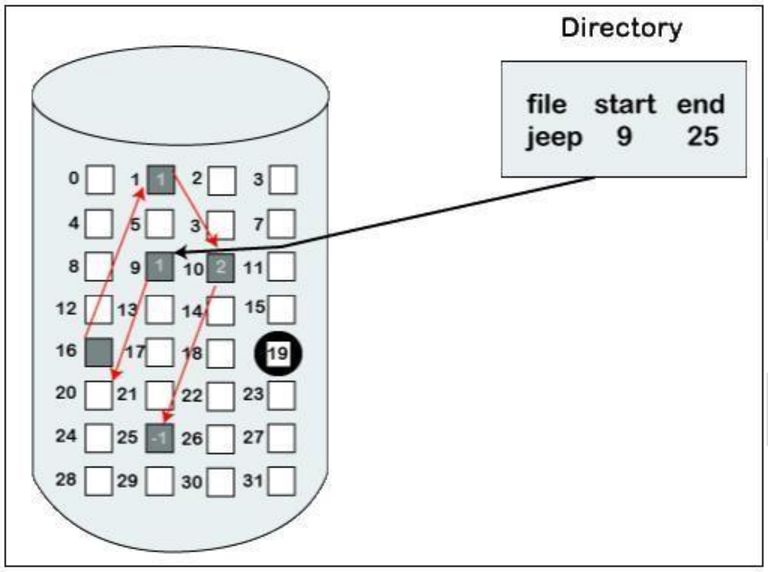
1. In the contiguous allocation method, sometimes disk can be fragmented.
2. In this method, it is difficult to increase the size of the file due to the availabilityof the contiguous memory block.

2- Linked List Allocation

The linked list allocation method overcomes the drawbacks of the contiguous allocation method. In this file allocation method, each file is treated as a linked list of disks blocks. In the linked list allocation method, it is not required that disk blocks assigned to a specific file are in the contiguous order on the disk. The directory entry comprises of a pointer for starting file block and also for the ending file block. Each disk block that is allocated or assigned to a file consists of a pointer, and that pointer point the next block of the disk, which is allocated to the same file.

Example of linked list allocation

We can see in the below figure that we have a file named 'jeep.' The value of the start is 9. So, we have to start the allocation from the 9th block, and blocks are allocated in a random manner. The value of the end is 25. It means the allocation is finished on the 25th block. We can see in the below figure that the block (25) comprised of -1, which means a null pointer, and it will not point to another block.



Advantages of Linked list allocation

There are various advantages of linked list allocation:

1. In liked list allocation, there is no external fragmentation. Due to this, we can utilizethe memory better.
2. In linked list allocation, a directory entry only comprises of the starting block address.
3. The linked allocation method is flexible because we can quickly increase the size of the file because, in this to allocate a file, we do not require a chunk of memory in a contiguous form.

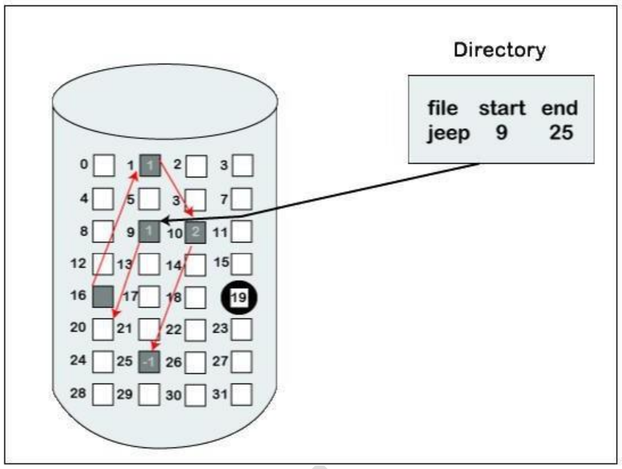
Disadvantages of Linked list Allocation

There are various disadvantages of linked list allocation:

1. Linked list allocation does not support direct access or random access.
2. In linked list allocation, we need to traverse each block.
3. If the pointer in the linked list break in linked list allocation, then the file gets corrupted.
4. In the disk block for the pointer, it needs some extra space.

3- Indexed Allocation

The Indexed allocation method is another method that is used for file allocation. In the index allocation method, we have an additional block, and that block is known as theindex block. For each file, there is an individual index block. In the index block, the ith entry holds the disk address of the ith file block. We can see in the below figure that the directory entry comprises of the address of the index block.



Advantages of Index Allocation
The advantages of index allocation are:
1. The index allocation method solves the problem of external fragmentation.
2. Index allocation provides direct access.

Disadvantages of Index Allocation
The disadvantages of index allocation are:
1. In index allocation, pointer overhead is more.
2. We can lose the entire file if an index block is not correct.
3. It is totally wastage to create an index for a small file.

Protection in File System:-
In computer systems, alot of user's information is stored, the objective of the operating system is to keep safe the data of the user from the improper access to the system. Protection can be provided in number of ways. For a single laptop system, we might provide protection by locking the computer in a desk drawer or file cabinet. For multi-user systems, different mechanisms are used for the protection.

Types of Access :
The files which have direct access of the any user have the need of protection. The files which are not accessible to other users doesn't require any kind of protection. The mechanism of the protection provide the facility of the controlled access by just limiting the types of access to the file. Access can be given or not given to any user depends on several factors, one of which is the type of access required. Several different types of operations can be controlled:
Read – Reading from a file.
Write – Writing or rewriting the file.
Execute – Loading the file and after loading the execution process starts.
Append – Writing the new information to the already existing file, editing must be at the end of the existing file.
Delete – Deleting the file which is of no use and using its space for the another data.

List – List the name and attributes of the file.

Operations like renaming, editing the existing file, copying; these can also be controlled. There are many protection mechanism. each of them mechanism have different advantages and disadvantages and must be appropriate for the intended application.

Access Control :

There are different methods used by different users to access any file. The general way of protection is to associate *identity-dependent access* with all the files and directories an list called access-control list (ACL) which specify the names of the users and the types of access associate with each of the user. The main problem with the access list is their length. If we want to allow everyone to read a file, we must list all the users with the read access. This technique has two undesirable consequences: Constructing such a list may be tedious and unrewarding task, especially if we do not know in advance the list of the users in the system.

Previously, the entry of the any directory is of the fixed size but now it changes to the variable size which results in the complicates space management. These problems can be resolved by use of a condensed version of the access list. To condense the length of the access- control list, many systems recognize three classification of users in connection with each file:

Owner – Owner is the user who has created the file.

Group – A group is a set of members who has similar needs and they are sharingthe same file.

Other – In the system, all other users are under this category.

The most common recent approach is to combine access-control lists with the normal general owner, group, and universe access control scheme. For example: Solaris uses the three categories of access by default but allows access-control lists to be added to specific files and directories when more fine-grained access control is desired.

Other Protection Approaches:

The access to any system is also controlled by the password. If the use of password are is random and it is changed often, this may be result in limit the effective access to a file. The use of passwords has a few disadvantages: The number of passwords is very large so it is difficult to remember the large passwords. If one password is used for all the files, then once it is discovered, all files are accessible; protection is on all-or-none basis.

Disk scheduling:-

Disc scheduling is an important process in operating systems that determines the order in which disk access requests are serviced. The objective of disc scheduling is to minimize the time it takes to access data on the disk and to minimize the time it takes to complete a disk access request. Disk scheduling is done by operating systems to schedule I/O requests arriving for the disk. Disk scheduling is also known as I/O Scheduling.

Importance of Disk Scheduling in Operating System

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. Two or more requests may be far from each other so this can result in greater disk arm movement. 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

- **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.
- **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.
- **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.

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

Disk scheduling algorithms are an essential component of modern operating systems and are responsible for determining the order in which disk access requests are serviced. The primary goal of these algorithms is to minimize disk access time and improve overall system performance.

Disk Scheduling Algorithms

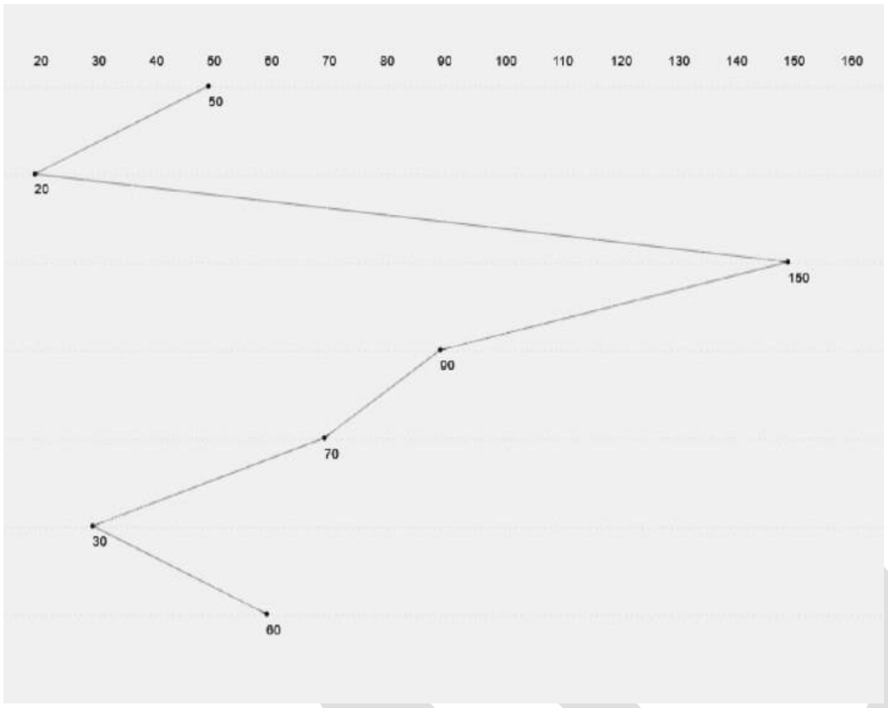
1. FCFS (First Come First Serve)
2. SSTF (Shortest Seek Time First)
3. SCAN (Elevator Algorithm)
4. C-SCAN (CIrcular SCAN)
5. LOOK
6. C-LOOK

1- First-Come-First-Serve

The First-Come-First-Served (FCFS) disk scheduling algorithm is one of the simplest and most straightforward disk scheduling algorithms used in modern operating systems. It operates on the principle of servicing disk access requests in the order in which they are received. In the FCFS algorithm, the disk head is positioned at the first request in the queue and the request is serviced. The disk head then moves to the next request in the queue and services that request. This process continues until all requests have been serviced.

Example

Suppose we have an order of disk access requests: 20 150 90 70 30 60. The disk head is currently located at track 50.

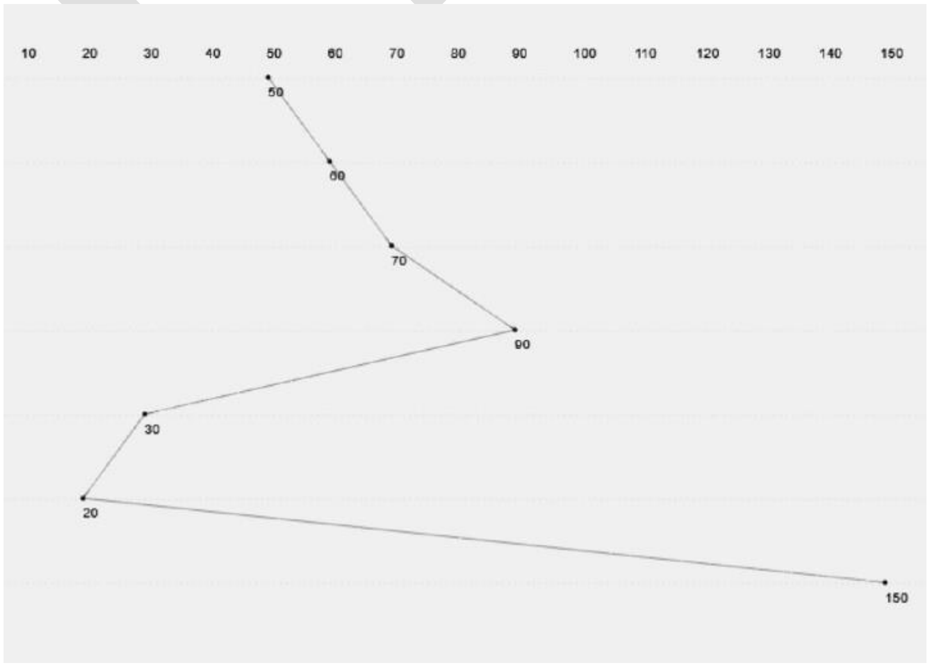


The total seek time = (50-20) + (150-20) + (150-90) + (90-70) + (70-30) + (60-30) = 310

2- Shortest-Seek-Time-First

Shortest Seek Time First (SSTF) is a disk scheduling algorithm used in operating systems to efficiently manage disk I/O operations. The goal of SSTF is to minimize the total seek time required to service all the disk access requests. In SSTF, the disk head moves to the request with the shortest seek time from its current position, services it, and then repeats this process until all requests have been serviced. The algorithm prioritizes disk access requests based on their proximity to the current position of the disk head, ensuring that the disk head moves the shortest possible distance to service each request.

Example

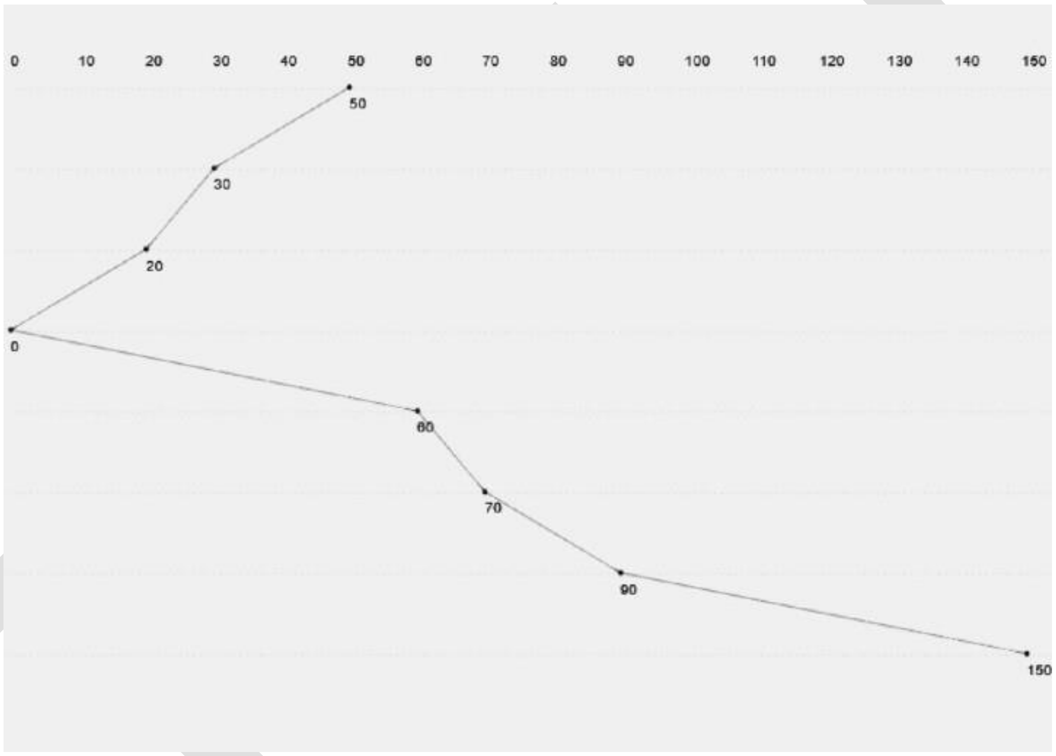


In this case, for the same order of success request, the total seek time = (60-50) + (70-60) + (90-70) + (90-30) + (30-20) + (150-20) = 240

3- SCAN

SCAN (Scanning) is a disk scheduling algorithm used in operating systems to manage disk I/O operations. The SCAN algorithm moves the disk head in a single direction and services all requests until it reaches the end of the disk, and then it reverses direction and services all the remaining requests. In SCAN, the disk head starts at one end of the disk, moves toward the other end, and services all requests that lie in its path. Once the disk head reaches the other end, it reverses direction and services all requests that it missed on the way. This continues until all requests have been serviced.

Example



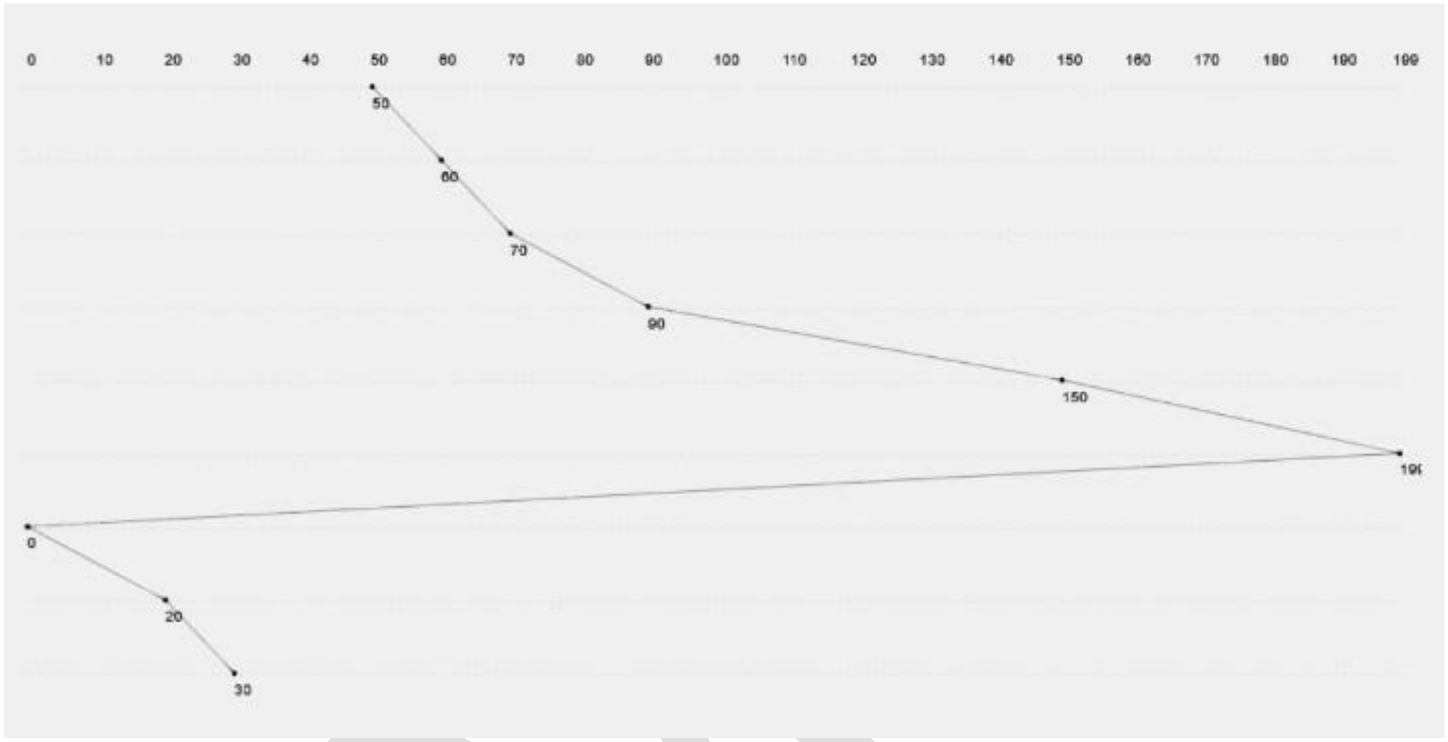
If we consider that the head direction is left in case of SCAN, the total seek time = (50-30) + (30-20) + (20-0) + (60-0) + (60-70) + (90-70) + (90-150) = 200

4- C-SCAN

The C-SCAN (Circular SCAN) algorithm operates similarly to the SCAN algorithm, but it does not reverse direction at the end of the disk. Instead, the disk head wraps around to the other end of the disk and continues to service requests. This algorithm can reduce the total distance the disk head must travel, improving disk access time. However, this algorithm can lead to long wait times for requests that are made near the end of the disk, as they must wait for the disk head to wrap around to the other end of the disk before they can be serviced. The C-SCAN algorithm is often used in modern operating systems due to its ability to reduce disk access time and improve overall system performance.

Example

For C-SCAN, the total seek time = (60-50) + (70-60) + (90-70) + (150-90) + (199-150) + (199-0) + (20-0) + (30-20) = 378

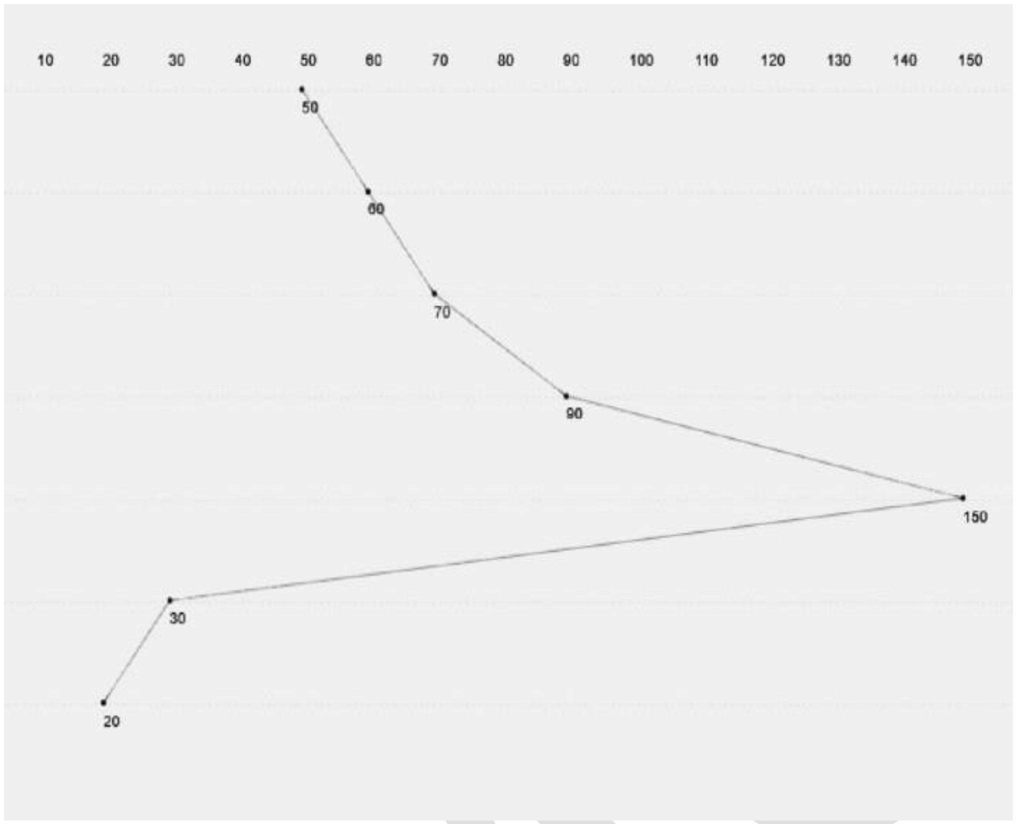


5- LOOK

The LOOK algorithm is similar to the SCAN algorithm but stops servicing requests as soon as it reaches the end of the disk. This algorithm can reduce the total distance the disk head must travel, improving disk access time. However, this algorithm can lead to long wait times for requests that are made near the end of the disk, as they must wait for the disk head to wrap around to the other end of the disk before they can be serviced. The LOOK algorithm is often used in modern operating systems due to its ability to reduce disk access time and improve overall system performance.

Example

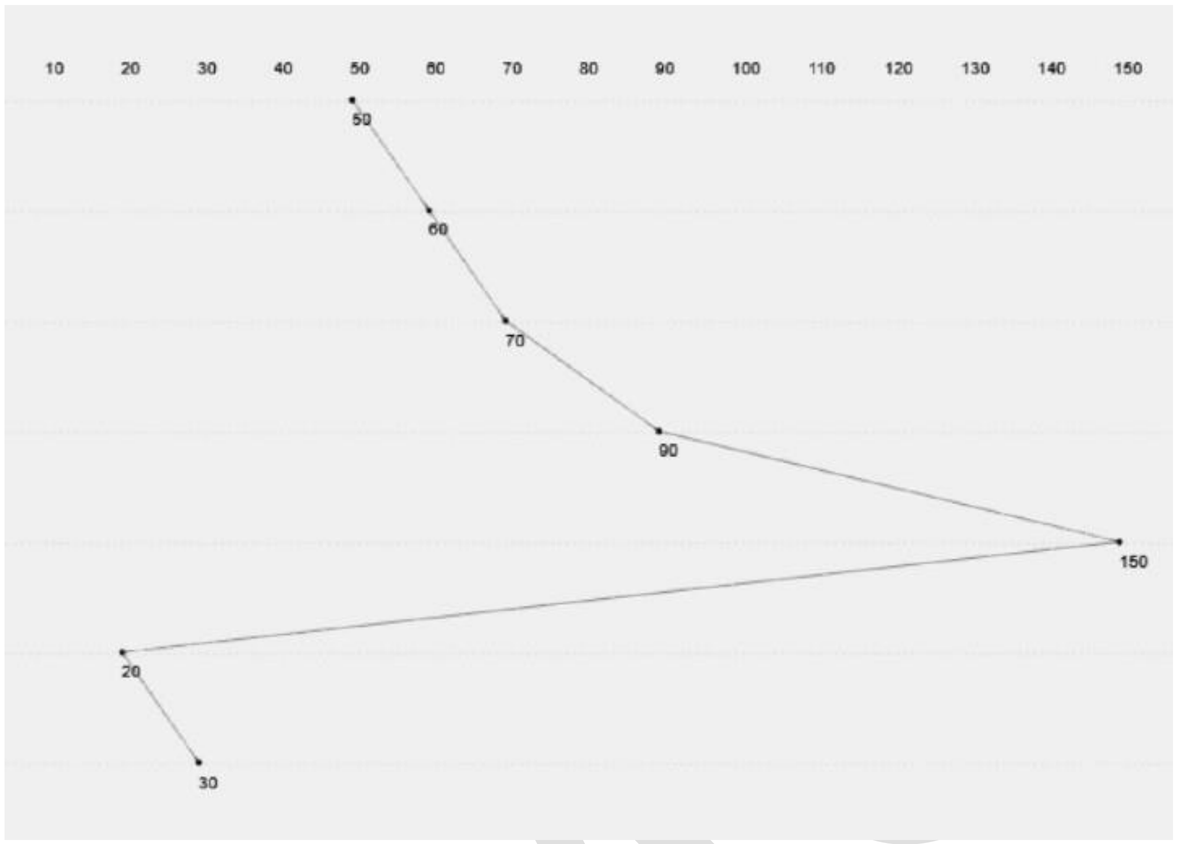
Considering the head direction is right, in this case, the total seek time = (60-50) + (70-60) + (90-70) + (150-90) + (150-30) + (30-20) = 230



6- C-LOOK

C-LOOK is similar to the C-SCAN disk scheduling algorithm. In this algorithm, goes only to the last request to be serviced in front of the head in spite of the disc arm going to the end, and then from there it goes to the other end's last request. Thus, it also prevents the extra delay which might occur due to unnecessary traversal to the end of the disk.

Example



For the C-LOOK algorithm, the total seek time = (60-50) + (70-60) + (90-70) + (150-90) + (150-20) + (30-20) = 240