# OPERATING SYSTEMS ASSIGNMENT

Name- Ansh Jaiswal
Roll no - 191541
Batch - IT-42

## * Structure of Page Table

The data structure that is used by the virtual memory system in the operating system of a computer in order to store the mapping between physical and logical and logical addresses is commonly known as Page Table.

Some of the common techniques that are used for structuring the Page Table are as follows :-

1. Hashed Page Tables
2. Inverted Page Tables.

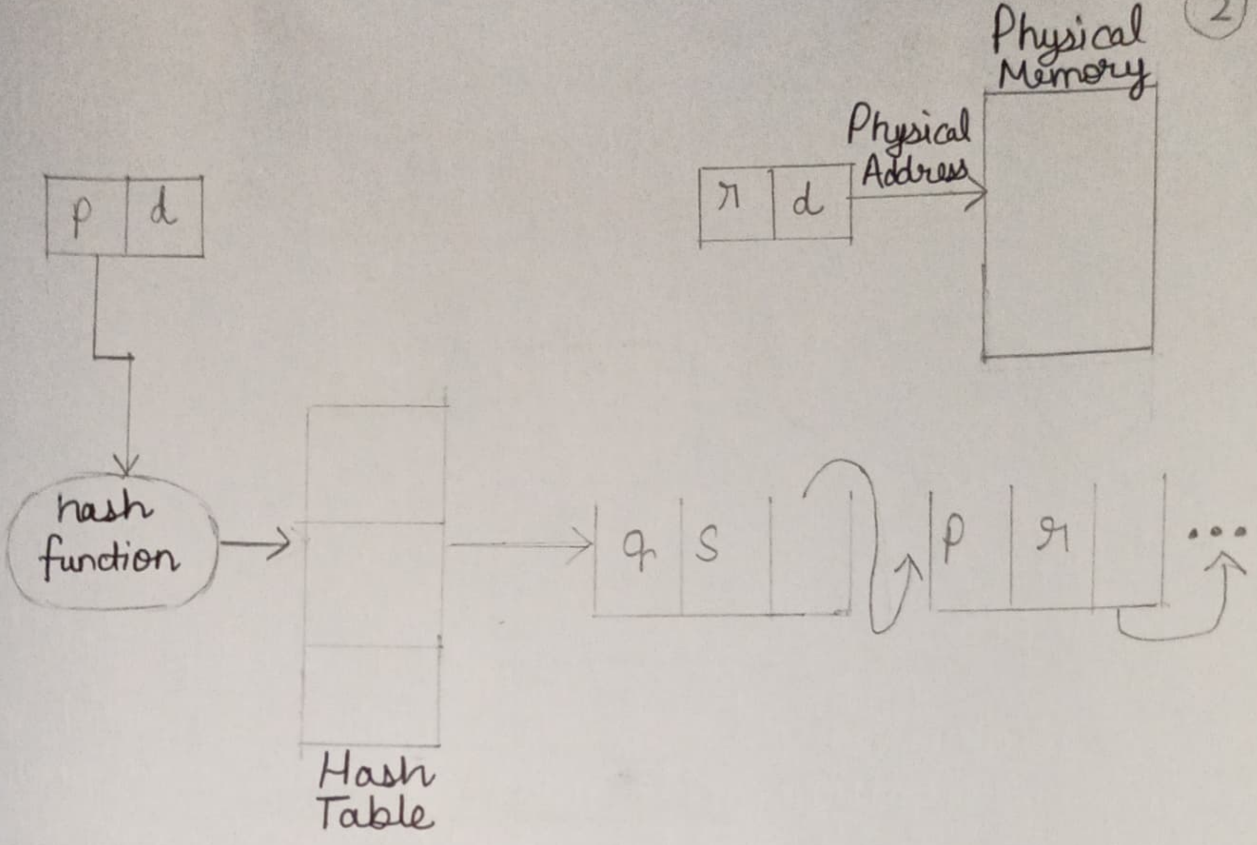
### Hashed Page Tables :-

This approach is used to handle address spaces that are larger than 32 bits.

- In this virtual page, the number is hashed into a page table.
- This page table mainly contains a chain of elements hashing to the same elements.

Each element mainly consists of :-

1. The virtual page number.
2. The value of the mapped page frame.
3. A pointer to the next element in the linked list.



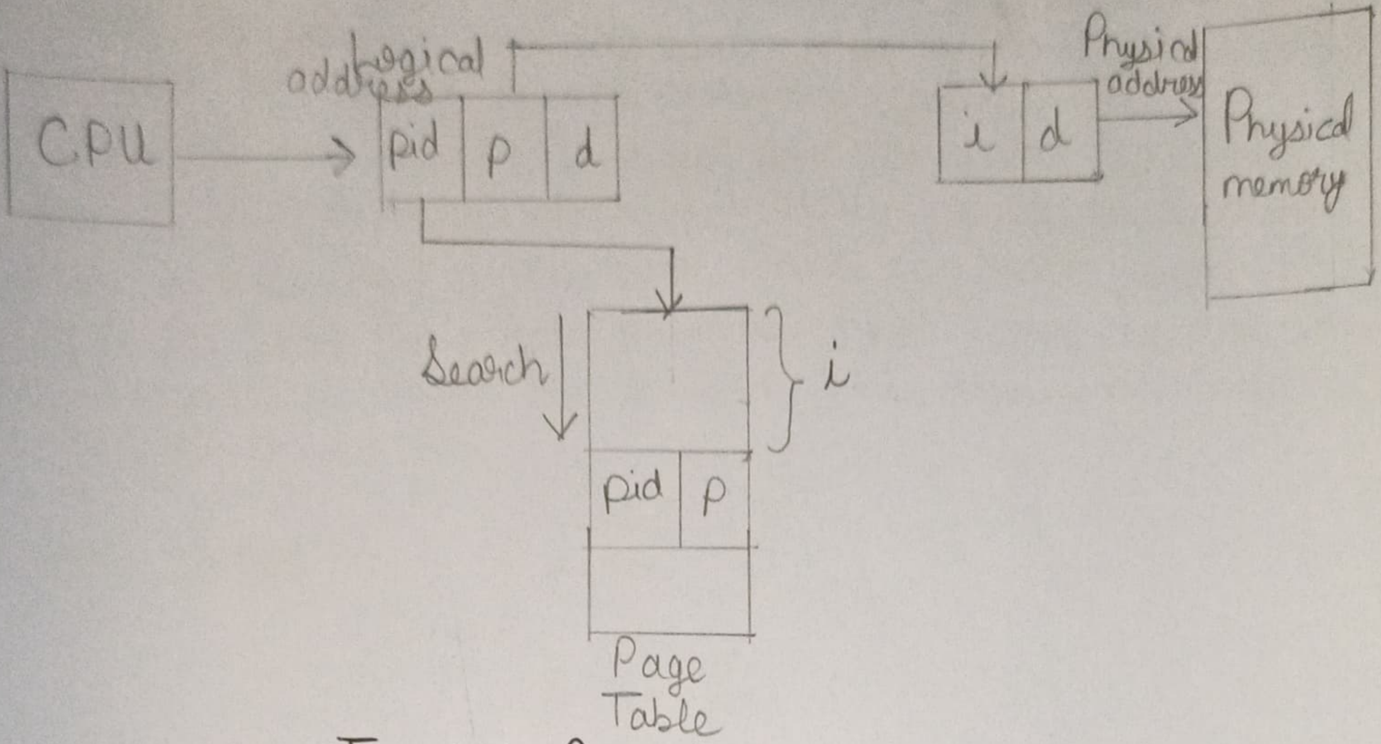
Hashed Page Table.

The virtual page numbers are compared in this chain searching for a match; if the match is found then the corresponding physical frame is extracted.

## Inverted Page Tables :-

The Inverted Page table basically combines a page table and a frame table into a single data structure.

- There is one entry for each virtual page number and a real page of memory.
- The entry mainly consists of the virtual addresses of the page stored in that real memory location along with the information about the process that owns the page.
- Though this process decreases the memory that is needed to store each page table; but it also increases the time that is needed to store each page table; but it also increases the time that is needed to search the table whenever a page reference occurs.



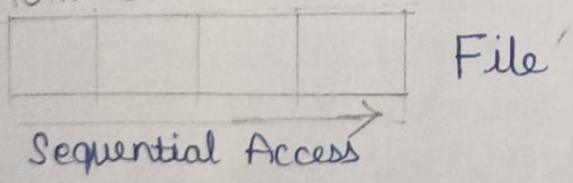
## Inverted Page Table

In this we need to track the process id of each entry, because many processes may have the same logical addresses and we need to keep it.

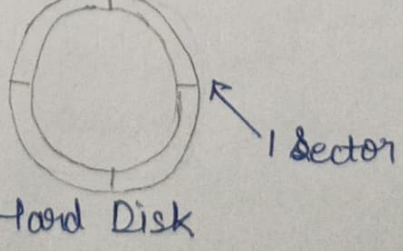
## * File Access Methods :-

### Sequential Access :-

1 block → 512 bytes



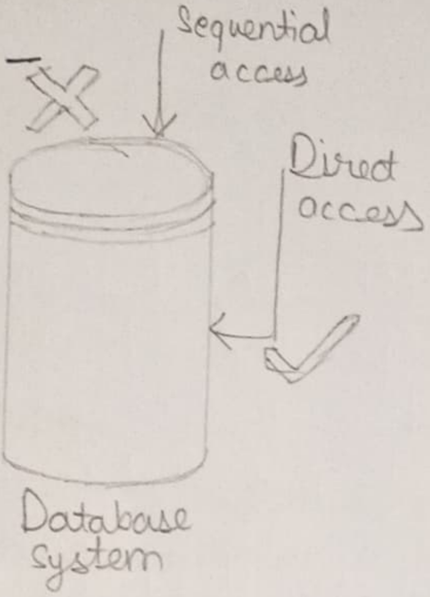
1 sector → 512 Bytes



Most of the operating systems access the file sequentially. In other words, we can say that 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 is 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.

## Direct Access :-



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 very slow and inefficient in such cases.

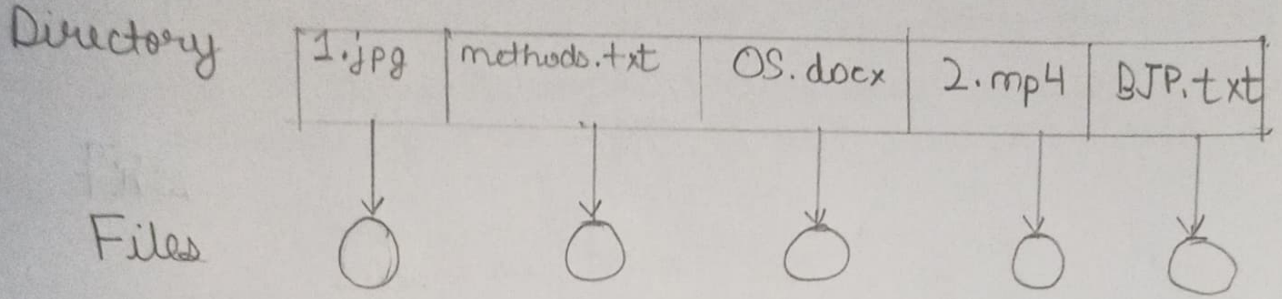
Suppose every block of the storage stores 4 blocks of records and we know that the record we needed is stored in the 10th block. In that case, the sequential access will not be implemented because it will traverse all the blocks in order to access the needed record. Direct access will give the required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

## * Directory Structure-

A directory can be viewed as a file which contains the Meto data of the bunch of files.

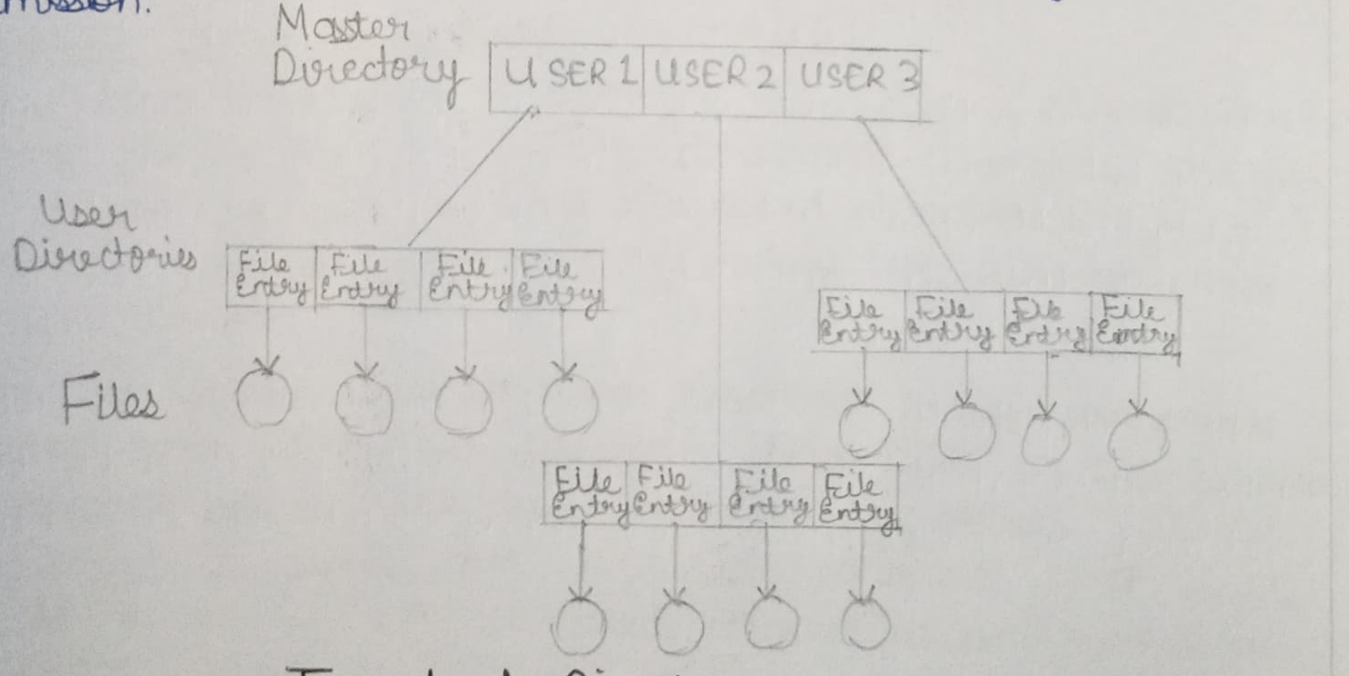
### Single level Directory-

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.



### Two level Directory-

In two level directory systems, we can create a separate directory for each other. There is one master directory which Contains separate directories dedicated to each user. For each user, there is a different directory present at the 2nd level containing group of user's files. The system doesn't let a user to enter in the other user's directory without permission.

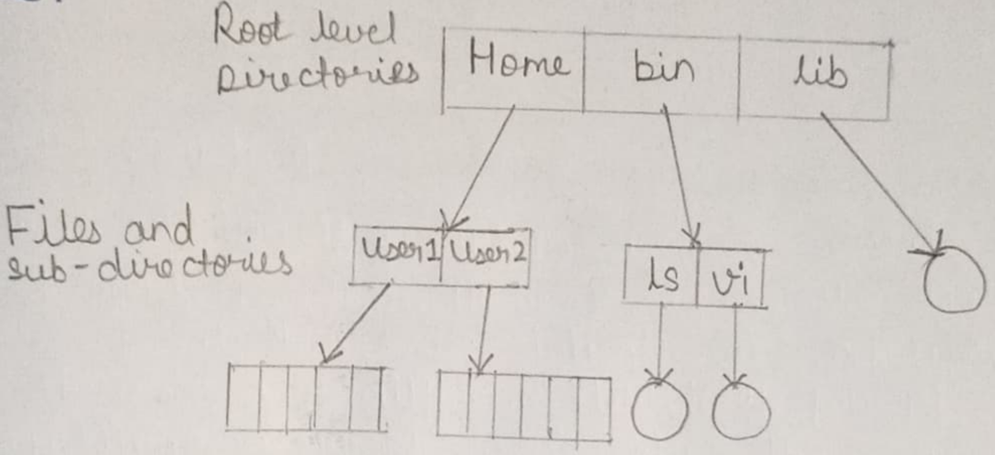


Two level Directory-

## Tree Structured Directory

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

Each user has its own directory and it cannot enter other people's directory. However, the user has the permission to read root data but he cannot write or modify it. Only the administrator of the system has complete access of this 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.



Tree Structured Directory

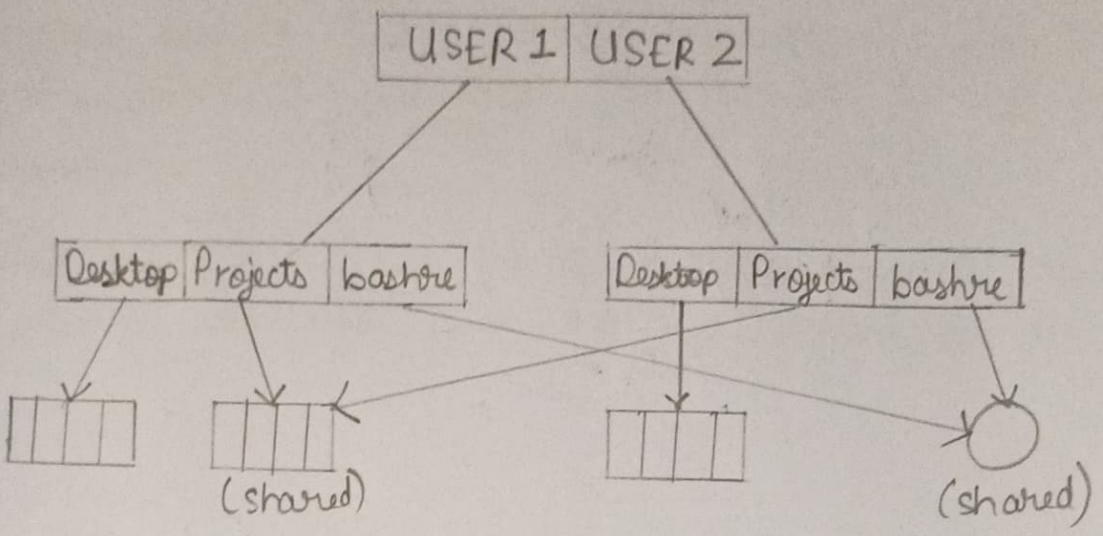
## Acyclic - Graph Structured Directories

The tree structured directory system doesn't allow the same file to exist in multiple directories therefore sharing is major concern in tree structured directories system. We can provide sharing by making the directory an acyclic graph.

In this system, two or more directory entry can pointe to the same file or sub-directory. That file or sub-directory is shared between the two directory entries.

Concept is simple but implementation is not that simple as there are a lot complexity involved in implementation, therefore noone is following it.

Simplest way is have only one copy of file but then let many user to point to it.

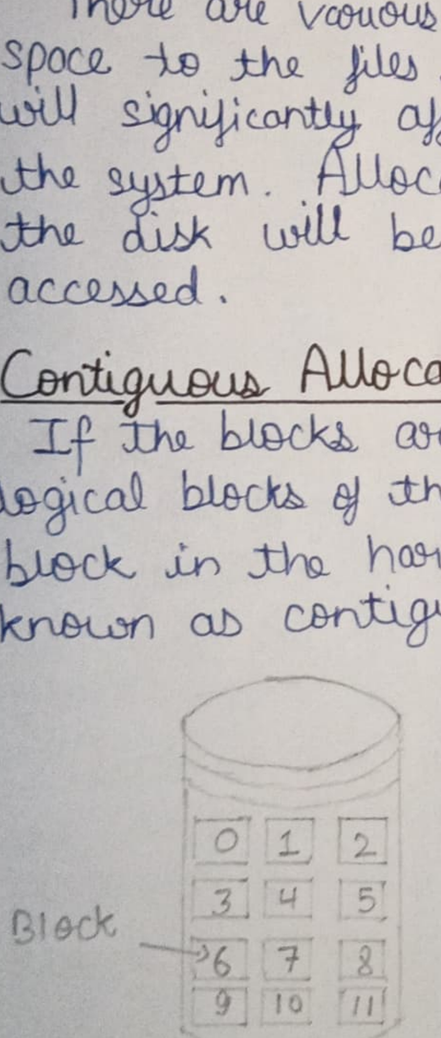


## * File Allocation Methods

There are various methods which can be used to allocate disk space to the files. Selection of any appropriate allocation method will significantly affect the performance and efficiency of the system. Allocation method provides a way in which the disk will be utilized and the files will be accessed.

### Contiguous Allocation:

If the blocks are allocated in such a way that all the logical blocks of the file get the contiguous physical block in the hard-disk then such allocation scheme is known as contiguous allocation.



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

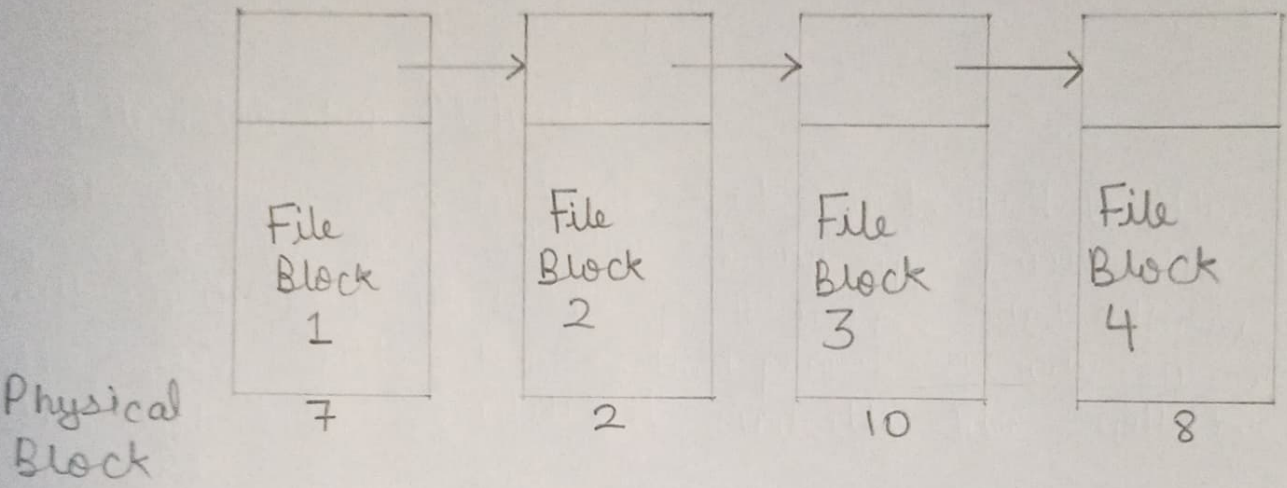
Directory

Contiguous Allocation

## Linked List Allocation :-

Linked list allocation solves all problems of contiguous allocation. In linked list allocation, each file is considered as the linked list of disk blocks. However, the disk blocks allocated to a particular file must not to be contiguous on the disk.

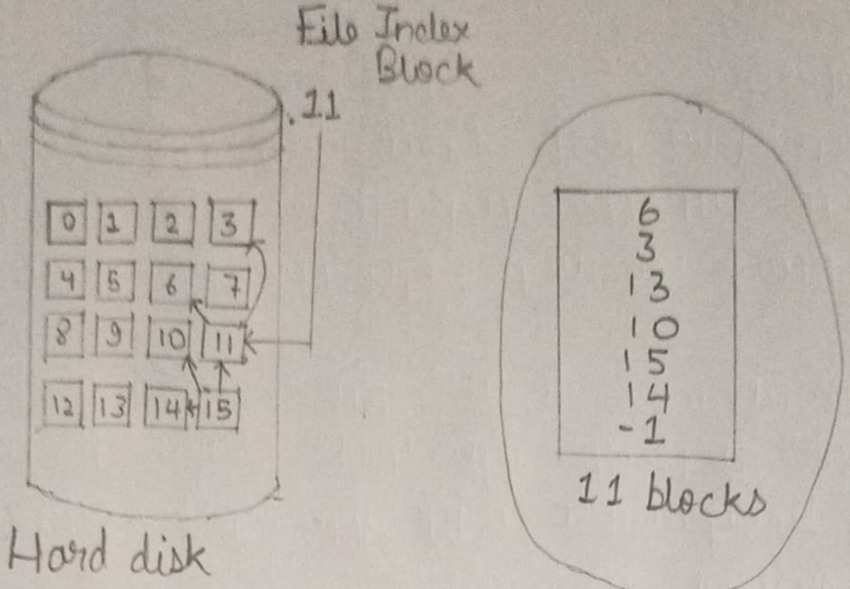
Each disk block allocated to a file contains a pointer which points to the next disk block allocated to the same file.



## Indexed Allocation :-

Instead of maintaining a file allocation table of all the disk pointers, Indexed allocation scheme stores all the disk pointers in one of the blocks called as the indexed block.

Indexed block doesnot hold the file data, but it holds the pointer to all the disk blocks allocated to that particular file. Directory entry will only contain the index block address.



Indexed Allocation