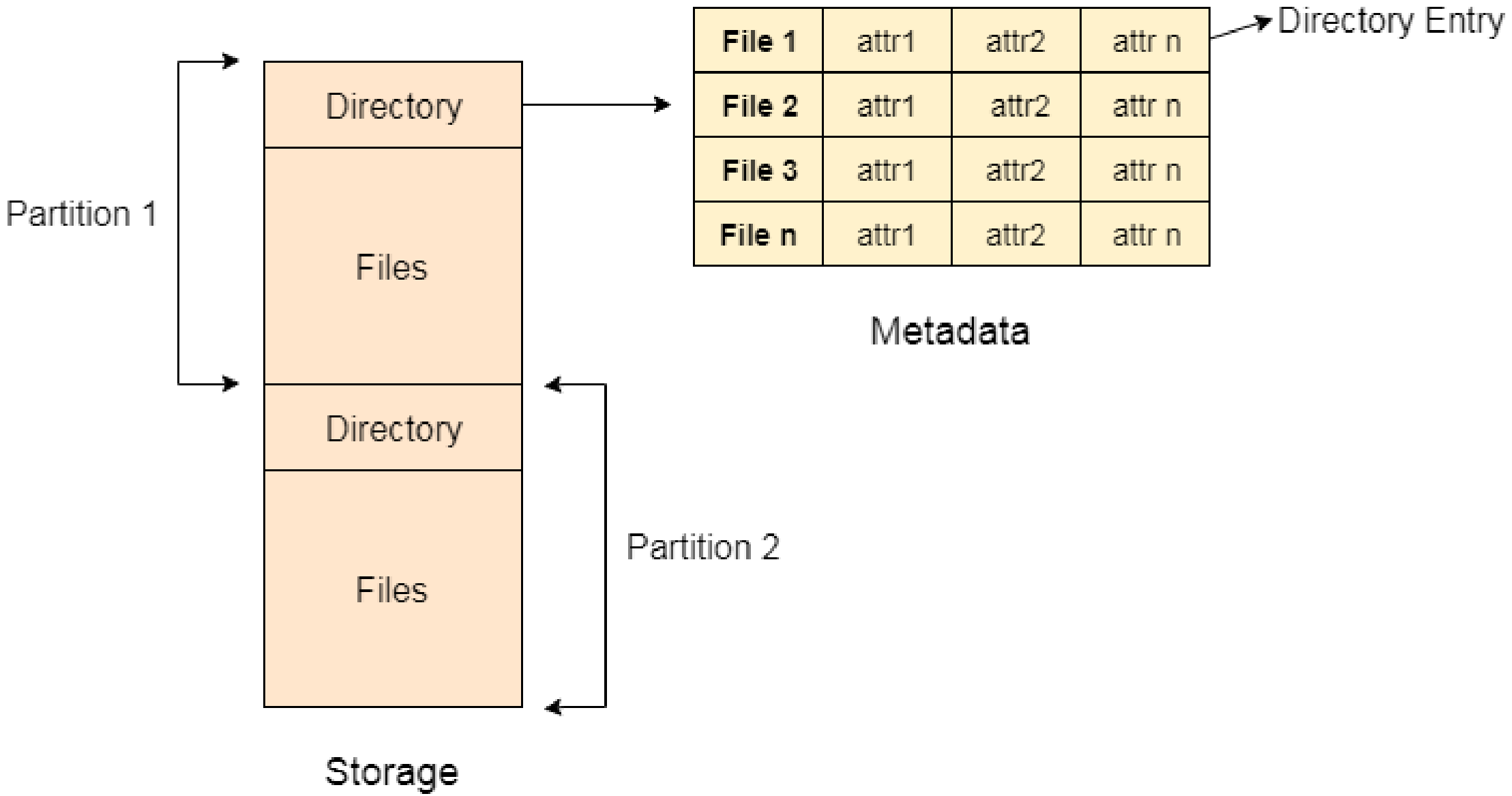
Directory Commands

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 be listed. A directory entry is maintained for each file in the directory which stores all the information related to that file.


Directory Entry
File 1
attr1
attr2
attr n
File 2
attr1
attr2
attr n
File 3
attr1
attr2
attr n
File n
attr1
attr2
attr n
Metadata
Partition 1
Directory
Files
Directory
Files
Partition 2
Storage

Types of Directory Structure

1.Single-Level Directory

2.Two-Level Directory

3.Tree-Structured Directory

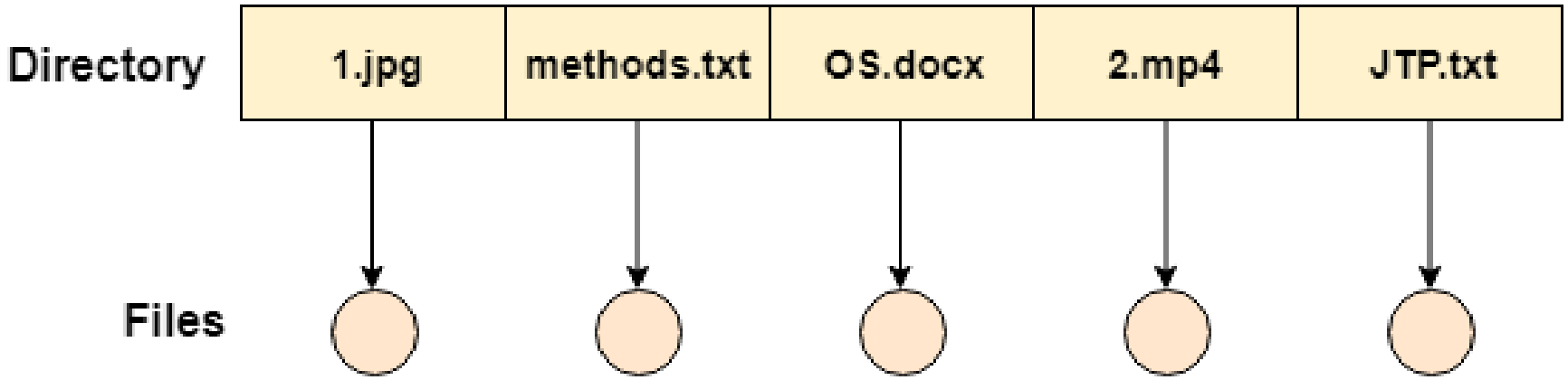
4.Acyclic Graph Directory

5.General-Graph Directory

6.Single-Level Directory: - Single-Level Directory is the easiest directory structure. There is only one directory in a single-level directory, and that directory is called a root directory. In a single-level directory, all the files are present in one directory that makes it easy to understand. In this, under the root directory, the user cannot create the subdirectories.

Single Level Directory

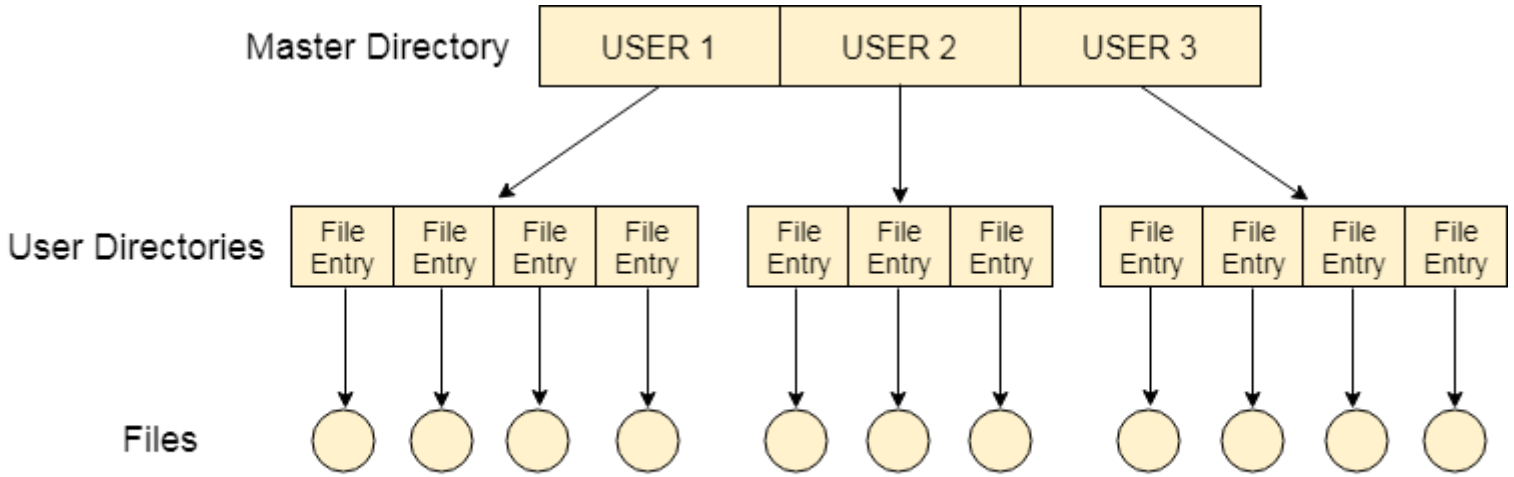
- The simplest method is to have one big list of all the files on the disk. The entire system will contain only one directory which is supposed to mention all the files present in the file system. The directory contains one entry per each file present on the file system.



Single Level Directory

Two Level Directory

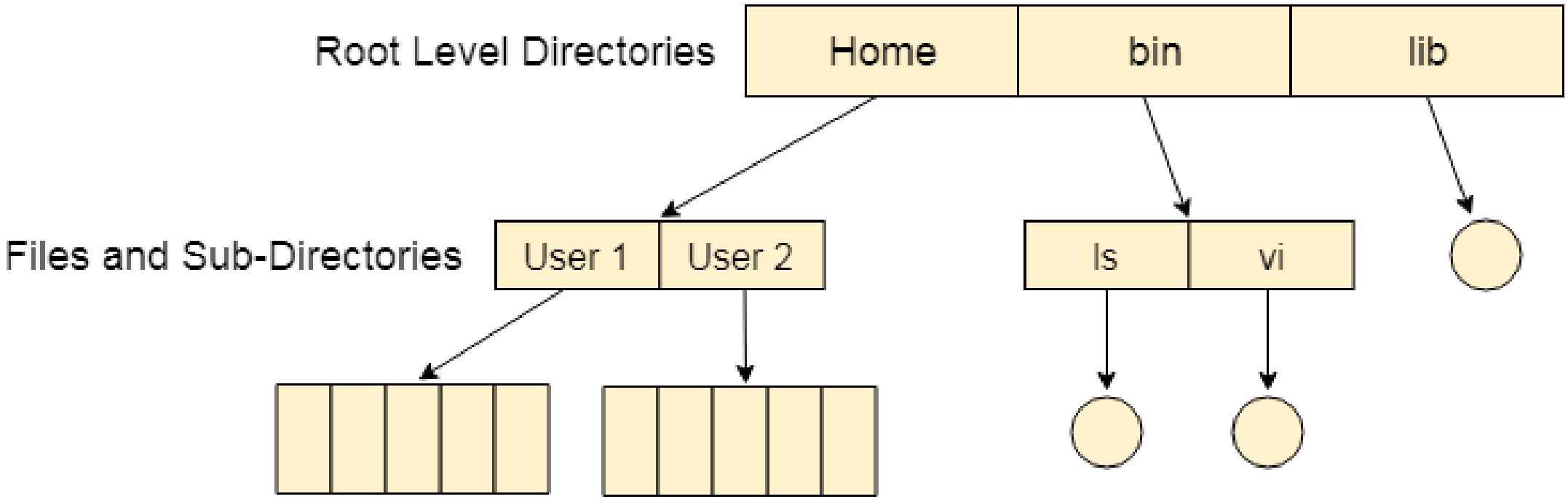
- In two level directory systems, we can create a separate directory for each user. There is one master directory which contains separate directories dedicated to each user. For each user, there is a different directory present at the second level, containing group of user's file. The system doesn't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



Two Level Directory

Three Level Directory

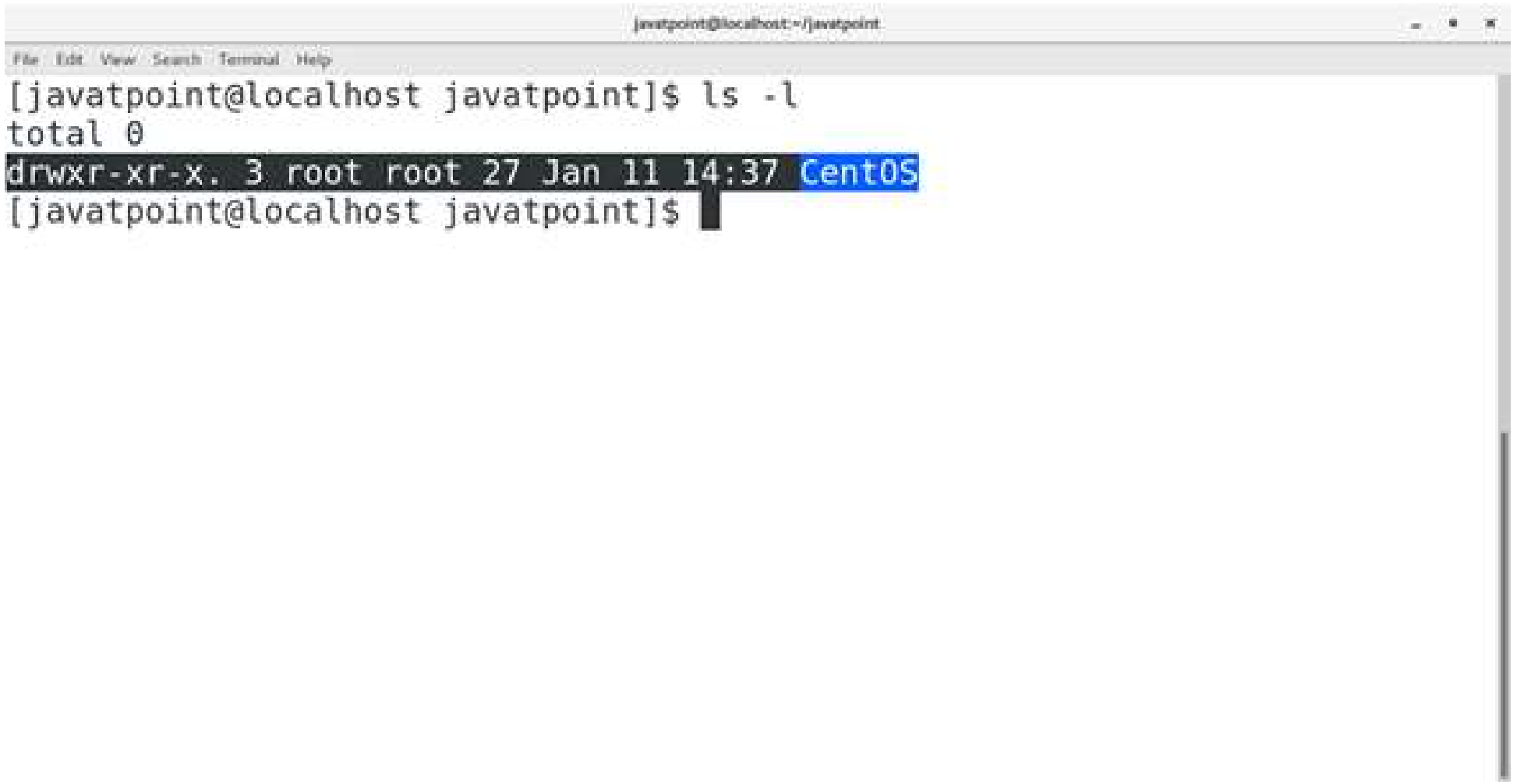
- In Tree structured directory system, any directory entry can either be a file or sub directory. Tree structured directory system overcomes the drawbacks of two level directory system. The similar kind of files can now be grouped in one directory.
- Each user has its own directory and it cannot enter 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

Permission on the File Directory

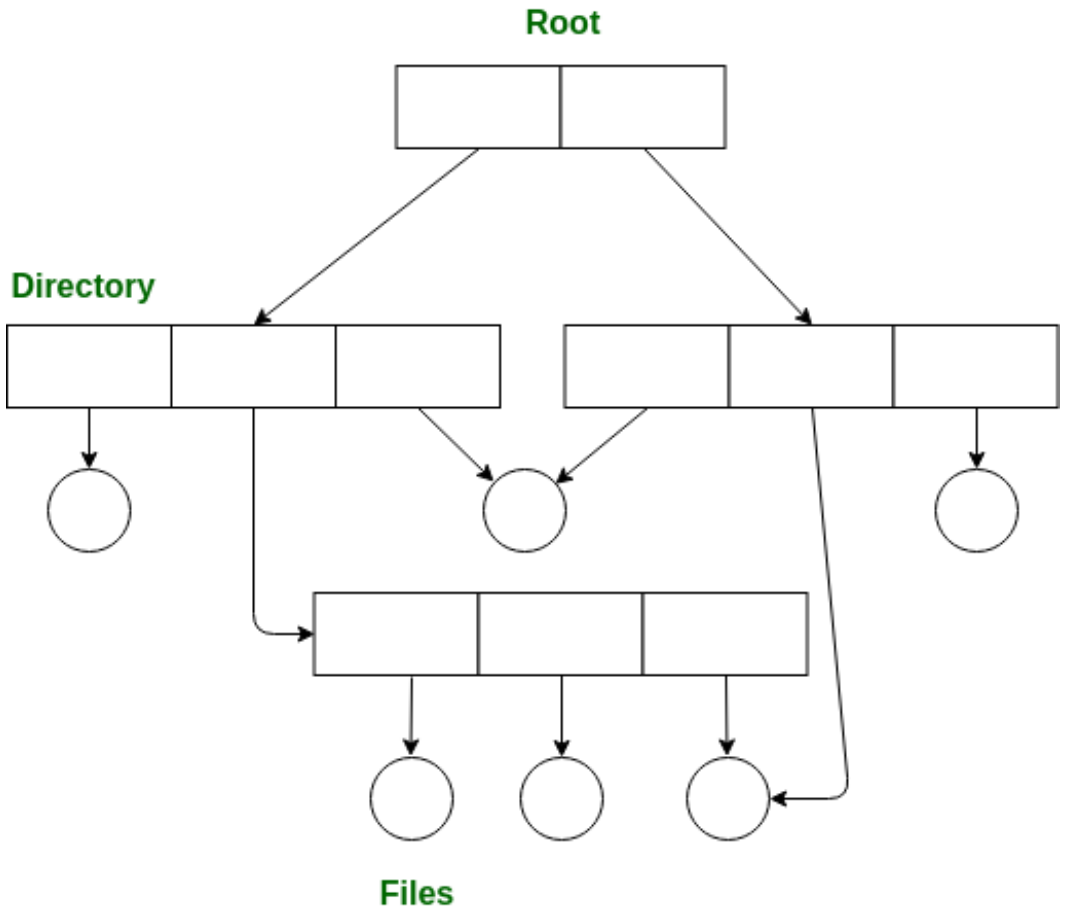
- A tree structured directory system may consist of various levels therefore there is a set of permissions assigned to each file and directory.
- The permissions are **R W X** which are regarding reading, writing and the execution of the files or directory. The permissions are assigned to three types 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 **(.)**


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]$

Acyclic-Graph Structured Directories

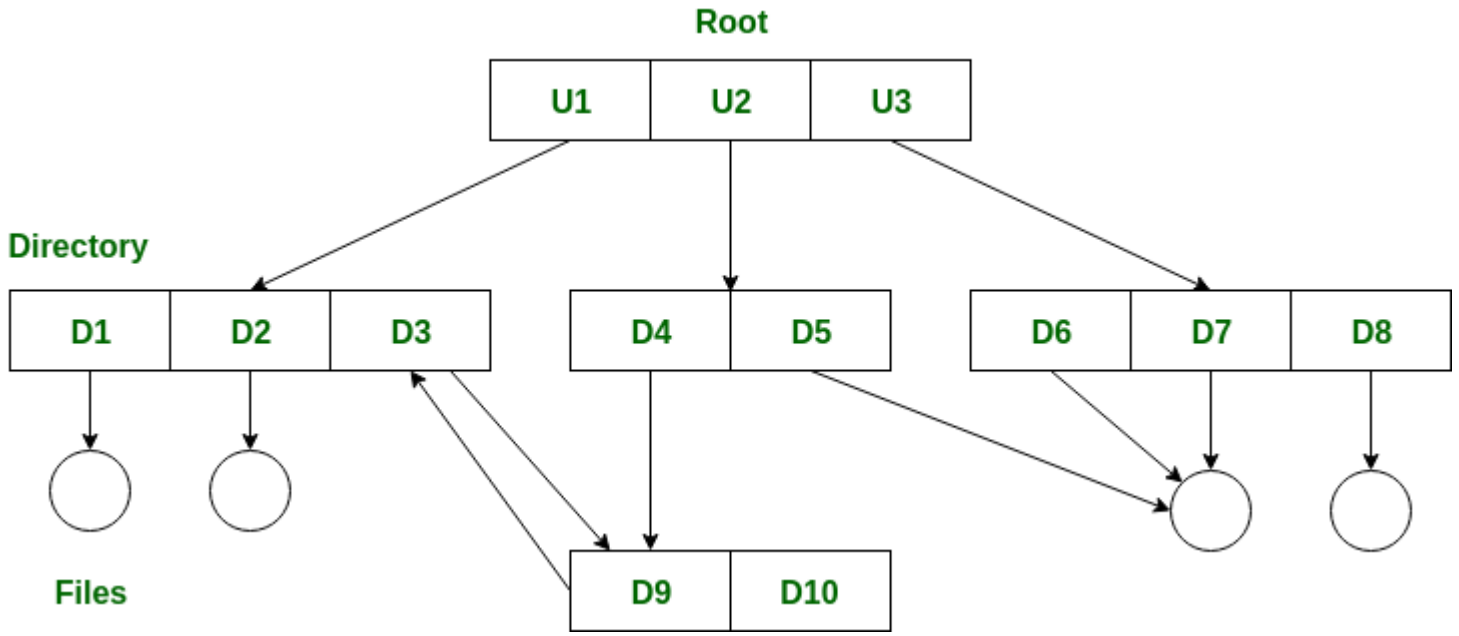
- The tree structured directory system doesn't allow the same file to exist in multiple directories therefore sharing is major concern in tree structured directory system. We can provide sharing by making the directory an acyclic graph. In this system, two or more directory entry can point to the same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

- It is the point to note that the shared file is not the same as the copy file. If any programmer makes some changes in the subdirectory it will reflect in both subdirectories.



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 the total size or space that has been taken by the files and directories.

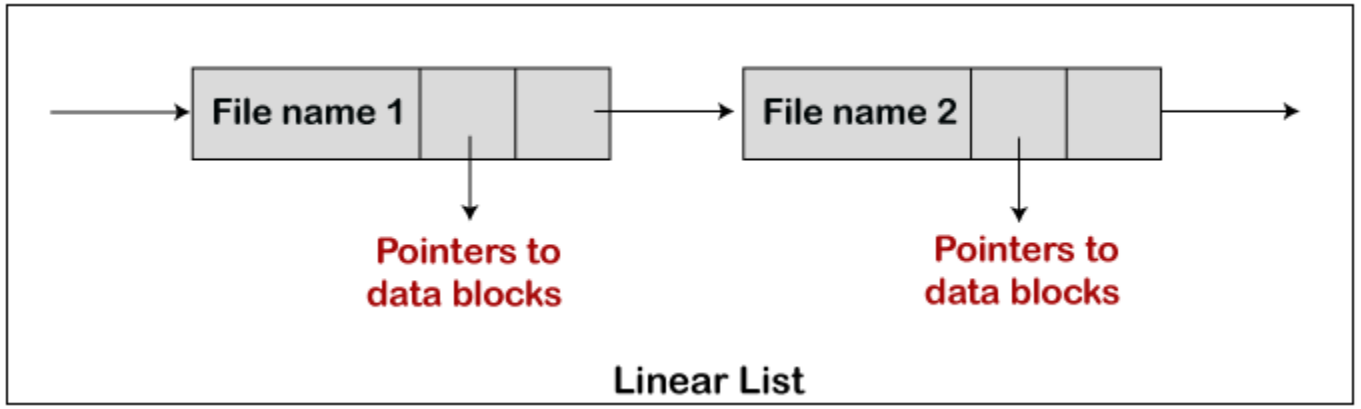


Directory Implementation

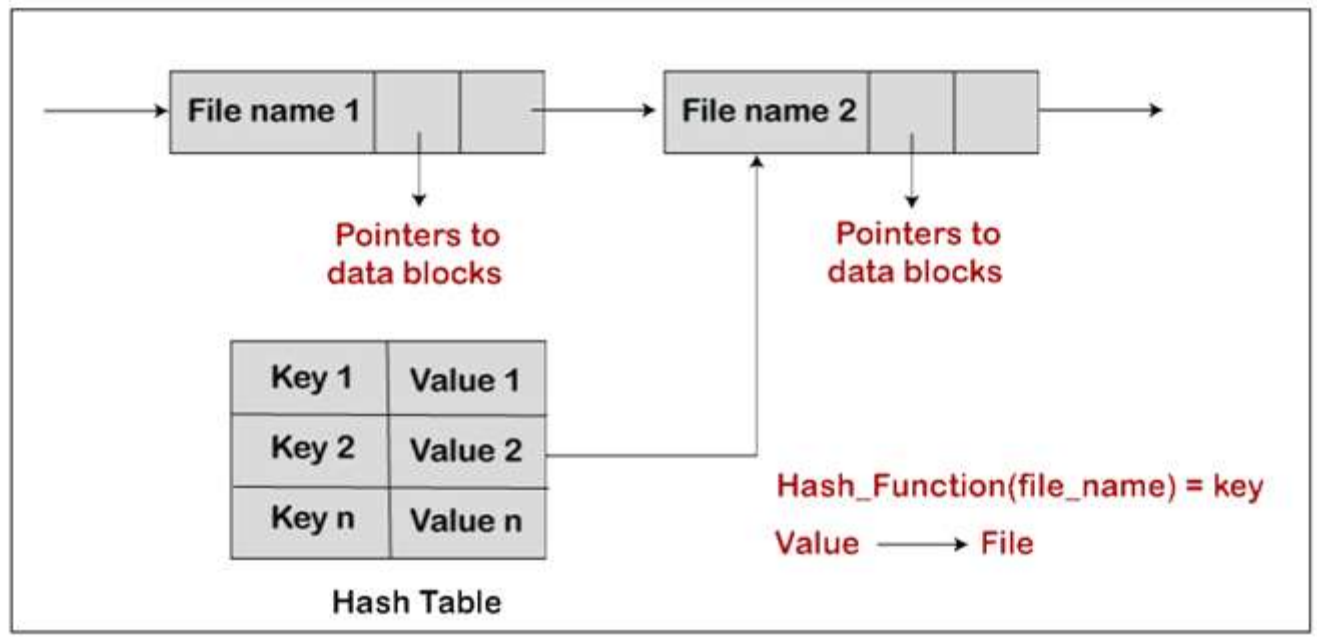
- There are various types of algorithm which we use for directory implementation. The selection of a suitable algorithm for directory implementation is an essential task because it directly affects system performance.
- We can classify the directory implementation algorithm based on the data structure.
- Mostly, we use two types of algorithms:
    1. Linear List
    2. Hash Table

Linear List

- **Linear List: -** The linear list is the most straightforward algorithm which is used for directory implementation. In this algorithm, we keep all the files in a directory like a singly linked list. Every file comprises of a pointer to the data blocks that are allocated to it and the next file in the directory.



- **Hash Table: -** There are some disadvantages in singly linked implementation of directories. So, to remove this drawback, we use another method that is called a hash table. In this method, the hash table is used with the linked list.
- In a directory, for every file, there is a key-value pair that is generated, and when the key-value pair is generated, then we store it into the hash table. With the help of the hash function on the file name, we can determine the key and key points to the respective file that are stored in a directory.
- In a linear list, the task of searching is difficult because, in a linear list, we have to search the entire list, but in hash table approach, there is no requirement of searching the entire list. So, in hash table searching is quite efficient. With the help of the key, we only have to check the entries of the hash table, and when we get the entry, then by using the value, we will fetch the corresponding file.


File name 1
File name 2
Pointers to data blocks
Pointers to data blocks
Key 1
Value 1
Key 2
Value 2
Key n
Value n
Hash Table
Hash_Function(file_name) = key
Value
File