 # Linux Filesystem Structure:
- While installation of Linux system we need to create two partitions:
1. "/" partition also called as **data partition**
2. swap partition
- Linux follows FHS i.e. **FileSystem Heirarchy Standards.**
- data can be store into the **"data partition"** in an organized manner, so linux filesystem structure is having **heirachical structure**, which is a like an inverted tree.
- when any user loggedin into the system/opens a terminal, system takes the user bydefualt into the user's home dir.
- linux filesystem structure starts with **root dir ("/")**, and all the data can be stored/kept inside it under sub-dir's in an organized manner.
- root dir i.e. "/" dir contains sub-dir's like "/boot", "/bin", "/sbin", "/etc", "/lib", "/usr", "/home", "/root" "/mnt", "/dev" etc....

1. "/boot": contains static files of the boot loader, kernel, initrd (initial ramdisk).

2. "/bin": contains user commands in binary format, like ls, cat, cp etc...

3. "/sbin": contains admin/system commands in binary format, like lscpu, adduser, deluser, etc...

4. "/home": contains home dir's of all user for multi-user system.
- for any user on its creation of an account bydefault subdir by the name of the user got created by the operating system, in which that user can store data, user can have read, write as well execute perms in that dir.

5. "/root": root is a home dir for root user

6. "/etc": contains host specific system configuration files - it is like a "control panel" in windows.

- in computing configuration files (or config files) are files used to configure the parameters and initial settings for some computer programs. They are used for user applications, server processes and operating system settings.
- e.g. **YaST: Yet another Setup Tool** (Linux operating system setup and configuration tool) or **debconf** (for performing system-wide configuration taskson UNIX-like operating systems).
"/etc/opt": contains configuration files for **add-on-packages** that are stored in /opt.

7. "/dev": contains device files

8. "/lib": contains shared libraries required for /bin and /sbin.

9. "/mnt": contains temporarily mounted filesystems

10. "/media": mount point for removable media such as CDROMs.

11. **"/opt":** contains optional application software packages.

12. "/proc": virtual filesystem provides process and kernel information as a files. In Linux , corresponds to a **procfs** mount. Generally automatically generated and populated by the system, on the fly.

 # path - it is a unique location of any file in a filesystem structure that can be represent in a string format i.e. in sequence of chars format seperated filenames/sub-dir names by **delimiter char** or **delimiter.**
e.g. "/home/sachin"
- there are three delimiter chars:
1. slash ("/") -- in linux
2. backslash ("\") -- in windows
3. colon (":") -- in linux as well as in windows
- there are two ways by which we can mention path of any file/dir:

1. absolute path: path of any file/dir with respect to root directory ("/") i.e. it is a full path of any file/dir which starts with root dir ("/").

2. relative path: path of any file/dir with respect to the current working dir.

- **UNIX consideres/treats everything as a file,** i.e. from UNIX point of veiw (as Linux is UNIX like/based OS) there are 7 types of files in UNIX/Linux:

1. regular file (-): linux treats all text files, source files, audio files vedio files, image files etc....

2. directory file (d): directory special file whose contents are name of files and sub dir's.

3. character special device file (c): devices from which data gets transferred character by character are called as character special devices e.g. KBD, Monitor, Printer, serial ports & parellel ports etc...

- information of device file gets loaded into "/dev" while booting.

4. block special device file (b): devices from which data gets transefered block by block are called as block devices, e.g. all storage devices are block devices

- OS treats all block devices as "block special device file".

5. socket file (s): this is special type of file can be used for an IPC.

6. named pipe file (p): this file can be used for an IPC

7. linkable file (l): this is special type of file which contains info about another file.

+ Command Name: "pwd" - print/present working directory

- this command displays absolute path (full path) of the present working directory

- pwd comamand internally refers the value of shell variable by the name "PWD".

+ Command Name: "cd" - to change current working directory
$cd <dirpath> -- change dir to the dirpath
$cd / -- goto the root directory -- "/" slash -- is user short hand variable
$cd ~ -- goto the user's home dir -- "~" tild symbol -- is user short hand variable
$cd -- goto the user's home dir
$cd - -- goto the prev directory
$cd . -- remains in current dir
$cd .. -- goto the parent dir of the current working dir

+ Command Name: "ls" - to display contents of the directory
- contents of the dir are nothing but name of files and sub-dir's.
$ls <dirpath> -- it display contents of the dir by the name "dirpath".
$ls -- bydefault it displays contents of the current directory
$ls -l -- it display contents of the dir in a listing format.

total n
n = total no. of data blocks allocated for the files and sub dir's in a given dir
feild-1: first char of feild 1 denotes type the file next 3 chars of field 1 denotes access perms for user/owner
next 3 chars of field 1 denotes access perms for group members last 3 chars of field 1 denotes access perms for group others.
Feild-2: denotes no. of links exists for that file for regular file bydefualt no. of links i.e. link count = 1
for directory file bydefualt no. of links i.e. link count = 2.
feild-3 : denotes name of an user/owner
feild-4 : denotes name of group
feild-5 : denotes name of size of the file
feild-6 : denotes time stamps - date of creation and last modified date & time in ISO format.

feild-7 : name of the file
- In UNIX/Linux filename starts with (".") are considered as a hidden file
- the period (".")

options/flags/args description:
-h : displays size of the files in human redable format
-l : displays contents of the dir in a listing format
-a : display all contents of the dir (including hidden files)
-A : display all contents of the dir (including hidden files but excluding entries for "." & "..").
-i : display inode number of the files inode number is an unique idenfier of the file.
etc..... explore more options from man pages

When we create any new file, first inode for that file gets created into the filesystem.
- when we create a new file it has 3 things:
1. inode
2. directory entry
3. data blocks (optional)

+ directory file (d):
- directory is a special file, whose contents are names of files and sub dir's.
- **directory is a container** which contains directory entries of files and sub dir's inside it.
- directory entry of a file is nothing but the link of that file.
- one file may has multiple dir entries but one file can have only one "inode".

- In a single filesystem we can have more than one files having same name, but we cannot have more than one files having same "inode number" i.e.

inode number is an unique identier of a file in that filesystem.
- We can have multiple directory entries (i.e. no. of links) for a same file at different locations.
- Directory file is a like a container whose contents are name of its files and subdirs/ directory file is a container whose contents are "dir entires".

- Directory entry of any file contains min two fields:

1. **inode number**
2. **name of the file**

- When we create a new file, filesystem assigns/creates/allocates three things for that file:

1. **inode** (compulsory)
2. **directory entry** - gets created in its parent dir location => link of that file (compulsory)
3. **data blocks** (optional)

- Directory entry is also reffered as "hard link" of that file.
- In Linux filename starts with (.) dot is reffered as **hidden file.**

- When we create a new dir, bydefault every dir contains two hidden entries:

. **--> current dir**
.. **--> parent dir**

- To maintain parent-child relationship between files **single dot (.) & two dots (..)** can be maintained in a filesystem.

- **"link count"** = no. of links exists for that file in a filesystem i.e. no. of ways which we can access the file.
- bydefualt link count any regular file = 1
- bydefualt link count any directory file = 2
(For a dir file -> 1 dir entry gets created inside its parent dir location & one hidden dir

entry gets created inside that dir itself by the name single dot (.)).

- Command to execute a program w.r.t. its parent dir -> **$./program.out**