COMP311
Linux OS Laboratory
Lab3:File Systems (I)
(Structure and File Types)

By

Alaa’ Omar

Objectives

1

Understand the structure of the Linux file system.

2

Build tree structures using absolute and relative paths.

3

Recognize and create the different main Linux file types.

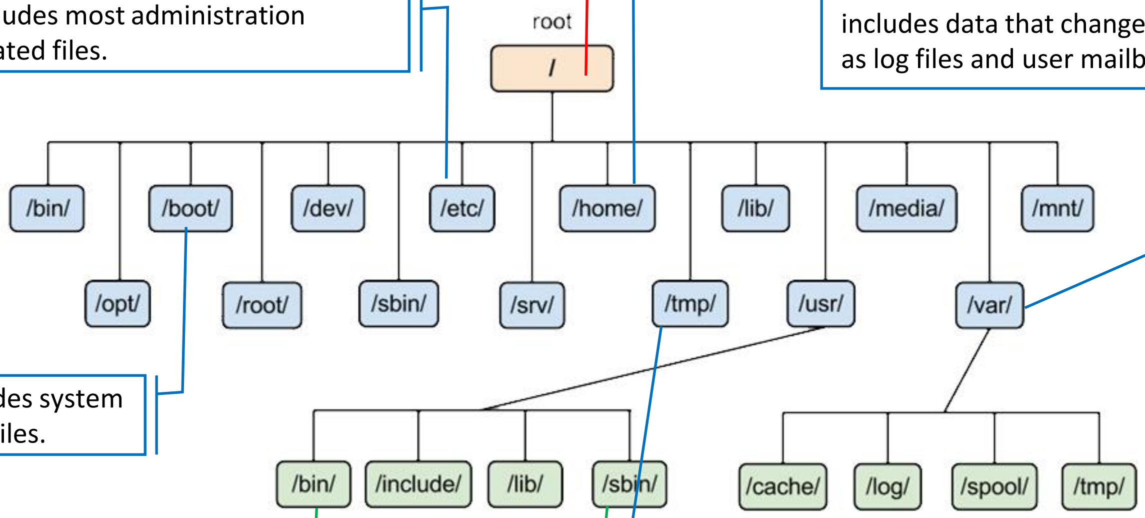
The Linux File system tree

/ is the root directory at the top of the file system hierarchy. It includes the root directory and a minimal set of files and subdirectories.

includes most administration related files.

includes user home directories

includes data that changes such as log files and user mailboxes.

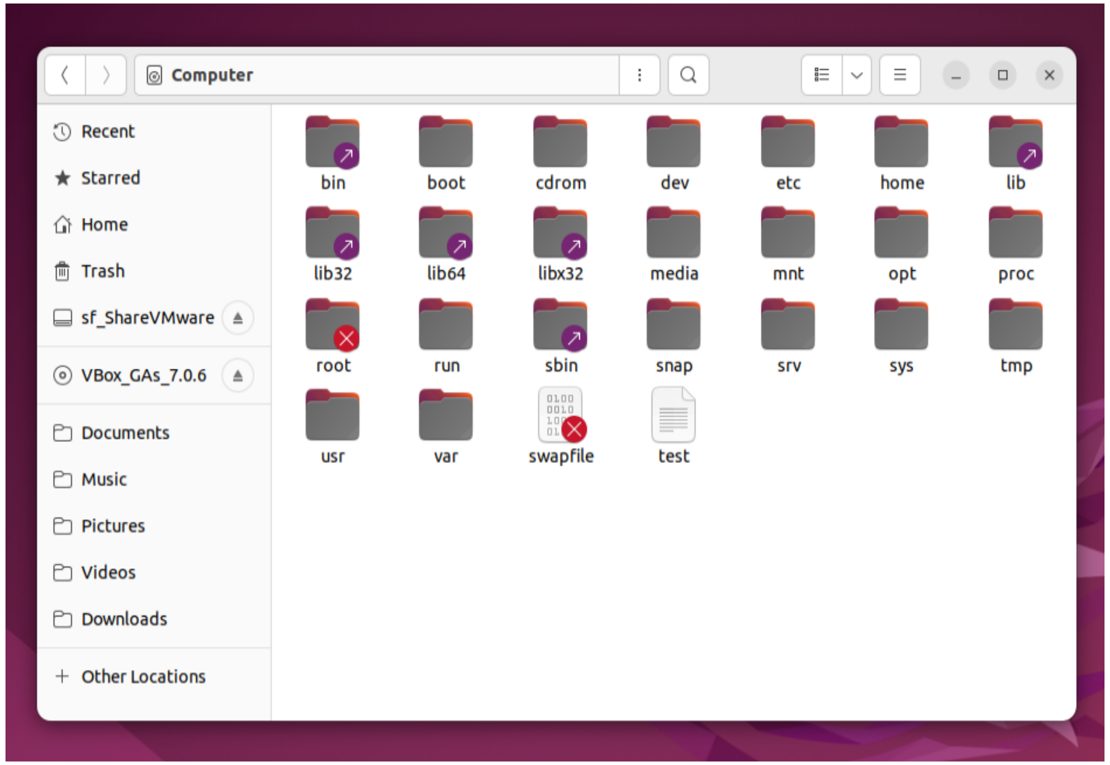


includes system boot files.

most commands and executable files

Less essential commands for administration repair

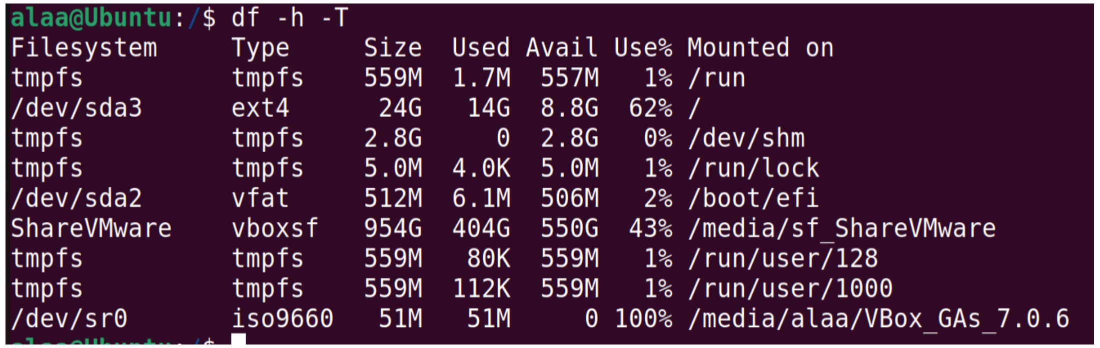
For temporary files.



File system disk usage commands

Type **man df**

- **df - report file system disk space usage**
  - **df -h: human readable format**
  - **df -T: add Type column**

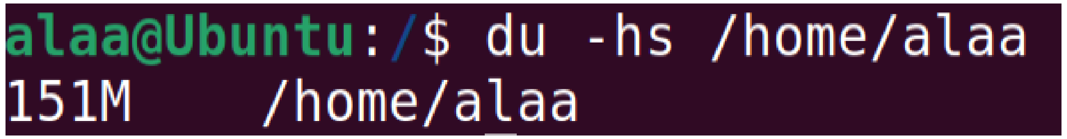


```
alaa@Ubuntu:/$ df -h -T
Filesystem     Type      Size  Used Avail Use% Mounted on
tmpfs          tmpfs     559M  1.7M  557M   1% /run
/dev/sda3      ext4       24G   14G  8.8G  62% /
tmpfs          tmpfs     2.8G     0  2.8G   0% /dev/shm
tmpfs          tmpfs     5.0M  4.0K  5.0M   1% /run/lock
/dev/sda2      vfat      512M  6.1M  506M   2% /boot/efi
ShareVMware    vboxsf    954G  404G  550G  43% /media/sf_ShareVMware
tmpfs          tmpfs     559M   80K  559M   1% /run/user/128
tmpfs          tmpfs     559M  112K  559M   1% /run/user/1000
/dev/sr0       iso9660    51M   51M     0 100% /media/alaa/VBox_GAs_7.0.6
```

- **du - estimate file space usage**
  - **du -h: human readable format**
  - **du -s: summary**

  - **Example: du -h /home/u145445**

    du -hs /home/u145445



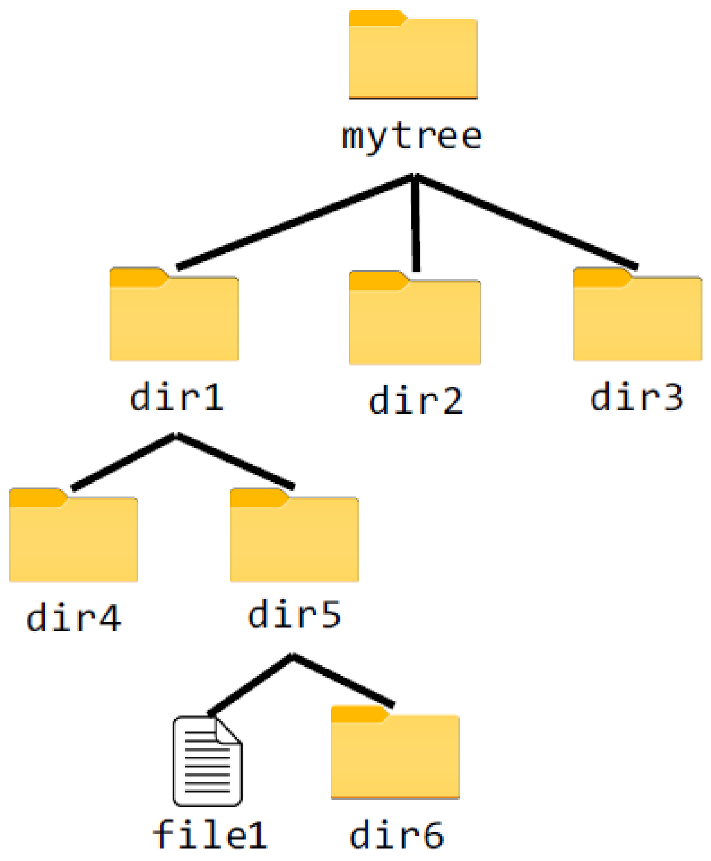
```
alaa@Ubuntu:/$ du -hs /home/alaa
151M    /home/alaa
```

File system commands

To Manipulate directories under a file system:

- **mkdir newdir** (creates a new directory called **newdir**)
- **cd newdir** (changes your position to **newdir**)
- **rmdir newdir** (removes directory new directory only if **newdir** is empty)
- **rm –rf newdir** (removes non-empty or empty directory **newdir**)
- **pwd** (displays present working directory)
- **touch** newfile Use the **touch** command to create an empty file. If a file

by the name you specify doesn't already exist, the **touch** command creates an empty file (if the file already exists, **touch** updates the last file access time).

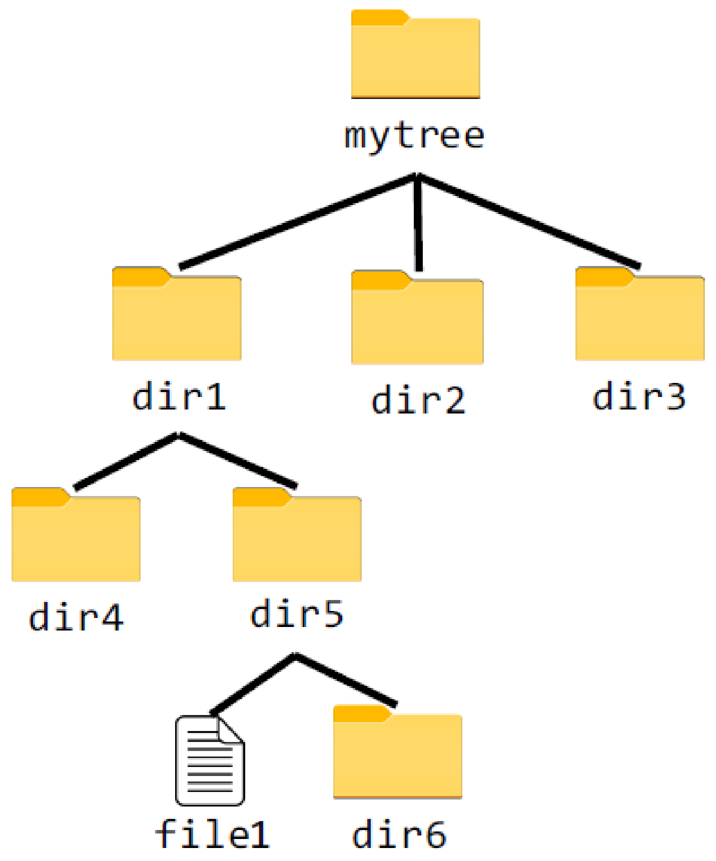


Practice building mytree

- Build mytree using mkdir and cd commands:

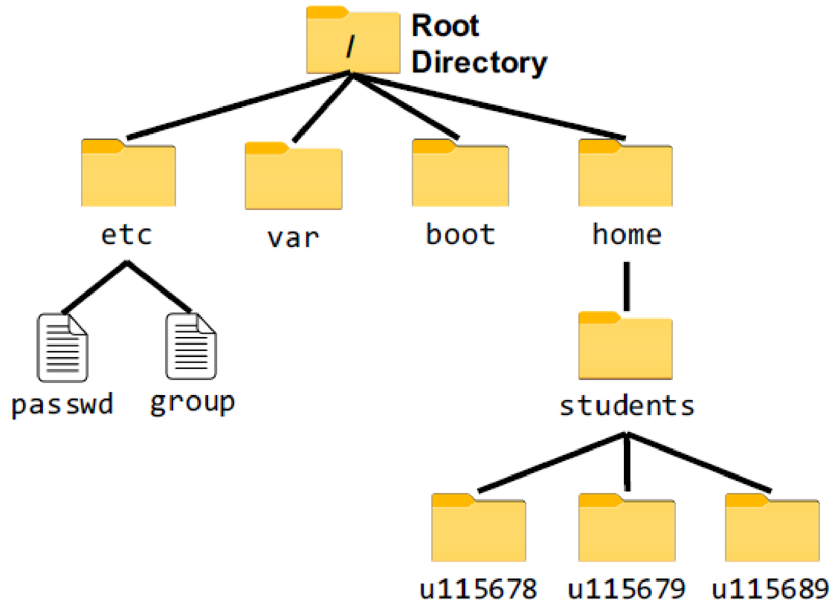
1. mkdir mytree
2. cd mytree
3. mkdir dir1 dir2 dir3
4. cd dir1
5. mkdir dir4 dir5
6. cd dir5
7. mkdir dir6
8. touch file1

To display mytree use the command (ls –R mytree)



Absolute vs relative paths

Absolute path: the exact address of the file of the directory **starting form the root directory '/'**

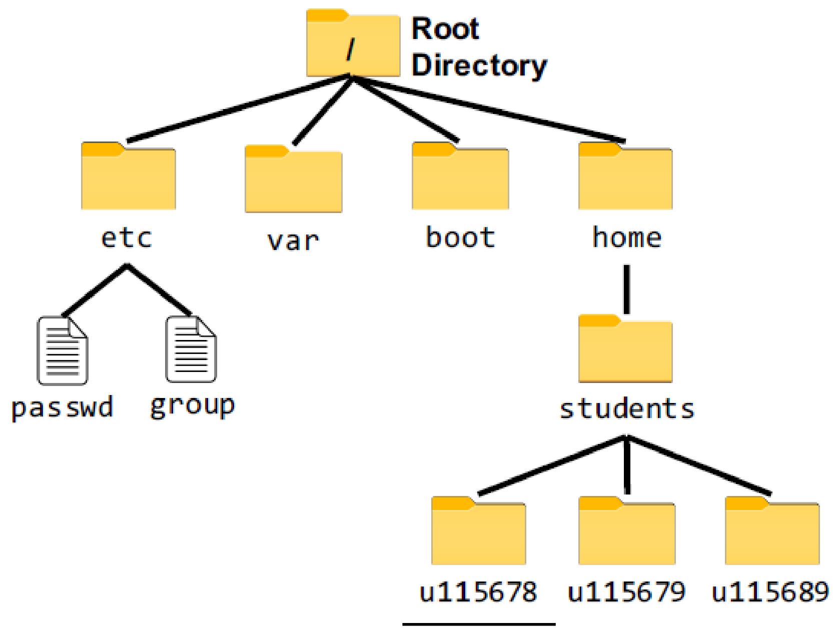


Relative path: the path of the file or the directory **based on the current working directory**

Any file can be referenced by its absolute or relative path name.
Each file has only one absolute path name while it may have an infinite number of relative paths.

Special Path

Path	Explanation
~	User’s home directory
	User’s home directory
/	Root Directory
..	Parent Directory
.	Current Directory
../..	Tow level parent directory



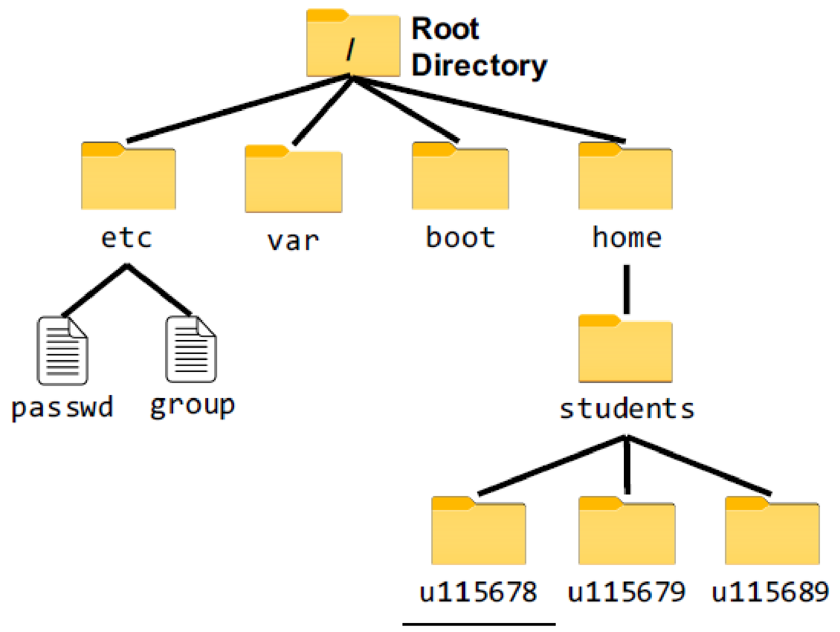
Absolute Path

```
cd ( change path)
$ cd path
```

```
$ cd /home/students/u115678
```

```
$ cd /home/students/u115679
```

```
$ cd /etc
```



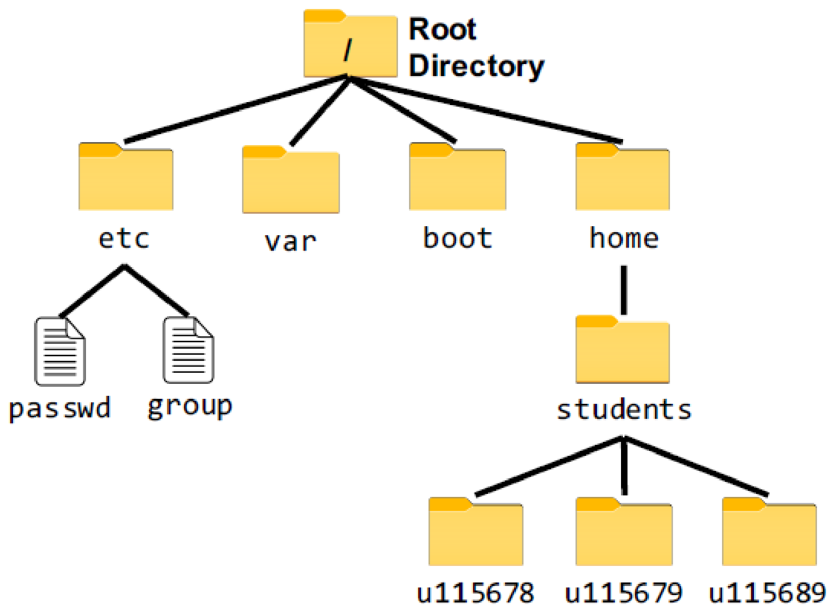
Relative path

```
pwd (print current directory)
$ cd path
```

```
$ cd students/u115678
```

```
$ cd ../home
```

```
$ cd  ../../etc
```



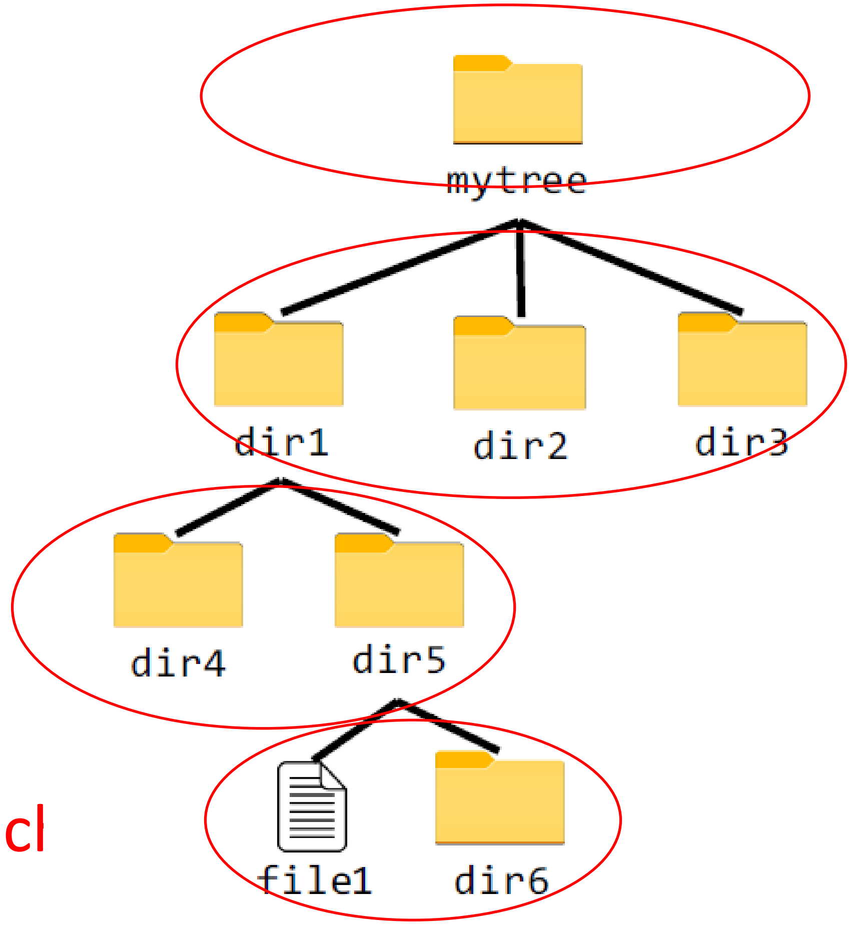
Practise building using relative path (1)

1. mkdir mytree
2. mkdir mytree/dir1 mytree/dir2 mytree/dir3
3. mkdir mytree/dir1/dir4
4. mkdir mytree/dir1/dir5
5. mkdir mytree/dir1/dir5/dir6

- rm -rf mytree

Try mkdir mytree/dir1 mytree/dir2 mytree/dir3

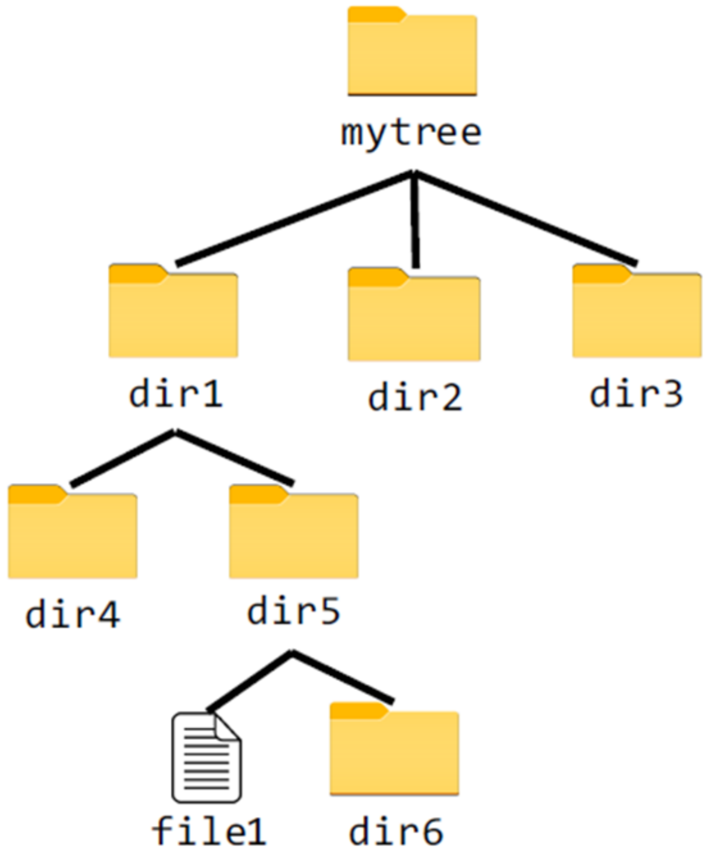
- mkdir -p mytree/dir1 mytree/dir2 mytree/dir3
- mkdir -p mytree/{dir2,dir3,dir1/{dir4,dir5/dir6}};toucl mytree/dir1/dir5/dir6/file1



Create directories using relative path (2)

Construct The previous tree (mytree) using relative path method in one line ?

mkdir –p mytree/dir1/../dir2/../dir3/../dir1/dir4/../dir5/dir6;
touch mytree/dir1/dir5/file1



Linux files types

- Regular files: Regular files consist of a series of bytes; Text files, data files, executable programs, and shared libraries are all stored as regular files. created using vi and touch commands.

- Directories: A directory contains named references to other files. You can create directories with **mkdir** and delete them with **rmdir** if they are empty. You can delete nonempty directories with **rm -r**.

  The special entries "**.**" and "**..**" refer to the directory itself and to its parent directory; they may not be removed. Since the root directory has no parent directory, the path "**/..**" is equivalent to the path "**/.**" (and both are equivalent to **/**).

- Character and block device files

Device files let programs communicate with the system's hardware and peripherals. The kernel includes (or loads) driver software for each of the system's devices. Character device files allow their associated drivers to do their own input and output buffering. Block device files are used by drivers that handle I/O in large chunks and want the kernel to perform buffering for them

Linux files types

- Symbolic links: A symbolic or “soft” link points to a file by name. When the kernel comes upon a symbolic link in the course of looking up a pathname, it redirects its attention to the pathname stored as the contents of the link.

The original links in Linux are called the hard links and are created using the command ln

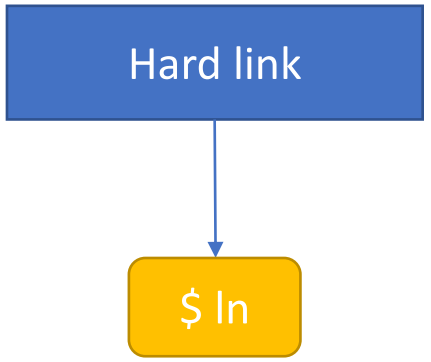
The difference between hard links and symbolic links is that a hard link is a direct reference, whereas a symbolic link is a reference by name. Symbolic links are distinct from the files they point to

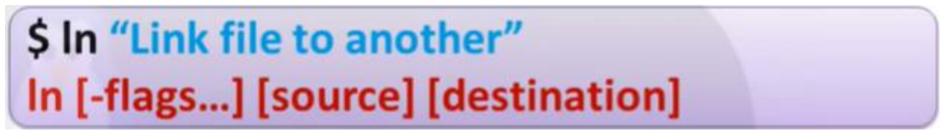
ln : make links between files

Practice : Lab manual page 14

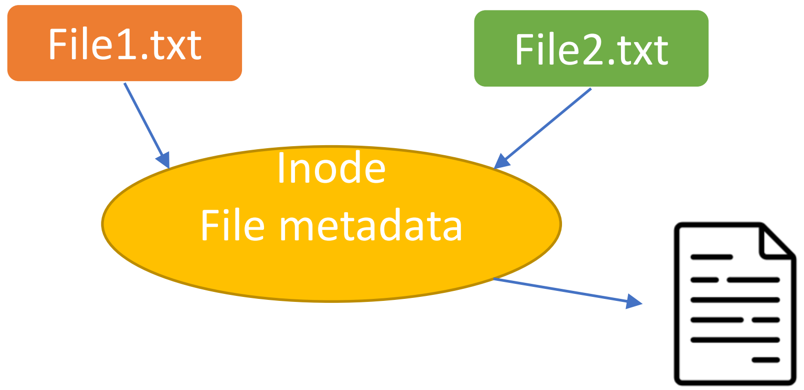
Hard link vs Soft link

Associate two files to the same Inode

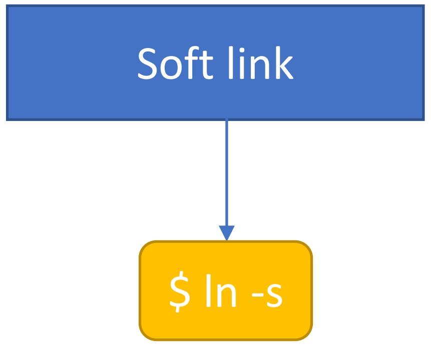


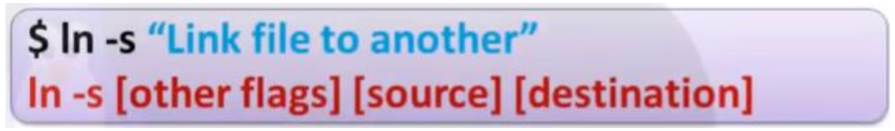


```
$ ln "Link file to another"
ln [-flags...] [source] [destination]
```

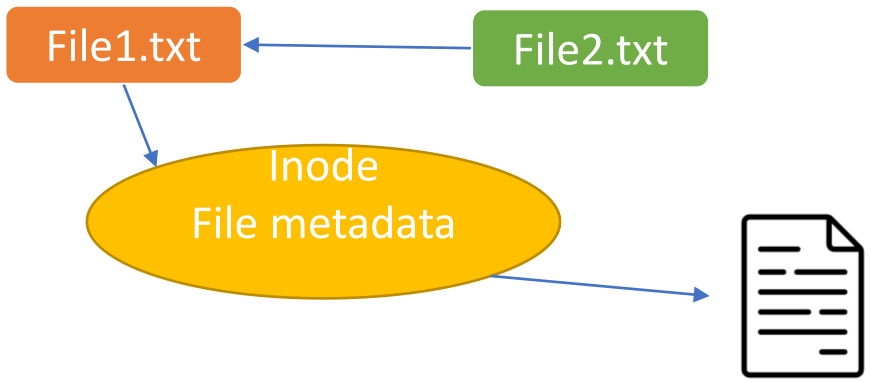


File pointing to another file "short-cut in windows"





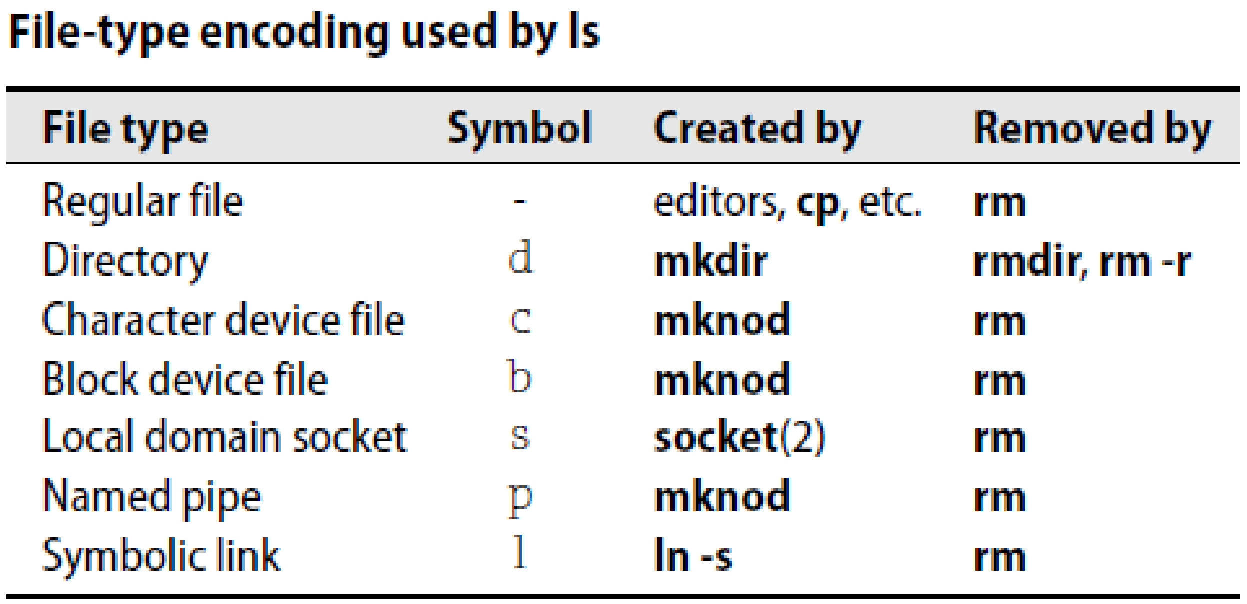
```
$ ln -s "Link file to another"
ln -s [other flags] [source] [destination]
```



Soft Link

- Syntax (ln -s srcpath destpath)
- The srcpath is how do you reach the source from destination
- The destpath is the path from where you are for example home

File-types summary



File-type encoding used by ls

File type	Symbol	Created by	Removed by
Regular file	-	editors, **cp**, etc.	**rm**
Directory	d	**mkdir**	**rmdir, rm -r**
Character device file	c	**mknod**	**rm**
Block device file	b	**mknod**	**rm**
Local domain socket	s	**socket**(2)	**rm**
Named pipe	p	**mknod**	**rm**
Symbolic link	l	**ln -s**	**rm**