`LAB 1 .md`

Objective: Practice essential Linux commands.

Linux Basic Commands

LAB 3

1. Navigation Commands

`pwd` – Print Working Directory

-Shows the current location in the filesystem.

```
pwd
```

📌 Output example:

`/Users/yourname/projects`

`ls` – List Directory Contents

-Lists files and folders in the current directory.

```
ls
```

- `ls -l` → Detailed list (permissions, size, date)
- `ls -a` → Shows hidden files (those starting with `.`)
- `ls -la` → Combined

`cd` – Change Directory

-Moves into a directory.

```
cd folder_name
```

Examples:

```
cd Documents        # Go to Documents
cd ..               # Go up one level
cd /                # Go to root
cd ~                # Go to home directory
```

---

2. File and Directory Management

`mkdir` – Make Directory

-Creates a new folder.

```
mkdir new_folder
```

`touch` – Create File

-Creates an empty file.

```
touch file.txt
```

`cp` – Copy Files or Directories

```
cp source.txt destination.txt
```

- Copy folder:

```
cp -r folder1 folder2
```

`mv` – Move or Rename Files

```
mv oldname.txt newname.txt
```

```
mv file.txt ~/Documents/     # Move file
```

`rm` – Remove Files

```
rm file.txt          # Delete file
rm -r folder_name    # Delete folder (recursively)
```

⚠️ **Be careful!** There is no undo.

---

3. File Viewing & Editing

`cat` – View File Contents

-Displays content in terminal.

```
cat file.txt
```

`nano` – Edit Files in Terminal

-A basic terminal-based text editor.

```
nano file.txt
```

- Use arrows to move
- `CTRL + O` to save
- `CTRL + X` to exit

`clear` – Clears the Terminal

```
clear
```

Shortcut: `CTRL + L`

---

4. System Commands

`echo` – Print Text

Useful for debugging or scripting.

```
echo "Hello, World!"
```

`whoami` – Show Current User

```
whoami
```

`man` – Manual for Any Command

```
man ls
```

Use `q` to quit the manual.

---

5. Searching and Finding

`find` – Locate Files

```
find . -name "*.txt"
```

🔍 Finds all `.txt` files in current folder and subfolders.

`grep` – Search Inside Files

```
grep "hello" file.txt
```

🔍 Searches for the word `hello` inside `file.txt`.

LAB 5

1. Basics of Permissions

Every file or directory in Linux has three categories of users:

- **Owner (User)** → The person who created the file.
- **Group** → Users grouped together with shared access.
- **Others** → All remaining users on the system.

Types of Permissions

- `r` → **Read** (numeric value = 4)
- `w` → **Write** (numeric value = 2)
- `x` → **Execute** (numeric value = 1)

Permission String Example

From `ls -l` you might see:

drwxr-xr--

Breakdown:

- `d` → This is a directory (`-` means regular file).
- `rwx` → Owner has read, write, and execute rights.
- `r-x` → Group can read and execute.
- `r--` → Others can only read.

---

2. Using `chmod` (Change Mode)

General Syntax

```
chmod [flags] mode filename
Permissions can be changed in octal (numeric) or symbolic form.
(A) Octal (Numeric) Form
Each permission has a number:
Permission  Value
Read    4
Write   2
Execute 1
Combine values:
7 = rwx
6 = rw-
5 = r-x
4 = r--
```

```
Example:
chmod 644 notes.txt
Owner → rw- (read + write)
Group → r-- (read only)
Others → r-- (read only)
```

```
(B) Symbolic Form
Characters used:
u (user), g (group), o (others), a (all).
Operators: + add, - remove, = set exactly.
Examples:
chmod u+x run.sh       # allow owner to execute
chmod g-w data.log     # remove write for group
chmod o=r file.txt     # others can only read
chmod a+rw project.md  # everyone can read & write
```

```
(C) Recursive Permission Change
chmod -R 755 myfolder
-R → applies permissions to all subdirectories and files inside.
```

3. Using chown (Change Ownership)

```
Syntax
chown [flags] new_user:new_group filename
Examples:
chown ritsika file.txt         # make 'ritsika' the owner
chown ritsika:staff file.txt   # owner = ritsika, group = staff
chown :staff file.txt        # only change group
chown -R root:admin /var/www # apply recursively
```

4. Example Workflow

```
touch sample.sh
ls -l sample.sh
Output:
-rw-r--r-- 1 ankit staff 0 Aug 25 09:00 sample.sh
Now apply changes:
chmod 700 sample.sh         # full access for owner only
chmod u+x,g-w sample.sh     # add execute for user, remove write from group
chown root:admin sample.sh # change ownership to root:admin
```

5. Quick Reference Table

Number Permission Meaning 0 --- No access 1 --x Execute only 2 -w- Write only 3 -wx Write + Exec 4 r-- Read only 5 r-x Read + Exec 6 rw- Read + Write 7 rwx Full access

✅ Tip: Use numbers (e.g., 755, 644) when you know the exact permission combo, and symbolic form (u+x, g-w) when you want fine control

alt text

Extra Questions:

What is the difference between chmod and chown?

chmod → change file permissions ~Controls who can read, write, or execute a file. → Gives owner full rights (read/write/execute), others can read & execute only. chown → change file owner ~Changes who owns a file or directory (the user and group). → Makes user1 the owner of file.txt.

How do you check current directory and user?

Check current directory → use pwd Check current user → use whoami