Lab 2: Demonstration of Basic Linux Commands

1. Introduction to Linux Commands

Linux commands are case-sensitive and executed in a terminal (command-line interface). They follow the general syntax:

```
command [options] [arguments]
```

- **Command:** The operation to perform (e.g., `ls`, `cd`).
- **Options:** Modify command behavior (e.g., `-l`, `-a`).
- **Arguments:** Files/directories the command acts upon.

---

2. File System Navigation Commands

a) `pwd` (Print Working Directory)

- **Purpose:** Displays the absolute path of the current working directory.
- **Use Case:** Helps identify where commands will execute.

Example:

```
$ pwd
```

Output:

```
/home/student
```

(This means the user is currently in the `/home/student` directory.)

---

b) `ls` (List Directory Contents)

- **Purpose:** Lists files and subdirectories in a directory.
- **Common Options:**

Option	Description	Example
`-l`	Long format (permissions, owner, size, modification time)	`ls -l`
`-a`	Shows hidden files (starting with `.`)	`ls -a`
`-h`	Human-readable file sizes (KB, MB, GB)	`ls -lh`
`-t`	Sorts by modification time (newest first)	`ls -lt`

Example:

```
$ ls -la
```

Output:

```
total 24
drwxr-xr-x  3 student student 4096 Jun 25 10:00 .
```

```
drwxr-xr-x  5 root    root    4096 Jun 24 09:00 ..
-rw-r--r--  1 student student  220 Jun 25 09:30 .bashrc
drwxr-xr-x  2 student student 4096 Jun 25 10:00 Documents
```

(Shows all files, including hidden ones, with detailed permissions.)

---

c) `cd` (Change Directory)

- **Purpose:** Moves between directories.
- **Special Symbols:**
  - `~` = Home directory
  - `..` = Parent directory
  - `-` = Previous directory

Examples:

```
$ cd /home/student/Documents  # Move to Documents
$ cd ..                      # Move back to /home/student
$ cd ~                       # Return to home directory
$ cd -                       # Switch to the last visited directory
```

---

3. File Manipulation Commands

a) `touch` (Create Empty File)

- **Purpose:** Creates a new empty file or updates the timestamp of an existing file.

Example:

```
$ touch notes.txt
```

(Creates `notes.txt` if it doesn’t exist.)

Verification:

```
$ ls
notes.txt
```

---

b) `mkdir` (Make Directory)

- **Purpose:** Creates a new directory.
- **Useful Option:**
  - `-p` = Creates parent directories if they don’t exist.

Example:

```
$ mkdir projects
$ mkdir -p projects/code/python  # Creates nested directories
```

Verification:

```
$ ls projects
code
```

---

c) `cp` (Copy Files/Directories)

- **Purpose:** Copies files or directories.
- **Key Options:**
  - `-r` = Recursive copy (for directories)
  - `-i` = Interactive (prompts before overwriting)

Examples:

```
$ cp notes.txt notes_backup.txt      # Copy file
$ cp -r projects projects_backup    # Copy directory
```

---

d) `mv` (Move/Rename Files/Directories)

- **Purpose:** Moves files/directories or renames them.

Examples:

```
$ mv notes.txt Documents/           # Move file
$ mv oldname.txt newname.txt        # Rename file
```

---

e) `rm` (Remove Files/Directories)

- **Purpose:** Deletes files/directories permanently.
- **Key Options:**
  - `-r` = Recursive deletion (for directories)
  - `-f` = Force deletion (no confirmation)

 Caution: `rm -rf` is irreversible!

Examples:

```
$ rm notes.txt                      # Delete file
$ rm -rf old_project/               # Delete directory
```

---

4. File Viewing & Editing Commands

a) `cat` (Concatenate & Display File Content)

- **Purpose:** Displays entire file content at once.
- **Best For:** Small files.

Example (`sample.txt`):

```
Line 1: Hello
Line 2: This is a sample file.
Line 3: Goodbye!
```

Command:

```
$ cat sample.txt
```

Output:

```
Line 1: Hello
Line 2: This is a sample file.
Line 3: Goodbye!
```

---

b) `less` / `more` (View Large Files Page by Page)

- **Purpose:** Displays large files interactively.
- **Difference:**
  - `less` allows backward/forward navigation.
  - `more` only moves forward.

Example (`large_file.log`):
(Assume this file has 1000+ lines of logs.)

Command:

```
$ less large_file.log
```

Navigation:

- `Space` = Next page
- `b` = Previous page
- `/search_term` = Search
- `q` = Quit

---

c) `head` / `tail` (View File Start/End)

- **Purpose:**
  - `head` = Shows first 10 lines (default).
  - `tail` = Shows last 10 lines.
- **Useful Options:**
  - `-n X` = Show X lines (e.g., `head -n 5`).
  - `-f` = Follow updates in real-time (`tail -f`).

Example (`server.log`):

```
[2024-06-25 09:00] Server started
[2024-06-25 09:05] User 'admin' logged in
[2024-06-25 09:10] Warning: Disk 80% full
[2024-06-25 09:15] Backup completed
[2024-06-25 09:20] Error: Connection timeout
```

Commands:

```
$ head -n 3 server.log
$ tail -n 2 server.log
```

Outputs:

```
[2024-06-25 09:00] Server started
[2024-06-25 09:05] User 'admin' logged in
[2024-06-25 09:10] Warning: Disk 80% full
```

```
[2024-06-25 09:15] Backup completed
[2024-06-25 09:20] Error: Connection timeout
```

---

5. System Monitoring Commands

a) `df` (Disk Free Space)

- **Purpose:** Shows disk usage for all mounted filesystems.
- **Key Option:** `-h` (human-readable format).

Example:

```
$ df -h
```

Output:

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        50G   20G   28G  42% /
tmpfs           1.9G     0  1.9G   0% /tmp
```

(Indicates `/dev/sda1` uses 42% of its 50GB capacity.)

---

b) `free` (Memory Usage)

- **Purpose:** Displays RAM and swap usage.
- **Key Option:** `-h` (human-readable).

Example:

```
$ free -h
```

Output:

```
              total        used        free
Mem:           7.7G        3.2G        4.5G
Swap:          2.0G        0.5G        1.5G
```

(Shows 3.2GB of 7.7GB RAM is used.)

---

6. Process Management Commands

a) `ps` (Process Status)

- **Purpose:** Lists running processes.
- **Common Usage:** `ps aux` (shows all processes).

Example:

```
$ ps aux | grep "nginx"
```

Output:

```
root      1234  0.0  0.5  50000  8000 ?        Ss   10:00   0:01 nginx: master
www-data  1235  0.0  0.3  30000  5000 ?        S    10:01   0:00 nginx: worker
```

(Shows Nginx processes.)

---

b) `kill` (Terminate Process)

- **Purpose:** Stops a process using its PID.
- **Common Signal:** `-9` (forceful termination).

Example:

```
$ kill -9 1234
```

(Force-kills process with PID `1234`.)

---

7. Summary Table: Key Commands

Command	Purpose	Example
`pwd`	Show current directory	`pwd`
`ls -l`	List files with details	`ls -l /home`
`cp -r`	Copy directories	`cp -r dir1 dir2`
`tail -f`	Monitor log updates	`tail -f /var/log/syslog`
`df -h`	Check disk space	`df -h`
`kill -9`	Force-stop process	`kill -9 1234`

---

8. Practical Exam Tips

1. **File Navigation:** Use `cd`, `ls`, `pwd` to explore directories.
2. **File Operations:** Practice `cp`, `mv`, `rm` (carefully!).
3. **Log Inspection:** Use `less`, `tail -f` for log analysis.
4. **Process Control:** Use `ps`, `kill` to manage tasks.