Name – Harshit

MIS – 112316018

Branch – ECE

OS lab Assignment 1

**Basic Unix Commands:**

**1. File and Directory Management**

1. **ls** - Lists the contents of a directory, including files and subdirectories.  
   Example: ls -l shows detailed information like permissions, owner, and size.
2. **cd** - Changes the current working directory.  
   Example: cd /var/log navigates to the /var/log directory.
3. **pwd** - Prints the absolute path of the current directory.  
   Example: Useful to verify your current working location.
4. **mkdir** - Creates a new directory.  
   Example: mkdir project creates a folder named project.
5. **rmdir** - Removes an empty directory.  
   Example: rmdir old\_folder deletes the folder if it's empty.
6. **rm** - Deletes files or directories.  
   Example: rm file.txt removes the file; rm -r folder removes a directory recursively.
7. **cp** - Copies files or directories.  
   Example: cp source.txt dest.txt copies a file.
8. **mv** - Moves or renames files or directories.  
   Example: mv old.txt new.txt renames a file.
9. **touch** - Creates an empty file or updates the timestamp of an existing file.  
   Example: touch newfile.txt.
10. **find** - Searches for files or directories matching a pattern.  
    Example: find / -name "\*.log" searches for all .log files.

**2. File Permissions and Ownership**

1. **chmod** - Changes file permissions.  
   Example: chmod 755 script.sh makes the file executable for the owner and readable for others.
2. **chown** - Changes file ownership.  
   Example: chown user:group file.txt changes the owner and group of the file.
3. **chgrp** - Changes the group ownership of a file.  
   Example: chgrp developers file.txt.
4. **umask** - Sets default permissions for newly created files.  
   Example: umask 022 ensures new files are readable by everyone.
5. **stat** - Displays detailed information about a file (size, modification date, etc.).  
   Example: stat file.txt.

**3. Disk and Filesystem Management**

1. **df** - Displays available and used disk space on filesystems.  
   Example: df -h shows disk usage in human-readable format.
2. **du** - Shows the size of directories and their contents.  
   Example: du -sh /var provides a summary of /var directory size.
3. **mount** - Attaches a filesystem to a directory.  
   Example: mount /dev/sdb1 /mnt mounts a partition to /mnt.
4. **umount** - Detaches a mounted filesystem.  
   Example: umount /mnt.
5. **fsck** - Checks and repairs filesystem errors.  
   Example: fsck /dev/sda1.
6. **blkid** - Displays UUID and other metadata of block devices.  
   Example: blkid /dev/sda1.
7. **mkfs** - Creates a new filesystem on a partition.  
   Example: mkfs.ext4 /dev/sdb1.
8. **tune2fs** - Adjusts filesystem parameters.  
   Example: tune2fs -c 20 /dev/sda1 sets maximum mount count before a filesystem check.
9. **lsblk** - Displays information about block devices and partitions.  
   Example: lsblk.
10. **parted** - Manages partitions on a disk.  
    Example: parted /dev/sda.

**4. Process Management**

1. **ps** - Displays information about active processes.  
   Example: ps aux lists all running processes with details.
2. **top** - Shows a real-time view of system processes.  
   Example: Use to monitor CPU and memory usage.
3. **htop** - Interactive process viewer, similar to top, with an enhanced interface.
4. **kill** - Terminates a process by its PID.  
   Example: kill 1234.
5. **killall** - Terminates processes by name.  
   Example: killall apache2.
6. **nice** - Starts a process with a specific priority.  
   Example: nice -n 10 ./my\_program.
7. **renice** - Changes the priority of a running process.  
   Example: renice 5 -p 1234.
8. **jobs** - Lists background jobs.  
   Example: jobs.
9. **fg** - Brings a background job to the foreground.  
   Example: fg %1.
10. **bg** - Resumes a paused job in the background.  
    Example: bg %1.

**5. System Information**

1. **uname** - Displays system information (OS, kernel, architecture).  
   Example: uname -a.
2. **uptime** - Shows how long the system has been running.  
   Example: uptime.
3. **who** - Displays users currently logged into the system.  
   Example: who.
4. **w** - Displays logged-in users and their activities.  
   Example: w.
5. **hostname** - Displays or sets the system hostname.  
   Example: hostname.
6. **vmstat** - Displays CPU, memory, and IO statistics.  
   Example: vmstat 5.
7. **iostat** - Shows CPU and I/O performance.  
   Example: iostat.
8. **free** - Displays memory usage.  
   Example: free -h.
9. **dmesg** - Prints kernel messages (e.g., hardware or boot info).  
   Example: dmesg | tail.
10. **lscpu** - Displays detailed CPU architecture info.  
    Example: lscpu.

**6. User and Group Management**

1. **id** - Displays the current user's ID and group IDs.  
   Example: id.
2. **whoami** - Displays the current username.  
   Example: whoami.
3. **groups** - Lists groups the current user belongs to.  
   Example: groups.
4. **passwd** - Changes the user's password.  
   Example: passwd.
5. **useradd** - Adds a new user to the system.  
   Example: sudo useradd new\_user.

**Unix Editors:**

**1. vi (Visual Editor)**

* **Overview**:
  + A screen-based text editor that comes standard on most Unix systems.
  + It’s lightweight, efficient, and great for quick edits.
* **Modes**:
  + **Command Mode**: For issuing commands like saving or exiting.
  + **Insert Mode**: For typing and editing text.
* **Key Commands**:
  + i: Enter insert mode.
  + :w: Save changes.
  + :q: Quit.
  + :q!: Quit without saving.
  + :wq: Save and quit.
* **Why Use It?**
  + Available on almost all Unix systems by default.
  + Ideal for quick, terminal-based text editing.

**2. ed (Line Editor)**

* **Overview**:
  + A line-based editor, one of the earliest text editors in Unix.
  + Works on one line of text at a time, without a screen-based interface.
* **Usage**:
  + Edit text files in a sequential and precise way.
  + Input commands rather than directly editing text visually.
* **Key Commands**:
  + a: Append text after the current line.
  + d: Delete the current line.
  + w: Write changes to the file.
  + q: Quit the editor.
* **Why Use It?**
  + Lightweight and minimal.
  + Useful for scripting or on systems with very limited resources.

**3. emacs (Extensible, Customizable Editor)**

* **Overview**:
  + A feature-rich text editor with a wide range of functionalities beyond basic text editing.
  + Highly extensible with the Emacs Lisp programming language.
* **Features**:
  + Syntax highlighting, spell checking, file browsing, email, and even games.
  + Can be customized into a full-fledged development environment.
* **Key Commands**:
  + Ctrl + x, Ctrl + s: Save.
  + Ctrl + x, Ctrl + c: Quit.
  + Ctrl + g: Cancel the current command.
* **Why Use It?**
  + Ideal for programmers and those needing extensive customization.
  + Supports complex workflows and large projects.

**Comparison**

| **Feature** | **vi** | **ed** | **emacs** |
| --- | --- | --- | --- |
| Interface | Screen-based | Line-based | Screen-based |
| Usability | Quick and simple | Minimal and precise | Feature-rich and extensible |
| Learning Curve | Moderate | Steep | Steep |
| Customization | Limited | Minimal | Highly customizable |