

**Kathmandu University**  
**Department of Computer Science and Engineering**  
**Dhulikhel, Kavre**



**A Lab Report**  
**on**  
**“Operating System Lab-I”**

**[Code No.: COMP 307]**

**Submitted by**

**Prajwal Ghimire**

**Roll No.: 22**

**Submitted to**

**Mr. Rabina Shrestha**

**Department of Computer Science and Engineering**

**December 10, 2025**

## Questions

### **Q1: What is Linux?**

Linux is an open-source operating system based on the Unix architecture. It manages hardware resources, executes commands, and provides a secure multi-tasking environment. It powers servers, desktops, embedded devices, and even supercomputers. Some of the most popular Linux distributions are ArchLinux, Ubuntu, RedHat, etc.

### **Q2: The Linux Hierarchical File System**

Linux uses a hierarchical file structure that begins at the root directory /. Everything in Linux is a file or a directory, and all paths originate from /. Common directories include:

- /home – User home directories
- /bin – Essential command binaries
- /etc – Configuration files
- /usr – User utilities and applications
- /var – Logs, caches, temporary data

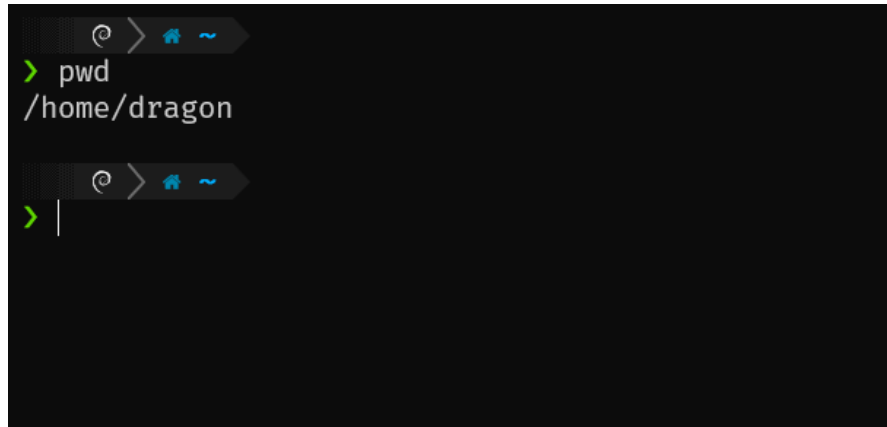
### **Q3: Importance of Linux commands in Operating Systems**

Linux commands are critical because they provide a direct and powerful interface to the operating system. They allow users and administrators to navigate the file system, manage files and directories, monitor system performance, and automate tasks through scripting. Unlike graphical interfaces, command-line commands are faster, use fewer resources, and provide more precise control. Mastering these commands enhances efficiency, troubleshooting capabilities, and overall understanding of how the OS operates, making it indispensable for system administrators, developers, and power users.

## Linux Commands

### 1. **pwd**

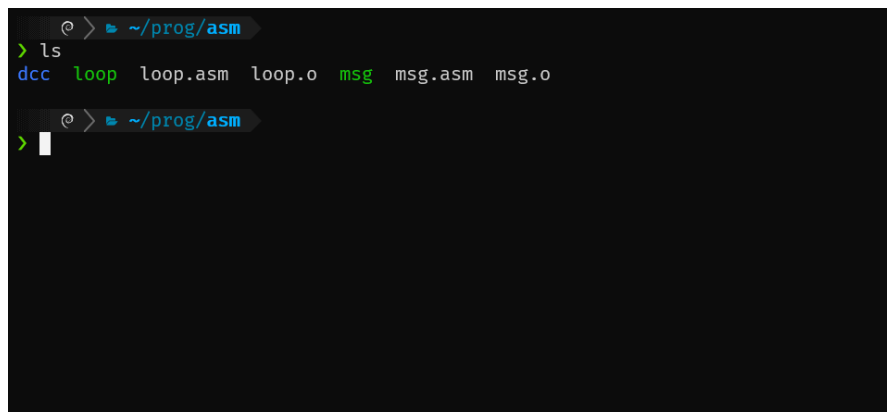
The `pwd` command prints your current working directory. It tells you exactly where you are located inside the Linux file system. This is extremely useful when navigating through multiple folders, working in scripts, or verifying paths before executing commands that affect files. Since Linux uses a hierarchical file structure starting at the root `/`, `pwd` helps ensure you never get lost.

A terminal window with a dark background. The prompt is a green greater-than sign. The user enters 'pwd' and the output is '/home/dragon'. The prompt is then shown again with a cursor, ready for the next command.

```
> pwd
/home/dragon
> |
```

### 2. **ls**

The `ls` command lists all files and directories in your current location. It gives a quick overview of the contents of a folder and is one of the most frequently used commands. By default, it shows only visible items (non-hidden files).

A terminal window with a dark background. The prompt is a green greater-than sign. The user enters 'ls' and the output is 'dcc loop loop.asm loop.o msg msg.asm msg.o'. The prompt is then shown again with a cursor, ready for the next command.

```
> ls
dcc loop loop.asm loop.o msg msg.asm msg.o
> |
```

### 3. **ls -a**

This version of `ls` displays all files, including hidden ones. Hidden files in Linux start with a dot (`.`), such as `.bashrc` or `.config`. These files usually store configurations and preferences.

```
❯ > ~/prog/asm
> ls -a
.  ..  dcc  loop  loop.asm  loop.o  msg  msg.asm  msg.o

❯ > ~/prog/asm
```

#### 4. **ls -l**

The **-l** option displays a long, detailed listing. It includes file permissions, owner, group, size, and last modification time. This format is essential for understanding access rights and managing file security.

```
❯ > ~/prog/asm
> ls -l
total 44
drwxr-xr-x 4 dragon dragon 4096 Jul  7 15:14 dcc
-rwxr-xr-x 1 dragon dragon 8928 Mar 30 2025 loop
-rw-r--r-- 1 dragon dragon 388 Mar 30 2025 loop.asm
-rw-r--r-- 1 dragon dragon 944 Mar 30 2025 loop.o
-rwxr-xr-x 1 dragon dragon 8864 Mar 28 2025 msg
-rw-r--r-- 1 dragon dragon 189 Mar 28 2025 msg.asm
-rw-r--r-- 1 dragon dragon 864 Mar 28 2025 msg.o

❯ > ~/prog/asm
```

#### 5. **cd**

The **cd** command lets you move between directories. It is used to navigate the Linux filesystem. You can move into subdirectories, return to the parent directory using **cd ..**, or jump to a specific absolute path.

```
@ > ~/prog/asm
> cd dcc

@ > ~/prog/asm/dcc > on master
> 
```

## 6. **mkdir**

**mkdir** creates a new directory. It is commonly used to organize files by grouping them into folders. You can also create multiple folders at once, or even nested folders using `mkdir -p`.

```
@ > ~/lab
> mkdir new

@ > ~/lab
> ls
new

@ > ~/lab
> 
```

## 7. **rmdir**

This command removes an empty directory. It cannot delete directories that contain files. It is useful for cleaning up folder structures or removing temporary empty folders. To remove the folder there must be a folder that is created.

```
Ⓢ > ~/lab
> rmdir new/

Ⓢ > ~/lab
> ll
total 16K
drwxr-xr-x  2 dragon dragon 4.0K Dec  9 21:07 .
drwx----- 74 dragon dragon 12K Dec  9 21:07 ..

Ⓢ > ~/lab
> 
```

## 8. **rm**

The **rm** command deletes files permanently (no recycle bin). It should be used carefully because deleted files are not easily recoverable. You can also remove multiple files at once.

```
Ⓢ > ~/lab
> ls
five.txt  four.txt  one.txt  three.txt  two.txt

Ⓢ > ~/lab
> rm one.txt two.txt three.txt four.txt five.txt

Ⓢ > ~/lab
> ls

Ⓢ > ~/lab
> 
```

## 9. **rm -r folder\_name**

The **-r** option stands for recursive deletion. It removes a directory and everything inside it — files, subfolders, and all. This is powerful and potentially dangerous, so double-check the directory before executing.

```
Ⓢ > ~/lab
> ls
hello

Ⓢ > ~/lab
> rm -r hello/

Ⓢ > ~/lab
> ls

Ⓢ > ~/lab
> 
```

#### 10. **touch**

**touch** is used to create an empty file or update the timestamp of an existing file. It is commonly used in scripting or when preparing placeholder files.

```
@ > ~/lab
> touch hello.txt

@ > ~/lab
> ls
hello.txt

@ > ~/lab
> 
```

#### 11. **cat**

The **cat** command reads and displays the content of a file directly in the terminal. It is also used to combine files or create files using output redirection.

```
@ > ~/lab
> cat hello.txt
Hello
This is the Lab Work for OS.

@ > ~/lab
> 
```

#### 12. **nano, vi, jed**

These are terminal-based text editors. **nano** is beginner-friendly, **vi** (or **vim**) is a powerful editor popular among developers, and **jed** provides a lightweight interface. They allow editing, writing, and saving files directly from the terminal.



### 13. **cp**

**cp** copies files from one place to another. You can also copy directories using the **-r** option. This command is essential for backups, duplication, and organizing files.

```

> cd ~/lab
> tree
.
├── dest
└── src
    └── hello.txt

3 directories, 1 file

> cd ~/lab
> cp src/hello.txt dest/

> cd ~/lab
> tree
.
├── dest
│   └── hello.txt
└── src
    └── hello.txt

3 directories, 2 files

>
  
```

### 14. **mv**

**mv** allows you to move or rename files and directories. Renaming is simply a move within the same folder. It's used for reorganizing or updating file names.



```
@ > ~/lab
> tree
.
├── dest
├── src
└── hello.txt

3 directories, 1 file

@ > ~/lab
> mv src/hello.txt dest/

@ > ~/lab
> tree
.
├── dest
│   └── hello.txt
└── src

3 directories, 1 file

@ > ~/lab
> 
```

### 15. **locate**

This command searches for files by name. It uses a system database, which makes the search extremely fast. It's ideal for finding misplaced files.

```
@ > ~/lab
> locate hello.txt
/home/dragon/lab/dest/ hello.txt

@ > ~/lab
> 
```

### 16. **echo**

Prints text or variable values to the terminal. It is commonly used in scripts to produce messages, debug values, or write text into files using redirection.

```
@ > ~/lab
> echo "Hello"
Hello

@ > ~/lab
> |
```

### 17. **uname -a**

Shows complete system information, including kernel version, machine architecture, hostname, and operating system. Useful for debugging or checking system specs.

```
@ > ~/lab
> uname -a
Linux dragon 6.6.87.2-microsoft-standard-WSL2 #1 SMP PREEMPT_DYNAMIC Thu Jun 5 18:30:46 UTC 2025 x86_64 GNU/Linux

@ > ~/lab
> |
```

### 18. **df -h**

Displays disk usage in human-readable format (MB/GB). It shows total size, used space, available space, and mounted filesystems. Handy for monitoring storage.

```
@ > ~/lab
> df -h
Filesystem      Size  Used Avail Use% Mounted on
none            3.9G   0    3.9G   0% /usr/lib/modules/6.6.87.2-microsoft-standard-WSL2
none            3.9G  4.0K   3.9G   1% /mnt/wsl
drivers         477G  312G  165G  66% /usr/lib/wsl/drivers
/dev/sdd        1007G   78G  879G   9% /
none            3.9G 200K   3.9G   1% /mnt/wslg
none            3.9G   0    3.9G   0% /usr/lib/wsl/lib
rootfs          3.8G  2.7M   3.8G   1% /init
none            3.8G   0    3.8G   0% /dev
none            3.9G 572K   3.9G   1% /run
none            3.9G   0    3.9G   0% /run/lock
none            3.9G  1.1M   3.9G   1% /run/shm
none            3.9G   80K   3.9G   1% /mnt/wslg/versions.txt
none            3.9G   80K   3.9G   1% /mnt/wslg/doc
C:\             477G  312G  165G  66% /mnt/c
tmpfs           780M   12K   780M   1% /run/user/1000

@ > ~/lab
> |
```

## 19. `ps -u $USER`

Lists all currently running processes for your user. It displays process IDs, CPU usage, memory usage, and command names. Very useful for identifying unnecessary or stuck processes.

```
@ > ~ /lab
> ps -u dragon
PID TTY          TIME CMD
762 ?            00:00:00 systemd
763 ?            00:00:00 (sd-pam)
793 pts/1        00:00:00 zsh
1065 ?          00:00:00 dbus-daemon
1068 ?          00:00:00 at-spi-bus-laun
1074 ?          00:00:00 dbus-daemon
1075 ?          00:00:00 gvfsd
1109 ?          00:00:00 at-spi2-registr
1341 pts/0        00:00:07 zsh
1345 pts/0        00:00:00 zsh
1466 pts/0        00:00:00 zsh
1467 pts/0        00:00:01 zsh
1469 pts/0        00:00:04 gitstatusd-linu
5638 pts/2        00:00:09 zsh
5645 pts/2        00:00:00 zsh
5764 pts/2        00:00:00 zsh
5765 pts/2        00:00:00 zsh
5767 pts/2        00:00:00 gitstatusd-linu
5982 pts/2        00:00:00 main
7839 pts/2        00:00:00 ps

@ > ~ /lab
> |
```

## 20. `top`

Displays real-time system activity. It shows active processes, CPU load, memory usage, and system uptime. It's one of the most important performance-monitoring commands.

```
top - 08:45:29 up 1 day, 6:36, 4 users, load average: 0.04, 0.02, 0.00
Tasks: 76 total, 1 running, 56 sleeping, 19 stopped, 0 zombie
Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
Mem Mem: 7792.2 total, 3174.3 free, 1307.5 used, 3516.3 buff/cache
Mem Swap: 2048.0 total, 2048.0 free, 0.0 used, 6484.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 273 mysql    20   0 2344624 396456 36992 S   0.6   5.0 13:23.52 mysqld
 820 root      20   0 553728 117108 43520 S   0.6   1.5 10:28.73 mongod
    1 root      20   0 167820 11720  8776 S   0.0   0.1 0:05.59 systemd
    2 root      20   0 3120 1920  1920 S   0.0   0.0 0:00.16 init-systemd(De
    6 root      20   0 3120 1792  1792 S   0.0   0.0 0:00.00 init
   58 root      20   0 49404 15360 14464 S   0.0   0.2 0:02.79 systemd-journal
   76 root      20   0 24984 5760  4608 S   0.0   0.1 0:11.06 systemd-udev
  178 root      20   0 1639732 36912 22400 S   0.0   0.5 0:09.21 bettercap
  179 root      20   0 3608 1920  1792 S   0.0   0.0 0:00.65 cron
  180 message+  20   0 8084 3968  3456 S   0.0   0.0 0:00.65 dbus-daemon
  189 root      20   0 475080 20352 18304 S   0.0   0.3 0:00.00 nis-daemon
  193 redis     20   0 60780 11392  8576 S   0.0   0.1 3:24.09 redis-server
  210 root      20   0 16724 7424  6528 S   0.0   0.1 0:00.76 systemd-logind
  214 root      20   0 394664 11872  9952 S   0.0   0.1 0:00.73 udiskd
  239 root      20   0 2524 1536  1536 S   0.0   0.0 0:00.00 agetty
  240 root      20   0 5880 1920  1792 S   0.0   0.0 0:00.00 atetty
  265 root      20   0 4668 908  640 S   0.0   0.0 0:00.00 in.tftpd
  271 root      20   0 6572 4756  3512 S   0.0   0.1 0:05.54 apache2
  335 polkitd   20   0 234408 7024  6384 S   0.0   0.1 0:00.04 polkitd
  368 postgres  20   0 216808 29568 27136 S   0.0   0.4 0:04.10 postgres
  392 postgres  20   0 216932 8580  6144 S   0.0   0.1 0:00.06 postgres
  393 postgres  20   0 216946 7172  4736 S   0.0   0.1 0:01.03 postgres
  400 postgres  20   0 216808 10372 7936 S   0.0   0.1 0:01.06 postgres
  401 postgres  20   0 218396 9092  6400 S   0.0   0.1 0:00.57 postgres
  402 postgres  20   0 218372 8324  5632 S   0.0   0.1 0:00.11 postgres
  746 debian+  20   0 30052 17784 5376 S   0.0   0.2 0:07.97 exim4
  752 root      20   0 3136 1164  1024 S   0.0   0.0 0:00.04 Relay(753)
  754 root      20   0 5804 3456  3072 S   0.0   0.0 0:00.00 login
  762 dragon    20   0 19200 10496 8704 S   0.0   0.1 0:00.20 systemd
  763 dragon    20   0 168480 4708  1536 S   0.0   0.1 0:00.00 (sd-pam)
  793 dragon    20   0 8876 4224  3584 S   0.0   0.1 0:00.01 zsh
  796 root      20   0 6548 3968  3584 S   0.0   0.0 0:00.00 sudo
 1065 dragon    20   0 7868 3840  3584 S   0.0   0.0 0:00.02 dbus-daemon
 1068 dragon    20   0 311032 7296  6656 S   0.0   0.1 0:00.01 at-spi-bus-laun
 1074 dragon    20   0 7784 3840  3584 S   0.0   0.0 0:00.00 dbus-daemon
```

## 21. **chmod**

Modifies file permissions. Permissions control who can read, write, or execute a file. `chmod` is essential for running scripts, securing files, and managing access rights.

```
@ > ~ /lab/dest
> ls -l
total 0
-rw-r--r-- 1 dragon dragon 0 Dec  9 21:39 hello.txt

@ > ~ /lab/dest
> chmod 755 hello.txt

@ > ~ /lab/dest
> ls -l
total 0
-rwxr-xr-x 1 dragon dragon 0 Dec  9 21:39 hello.txt

@ > ~ /lab/dest
> 
```