

ASSIGNMENT-1

NAME: Suraj P Das

USN: ENG25CY1006

SECTION: 3C

Roll no: 66

1. Linux programming is an open source and free. Unix like operating system. It is widely used in servers, desktops, embedded systems, and mobile devices. Linux is known for its stability, flexibility, and strong community support.

- Pros of Linux:

1. Most Linux distributions can be downloaded and used without cost.
2. Linux is less vulnerable to viruses and malware, and it can run for years without needing a reboot.
3. **Flexibility & Customization** – Users can customize the OS deeply, from the desktop environment to the kernel.

- Cons of Linux:

1. **Software Compatibility** – Some popular commercial software is not available natively.
2. **Learning Curve** – Beginners may find Linux harder to use compared to Windows or macOS.
3. **Hardware Support Issues** – Certain hardware may not work perfectly without manual setup.

2. Difference between Linux, MacOs, Android, Windows Os:

Features	Linux	MacOS	Android	Windows Os
Type	1. Open-source, free	1. Proprietary, Apple-only	1. Open-source	1. Proprietary (Microsoft)
Kernel	2. Linux kernel	2. Hybrid kernel (XNU)	2. Modified Linux kernel	2. Hybrid kernel (Windows NT)
Primary Use	3. Servers, supercomputers, developers	3. Apple desktops & laptops	3. Mobile devices, tablets, smart TVs	3. Personal PCs, enterprises, gaming
Customization	4. Highly customizable	4. Very limited (locked to Apple design)	4. Moderate (themes, OEM skins, root access)	4. Limited (some personalization options)
Software Support	5. Strong open-source apps; fewer native commercial apps	5. Optimized for creative tools (Final Cut Pro, Logic Pro)	5. Millions of mobile apps (Google Play Store)	5. Vast software library (MS Office, games, CAD tools)
Security	6. Very secure, less prone to malware	6. Strong, but more targeted than Linux	6. Secure but vulnerable to app-based malware	6. Frequent target for malware & viruses

3. Linux preferred for mainframe servers running legacy applications because it balances **backward compatibility** with **modern performance and flexibility**.

1. Virtualization & Partitioning at Scale (z/VM + KVM):

- Linux on mainframes supports **thousands of isolated virtual machines (VMs)** within a single physical server.
- Legacy applications can run in their own **virtual partitions**, while newer workloads (containers, microservices) coexist on the same machine without conflict.
- This allows legacy COBOL/CICS applications to run **side by side** with modern cloud-native workloads.

2. Binary & API Compatibility for Legacy Apps:

- IBM's **z/Linux (Linux on Z)** provides **binary compatibility** layers and libraries, so **old COBOL, PL/I, and FORTRAN apps** can run without needing full rewrites.
- Legacy middleware (like DB2, IMS, MQ) integrates seamlessly with Linux APIs, letting enterprises **wrap old apps with modern APIs (REST, SOAP, JSON)** instead of redeveloping them.

3. Extreme I/O Throughput & Reliability:

- Linux on mainframes leverages **specialized I/O processors (zIIPs, zAAPs)** that offload legacy application tasks.
- Supports **Parallel Sysplex clustering**, which gives “near-zero downtime” for mission-critical legacy workloads.
- Unlike typical servers, mainframe Linux can process **millions of transactions per second** while ensuring **fault-tolerance**—perfect for legacy banking, insurance, and telecom systems.

- The Linux file system follows a hierarchical, tree-like structure, with the root directory (/) at the top. All files and directories in a Linux system are organized under this single root, regardless of the physical storage devices they reside on. This structure is standardized by the Filesystem Hierarchy Standard (FHS).

Explanation of Key Directories:

- /bin: Contains essential user commands.
- /boot: Stores files required for booting the system, including the Linux kernel and GRUB bootloader files.
- /dev: Contains device files that represent hardware devices.
- /etc: Houses system-wide configuration files.
- /home: Contains individual user's home directories.
- /lib: Stores shared libraries needed by essential programs and kernel modules.
- /media: Provides mount points for removable media like USB drives and CDs.
- /mnt: A general-purpose mount point for temporarily mounting file systems.
- /opt: Used for installing optional, third-party software packages.
- /proc: A virtual filesystem that provides information about running processes and kernel parameters.
- /sbin: Contains essential system administration binaries.
- /tmp: Stores temporary files that are often cleared on reboot.
- /usr: Contains user-related programs, libraries, documentation, and other data that is not essential for booting the system.
- /var: Contains variable data, such as log files mail queues, and temporary files for services.

We can use the `tree` command in a Linux terminal to visualize the directory structure of a specific path. For example, to see the structure of your home directory:

```
tree -L 2 /home/username
```

5. Red Hat enterprise earn money by selling their support and training courses/certification. As we said Linux is an open source software so no one can sell it, but you can always sell your support as a service and that is what exactly Red Hat does.

Red Hat, the maintainer of one of the most popular Linux distributions, is another example of a large company. While they continue to build some free, open source solutions, they offer high-touch services and remote support to maintain the business. open source.

6 Ways Open Source Companies Can Make Money:

- Donations
- Hosted Version of the Project
- Paid Supported or Courses
- Open Core
- Dual Licensing
- Selling Other Products.

6. In Linux/Unix system we will use date command to display the current system date and time.

Command:

- date

Example:

- Mon Sep 22 21:30:12 IST 2025

7. In Linux/Unix, the command used to check how long the system has been running is actually in two ways but mainly we use one

command that is uptime command. It shows how long the system is running.

Command:

- `uptime`

Another way:

- `who -b`

This shows the last system boot time.

8. The difference between shutdown -h now and halt:

Shutdown -h now: Shut the system down and do it immediately

- Sends a warning to logged-in users.
- Notifies all processes and gives them time to exit cleanly.
- Unmounts file systems properly.
- Then halts or powers off.

It is **safer way** to bring down the system because it ensures data integrity.

halt: Stop all CPU functions immediately.

- Depending on how it is configured, it may **not** notify users or safely stop services.
 - It can behave like a "hard stop".
- In modern Linux systems, halt usually calls `systemctl halt`.

9. The difference between init 0 and shutdown -h:

Command	Action	Behavior	Safety Level
init 0	Switch system to runlevel 0	<ul style="list-style-type: none"> • Directly changes runlevel to 0. • Stops services and halts system. • May not warn users or give processes time to exit 	Less safe
shutdown -h	Shutdown and halt	<ul style="list-style-type: none"> • Sends warning to logged-in users. • Stops all services cleanly. • Unmounts file systems properly. • Halts or powers off the system. 	Safer

10. If a **server is powered off without a proper shutdown**, several problems can occur because the **shutdown sequence** like stopping services, flushing caches, unmounting file systems is skipped.

Here are the possible issues:

- File System Corruption
- Data Loss
- Database Corruption
- Hardware Stress
- Process Failures
- Longer Boot Time
- Configuration Loss
- Security Risks.