

Linux Programming: Assignment-1

NAME-Abinaya.P USN: ENG24CY0075 ROLL:44

Q1) what is Linux Operating System ?

Ans: In general Linux is basically a open-source developed by Linus Torvalds in 1991. Linux acts as an Core component of many operating systems Called Linux Distributions.

Linux OS is widely used in servers, desktops, embedded systems, and mobile devices due to its stability, security, and flexibility.

3-Pros of Linux :

1. Open Source and Free

Linux is freely available and users can view, modify, and distribute its source code.

2. Security and Stability

Linux is less vulnerable to malware and viruses compared to other OSes, making it highly secure and stable for long-running systems.

3. Highly Customizable

Users can customize Linux extensively, from the kernel to the desktop environment and applications, to suit specific needs.

3-Cons of Linux :

1. Steeper Learning Curve for Beginners

New users, especially those familiar only with Windows or macOS, may find Linux commands and system management challenging initially.

2. Limited Support for Some Commercial Software and Games

Many popular commercial applications and games do not natively support Linux, which can limit usability for some users.

3. Hardware Compatibility Issues

Some hardware manufacturers do not provide Linux drivers, leading to compatibility problems with certain devices like printers or graphics cards.

Q2: Differentiate between Linux, Mac, Android, and Windows OS with at least six unique features.

ANS:

When we compare these four operating systems, we see that each one is made for a different purpose and user base. Here's how they differ:

1. Kernel and base design

2. Linux runs on a monolithic kernel, it is flexible and you can add/remove modules.
3. macOS has the XNU hybrid kernel (Mach + BSD).

2. and availability

1. Linux is open-source (GPL license), anyone can use or modify it.

3: macOS is closed, only for Apple devices.
Android is partly open (AOSP free), but most phones add their own skin and apps.

3. User interface and experience

1. Linux depends on the desktop environment — GNOME, KDE, XFCE etc. It's very customizable but not always consistent.
2. macOS has a polished Aqua UI and works smoothly with iPhones and iPads.
3. Android is touch-based, Material Design style, focused on mobiles.

4. Applications and ecosystem

1. Linux software mostly comes through package managers (apt, dnf, pacman). Great for open-source apps but limited commercial ones.
2. macOS apps are via App Store or dmg/pkg downloads, with Xcode for developers.
3. Android apps are mostly from Google Play, also allows APK sideloading.

5. Security model

1. Linux has permissions (DAC), plus SELinux/AppArmor for strict control. It is considered very secure for servers.
2. macOS uses Gatekeeper and System Integrity Protection.
3. Android isolates each app in a sandbox, adds SELinux enforcing, but updates depend on phone brands.

Q3) Why is Linux preferred for Mainframe Servers for legacy applications? (3 technical reasons)

ANS

1. Rock-solid stability and long uptimes

Linux is famous for running non-stop for years without crashing. Mainframes handle critical workloads (like banks or government records) that can't afford downtime. Linux kernels with LTS (long term support) patches are very reliable, which makes it safe for old legacy apps to keep running smoothly.

2. Better workload isolation and resource control

Linux has features like cgroups and namespaces that can isolate applications, limit how much CPU or memory they use, and prevent one old app from crashing the entire system. This makes it easier to run many legacy programs together on the same big mainframe hardware without interference.

Q4) Structure of the Linux File System

ANS-- Linux follows a **hierarchical file system** structure, meaning everything starts from a single root directory / and branches out like a tree. Unlike Windows which has drives like C:\ or D:\, Linux treats everything (files, devices, configs) as part of one unified directory tree.

Key directories inside /:

1. /bin → Essential user commands (ls, cp, mv, etc.)
2. /sbin → System admin commands (mount, shutdown, etc.)
3. /etc → Configuration files of the system
4. /lib → Shared libraries needed by programs in /bin and /sbin

5. /usr → User programs, libraries, documentation (like /usr/bin, /usr/lib)
6. /var → Variable data like logs, mail, cache, spool
7. /home → Personal directories for users
8. /tmp → Temporary files (auto-cleared)
9. /boot → Bootloader files, kernel image
10. /dev → Device files (disks, USB, etc.)

Q5) If Linux is open-source, how does Red Hat make money?

ANS-- 1) What are customers actually buying?

1. RHEL subscriptions
Same Linux you can compile yourself, but curated, tested, and supported for 10 years. Stable kernels, backported fixes, predictable updates. Enterprises hate surprises.
2. OpenShift (Kubernetes platform)
Red Hat's opinionated K8s stack: cluster lifecycle, upgrades, security, logging, service mesh, operators. Basically Kubernetes that won't blow up your weekend.
3. Automation (Ansible)
Enterprise automation + content catalogs + controller/analytics. Teams pay to standardize how they configure, patch, and deploy thousands of nodes.
4. Certifications & compliance
FIPS/CIS/STIG baselines, signed updates, provenance. Also "this OS is certified with SAP/Oracle/NVIDIA/VMware/your hardware." That certification matrix is the moat.

2) Who pays and why?

1. Banks, telecom, govt, healthcare, manufacturing—places where downtime is expensive or illegal.
2. They pay because:
 - a. Risk transfer: "If it breaks at 2 AM, someone accountable fixes it."
 - b. Lifecycle guarantees: 8–10 years of updates on the same major release.
 - c. Ecosystem lock-in (the positive kind): Works out-of-the-box with their storage, NICs, GPUs, HSMs, databases.

3) How do they price it ?

1. Per node / per core subscriptions for RHEL (standard vs premium support tiers).
2. Per cluster / per worker style for OpenShift (different SKUs for datacenter, edge, cloud).
3. Enterprise license for Ansible Automation Platform (nodes + controllers).

4) Why does open-source not kill their business?

1. Curation > code. The raw code is free; the predictable, tested, certified build is not.
2. Security & backports. They ship CVE fixes without forcing you to jump to a new upstream version; that's harder than it looks.

3. Enterprise rituals. Change windows, long-term support, break-glass patches, and someone to sign off on risk.

5) Competitors and substitutes (so you sound market-aware)

1. Canonical (Ubuntu), SUSE—similar open-source + support play.
2. Cloud-native substitutes: AWS/Azure/GCP managed Kubernetes instead of OpenShift; AMIs and cloud images instead of on-prem RHEL.

Q6) Write the command to display today's date and time

In Linux we can get it if we type:

Bash programming --- date

Then the Output Looks like:

Sat Sep 21 08:42:31 IST 2025

This shows day, month, date, current time, time zone, and year

Q7) Which command is used to check how long the system has been running?

The most common command is:

In Bash programming we type ---- uptime

output:

08:55:03 up 3 days, 4:17, 2 users, load average: 0.12, 0.08, 0.01

This means that:

1. Current time → 08:55:03
2. System uptime → 3 days, 4:17
3. Number of logged-in users → 2 users

Q8) Difference between shutdown -h now and halt

1. shutdown -h now

This command tells the system to *shutdown immediately*. It first sends a warning message to all logged-in users, then stops running processes in an orderly way, unmounts file systems safely, and finally powers off the machine.

2. halt

The halt command simply stops all CPU functions. On modern Linux systems (with systemd), it also goes through a safe shutdown sequence, but traditionally it just halts the CPU and doesn't always power off the machine. Sometimes after halt, you may still need to press the power button to switch it off.

Q9) Compare init 0 and shutdown -h. Which is safer? Why?

a) init 0

This command changes the system runlevel to 0, which is the runlevel for system halt. It stops all processes and powers down the system. It works, but it doesn't

always notify logged-in users or give a clean timed shutdown.

b) shutdown -h

This command is the standard way to shut down. It safely brings the system to a halt by:

1. Sending warning messages to all users
2. Stopping services in the proper order
3. Unmounting filesystems cleanly
4. Finally powering off the machine

Q10) If a server is powered off without proper shutdown, what problems can occur?

1. File system corruption

Since the system didn't unmount disks properly, there may be incomplete writes left in memory. This can damage file system structures, forcing fsck checks at the next boot.

2. Data loss

Any unsaved data sitting in cache or RAM will be lost. Databases or applications may lose recent transactions or need recovery.

3. Service disruptions

Critical services (like web servers, databases, mail servers) stop abruptly. On restart, they may take extra time to recover or rebuild indexes.

4. Hardware stress

Repeated sudden power cuts can affect storage devices (especially HDDs and SSDs) because heads or flash controllers don't get a chance to park or flush data.