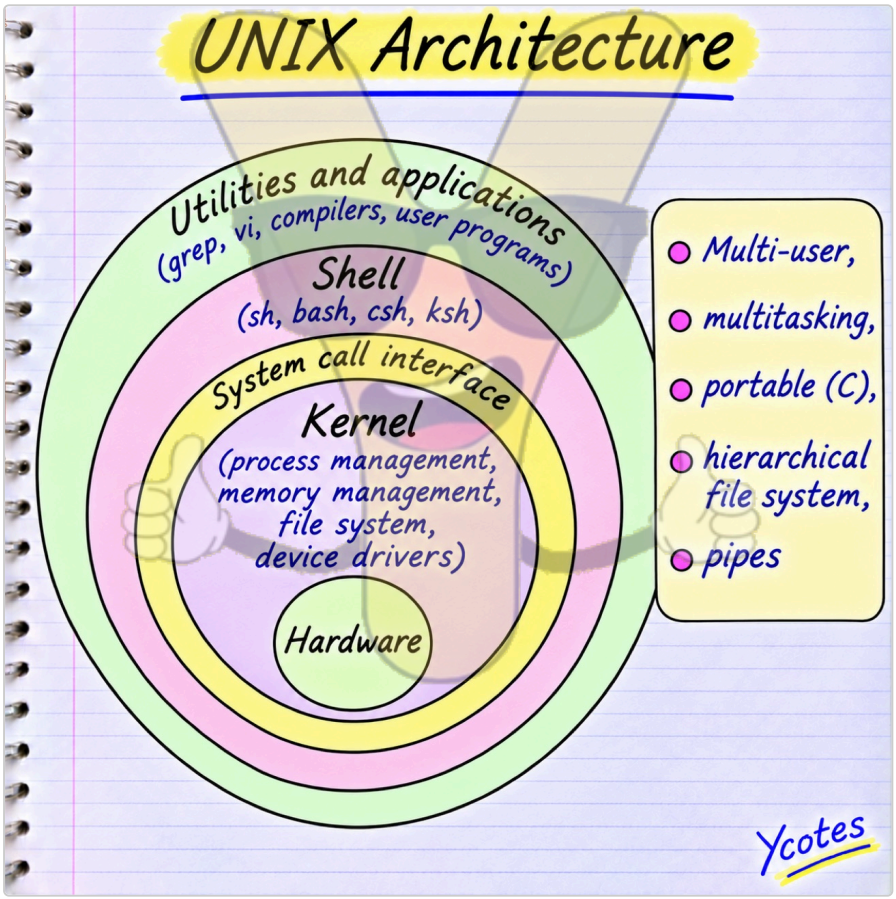
UNIT–5 : UNIX, Linux, Real-Time & Mobile OS (PYQ Solution)

PART–B Answers (≈250 Words Each)

Q1. Differentiate between UNIX and Windows based operating systems. (2021)

UNIX (Bell Labs, 1969) is a multi-user, multitasking, portable operating system written in C. **Windows** is a GUI-first operating system from Microsoft.



Basis	UNIX	Windows
Developer	Bell Labs, later AIX, Solaris, HP-UX, BSD	Microsoft
Source and cost	Mostly proprietary and costly (BSD is free)	Closed source, licence fee
User interface	Command line (shell) mainly, GUI optional (CDE, X11)	GUI first, command prompt and PowerShell secondary
Users	Multi-user by design	Originally single-user, multi-user now supported
Kernel	Monolithic (or microkernel)	Hybrid (NT kernel)
File system	Single root `/`, UFS, ZFS; everything is a file	Drive letters (`C:\`), NTFS, FAT32
Case sensitivity	Case-sensitive	Case-insensitive
Path separator	`/`	`\`
Security	Strong rwx permissions, fewer viruses	More targeted by malware, ACL based
Process creation	`fork()` and `exec()`	`CreateProcess()`
Used for	Servers, workstations, mainframes	Desktops, business, gaming
Customisation	Highly scriptable (shell scripts, pipes)	Limited, mostly through GUI

Architecture: UNIX is layered as hardware, kernel, shell and applications. Windows has a user mode (applications, subsystems) and a kernel mode (executive, kernel, HAL).

Conclusion: UNIX is preferred for servers and programming because it is stable, scriptable and multi-user, while Windows is preferred for desktop use because of its GUI and software support.

Q2. Differentiate between Windows and Linux based operating system. (2022, Feb 2024)

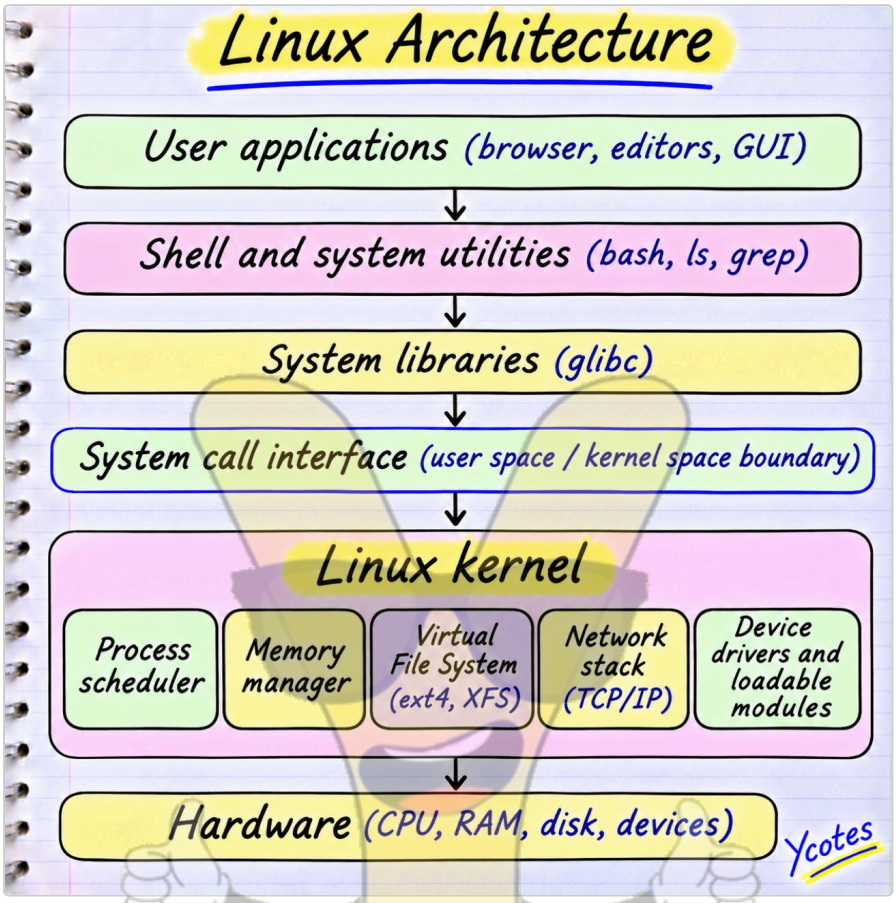
Windows is a proprietary, GUI-based operating system by Microsoft. **Linux** is a free, open-source, UNIX-like operating system whose kernel was created by Linus Torvalds in 1991.

Basis	Windows	Linux
Source	Closed, proprietary	Open source (GPL)
Cost	Licence fee	Free
Kernel	Hybrid (NT)	Monolithic with loadable modules
Interface	GUI first	CLI plus many GUIs (GNOME, KDE)
File system	NTFS, FAT32; drive letters (`C:\`)	ext4, XFS; single root `/`
Case sensitivity	Case-insensitive	Case-sensitive
Security	More exposed to viruses; antivirus needed	Strong permissions, fewer viruses, quick patches
Customisation	Limited	Highly customisable, many distributions (Ubuntu, Fedora, Debian)
Software	Huge commercial and game software	Open-source tools, servers; fewer games
Hardware support	Wide, vendor drivers	Good, drivers in kernel
Users	Home, office, gaming	Servers, developers, embedded, cloud
Updates	Controlled by Microsoft, may force reboot	User-controlled, rarely needs reboot
Support	Official paid support	Community and paid vendors
Command shell	Command Prompt, PowerShell	bash, zsh

Conclusion: Windows is easy for ordinary users and widely supported, while Linux is cheaper, more secure and flexible, and runs most servers and supercomputers.

Q3. Explain the architecture of Linux operating system with proper diagram. (2023)

Linux is a free, open-source, UNIX-like operating system with a **monolithic but modular** kernel. Its architecture is layered, as shown below.



Layers (bottom to top):

1. **Hardware:** CPU, RAM, disks, network cards and other devices.
2. **Kernel space:** the core of the OS, which runs in privileged mode and has these subsystems:
   - **Process scheduler:** the Completely Fair Scheduler (CFS) shares the CPU.
   - **Memory manager:** virtual memory, paging and swapping.
   - **Virtual File System (VFS):** a common interface to ext4, XFS, NTFS and others.
   - **Network stack:** TCP/IP and sockets.
   - **IPC:** pipes, signals, shared memory and message queues.
   - **Device drivers and loadable kernel modules:** add or remove drivers without rebooting.

3. **System call interface (SCI):** the boundary through which user programs request kernel services (`open`, `read`, `fork`).
4. **System libraries:** glibc, which wraps the system calls for programs.
5. **User space:** the **shell** (bash), system utilities, daemons and applications (GUI desktop, browser, compilers).

How it works: an application calls a library function, which makes a system call. The CPU switches to kernel mode, the kernel subsystem does the work (using the driver for hardware), and control returns to user mode with the result.

Features: open source, multi-user, multitasking, portable, secure permissions, stable and many distributions.

Advantages: free, fast, secure and customisable, and modules keep the kernel flexible. **Disadvantage:** because it is monolithic, a faulty driver in kernel space can crash the system.



PART–C Answers (≈500 Words Each)

Q1. What are the different mobile operating systems used nowadays? Discuss their features. (2019)

A **mobile operating system** runs on smartphones and tablets. It is designed for **touch input, battery life, limited memory, wireless connectivity and apps**.

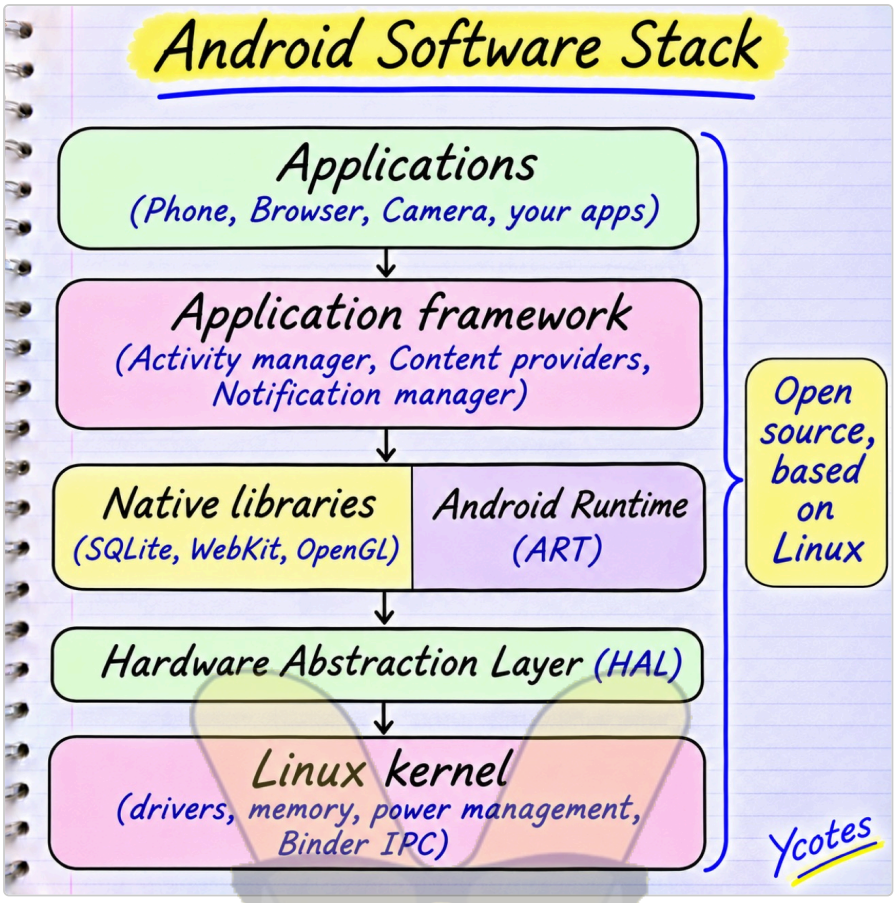
Common mobile operating systems:

OS	Developer	Base and language	Notes
Android	Google	Linux kernel; Java, Kotlin	Open source, most widely used, Google Play
iOS	Apple	XNU (Darwin); Swift, Objective-C	Closed, smooth, App Store, strong security
HarmonyOS	Huawei	Microkernel	Works across phone, watch and TV
KaiOS	KaiOS Technologies	Linux-based	Feature phones
Windows Phone	Microsoft	Windows NT based	Discontinued

Features of mobile OS:

- **Touch and gesture UI** with on-screen keyboard and voice assistant.
- **Power management** to save battery.
- **Multitasking with app sandboxing**, where each app runs in its own protected space.
- **Sensors:** GPS, accelerometer, gyroscope, camera and fingerprint.
- **Connectivity:** Wi-Fi, 4G/5G, Bluetooth, NFC.
- **App stores** (Play Store, App Store) and push notifications.
- **Security:** permissions, encryption, biometric lock and remote wipe.
- **Over-the-air updates.**

Android software stack (top to bottom):



1. **Applications:** phone, browser and user apps.
2. **Application framework:** activity manager, content providers, notification manager.
3. **Libraries and Android Runtime (ART):** SQLite, WebKit, OpenGL and the runtime that executes apps.
4. **Hardware Abstraction Layer (HAL):** standard interface to camera, audio and sensors.
5. **Linux kernel:** drivers, memory, power management and binder IPC.

Basis	Android	iOS
Source	Open source	Closed
Devices	Many manufacturers	Apple only
Customisation	High	Limited
Updates	Vary by manufacturer	Fast and uniform
Store	Google Play	App Store

Advantages: portable computing, huge app ecosystem and always-on connectivity.
Disadvantages: limited battery and memory, security threats from malicious apps

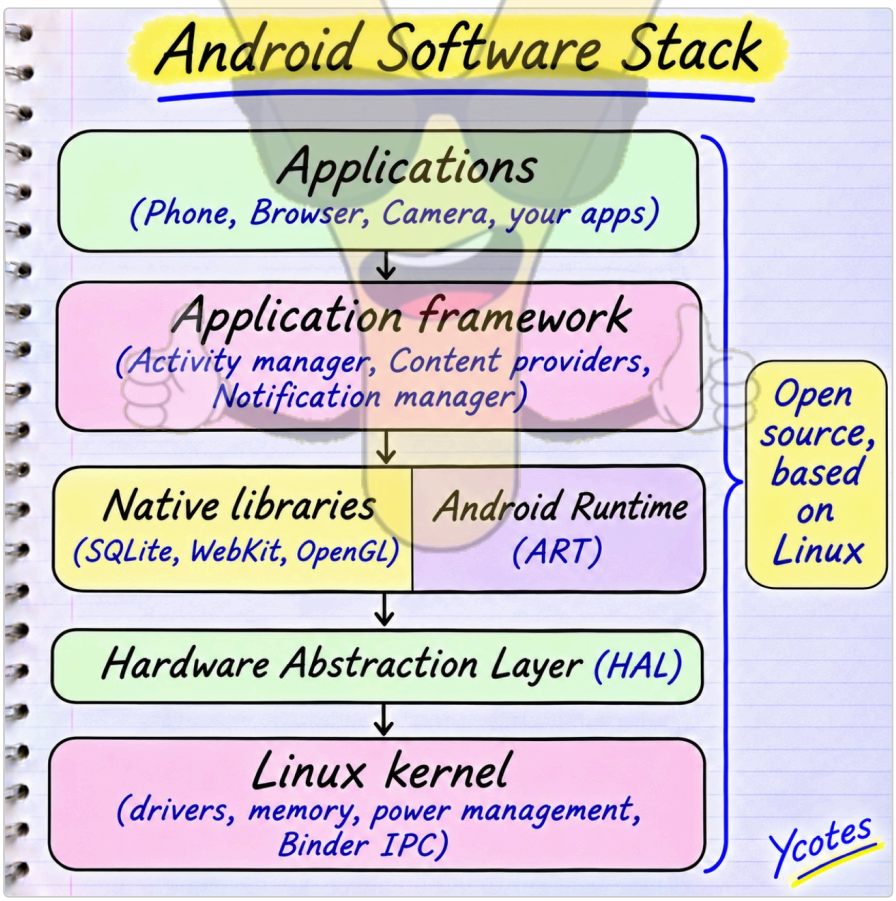
and fragmentation of Android versions.

Q2. Short notes - (a) Mobile OS (b) Time OS (real-time OS). (2023, 2024)

(a) Mobile OS

A **mobile OS** is an operating system built for smartphones, tablets and wearables. It manages touch input, battery, memory, sensors, wireless networks and apps within small hardware limits.

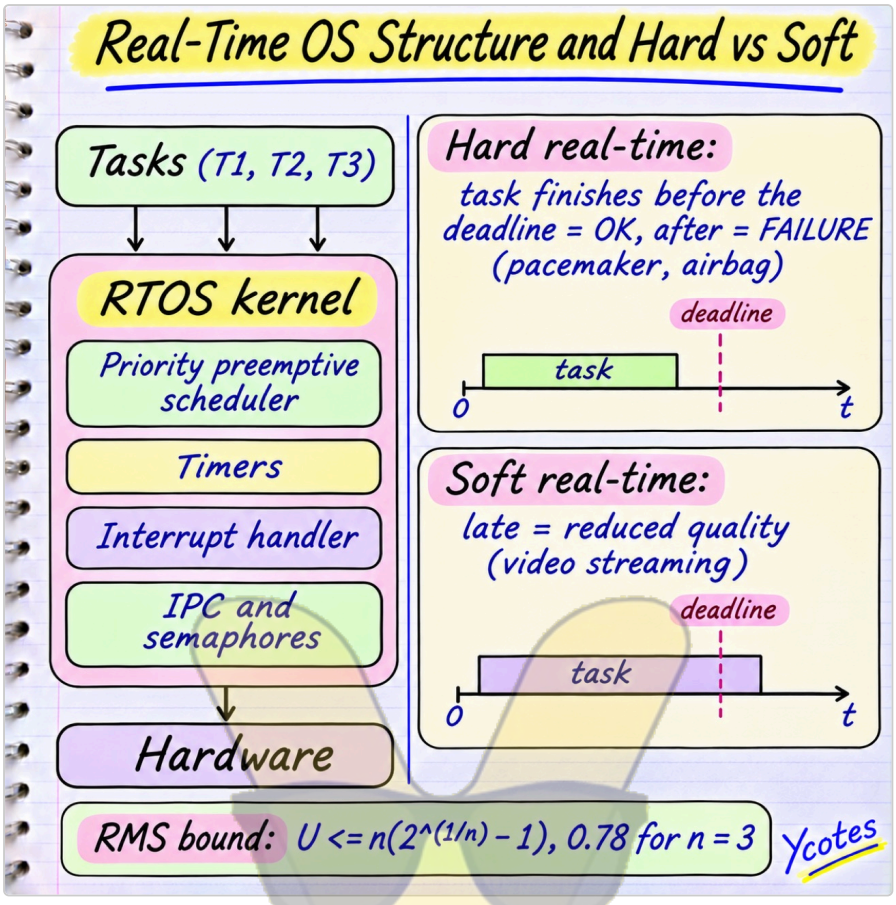
- **Examples:** Android (Linux kernel, open source), iOS (Apple, closed), HarmonyOS, KaiOS.
- **Features:** touch UI, power management, app sandboxing, sensors, app stores, permissions-based security, Wi-Fi, 4G/5G, Bluetooth, over-the-air updates.
- **Architecture (Android):** apps, application framework, libraries with ART, HAL and Linux kernel.



(b) Time OS (real-time operating system, RTOS)

An **RTOS** is an operating system that guarantees tasks complete **within strict time limits called deadlines**. Its correctness depends on both the **result** and the **time** at

which it is produced.



Hard real-time	Soft real-time
Missing a deadline is a **failure** and may be a disaster	Missing a deadline only reduces quality
Examples: pacemaker, airbag, flight control, missile guidance	Examples: video streaming, online games, multimedia
Little or no virtual memory and secondary storage	May use more general features

A **firm** real-time system is one where a late result is useless but not harmful.

Features: deterministic and predictable response, **priority-based preemptive scheduling**, low interrupt latency, fast context switch, small kernel, timers, and semaphores and message passing for IPC.

Scheduling:

- **Rate Monotonic (RM):** fixed priority, shorter period gets higher priority. Schedulable if $U = \sum C_i/T_i \leq n(2^{1/n} - 1)$. For $n = 3$ the bound is 0.78.

- **Earliest Deadline First (EDF):** dynamic priority, the earliest deadline runs first. Schedulable if $U \leq 1$.

Example: three tasks with $(C, T) = (1, 4), (2, 6), (1, 8)$ give $U = 0.25 + 0.333 + 0.125 = 0.708$, which is below 0.78, so RM can schedule them.

Examples of RTOS: VxWorks, QNX, FreeRTOS, RTLinux, eCos.

Applications: industrial control, robotics, aircraft, medical devices, automotive systems.

