**1. Web App Development Across Platforms**

To build a web application that functions across all platforms, developers use frameworks and libraries like **React**, **Angular**, and **Vue**. Among these, **React** is a JavaScript library widely preferred for its component-based architecture, virtual DOM, and strong community support. It ensures cross-platform compatibility, making it an ideal choice.

**2. Purpose of RAID**

**RAID (Redundant Array of Independent Disks)** enhances **data reliability** by distributing data across multiple drives. There are different RAID levels:

* **RAID 0** (Striping) – Improves speed but has no redundancy.
* **RAID 1** (Mirroring) – Duplicates data for reliability.
* **RAID 10** – Combines striping and mirroring for performance and fault tolerance.

The primary goal is to **improve data reliability** by reducing the risk of data loss due to hardware failures.

**3. Communication Channels and Full-Duplex**

Communication can occur in different modes:

* **Simplex**: One-way communication (e.g., radio broadcasting).
* **Half-Duplex**: Two-way communication but one direction at a time (e.g., walkie-talkies).
* **Full-Duplex**: Simultaneous two-way communication (e.g., phone calls).
* **Multi-Duplex**: Not a standard term in networking.

Full-duplex systems enhance efficiency by allowing data to flow in both directions at the same time.

**4. & 5. Programming Languages for iOS and Android**

* **Swift**: Apple's official language for iOS/macOS development, optimized for performance and safety.
* **Java**: The primary language for Android development, though **Kotlin** is also widely used today.

These languages provide native performance and API access.

**6. Benefits of CLI**

A **Command Line Interface (CLI)** provides:

* Faster execution of commands.
* More direct control over the system.
* Ability to automate tasks through scripts. Graphical interfaces are user-friendly, but CLI is preferred for efficiency in system administration.

**7-9. Motherboards and Form Factors**

Motherboards differ in size and capability:

* **ATX**: Supports full-size expansion slots.
* **Mini-ITX**: Compact, with limited slots.
* **Micro-ATX**: A balance between size and expansion options.
* **Riser Card-Based Motherboard**: Uses right-angled components for better airflow and cooling, often found in server configurations.

**10. Network Topologies**

* **Ring**: Easy to install but difficult to reconfigure.
* **Star**: Most common; central switch/hub connects devices.
* **Tree & Hybrid**: Used for large-scale networks.

Ring topology minimizes cabling costs but lacks flexibility.

**11. Memory Types**

* **RAM (Random Access Memory)**: Temporary, volatile memory.
* **ROM (Read-Only Memory)**: Non-volatile memory, hardwired onto a chip.
* **Cache & DRAM**: Used for faster data access.

ROM is critical for boot processes as it retains essential instructions.

**12. Cost-Effective Data Storage**

For large storage needs (e.g., **16TB**), **HDDs** are more affordable than **SSDs**. However, SSDs offer higher speed and reliability.

**13. Interpreted Languages**

An **interpreted language** executes code line-by-line, unlike compiled languages that convert code into machine instructions beforehand. **Python** is a key example, offering flexibility for development.

**14. Authentication Methods**

* **Something you know**: Passwords, PINs.
* **Something you have**: Tokens, smart cards.
* **Someone you know**: Biometric authentication (fingerprints, face recognition).
* **IP Address**: A network-based identifier.

Passwords vary, making them a common authentication factor.

**15. Formatting a Website UI**

**CSS (Cascading Style Sheets)** controls a website's visual design. Unlike **PHP (backend)** or **SQL (databases)**, CSS focuses on presentation.

**16. & 17. Storage and RAID 10**

* **NVM (Non-Volatile Memory)** retains data after power loss, making it useful for SSDs.
* **RAID 10** provides speed (striping) and redundancy (mirroring), combining the advantages of RAID 0 and RAID 1.

**18-19. Network Connectivity**

* **WiFi** enables wireless device connectivity in environments like warehouses.
* **Fiber Optic Cables** provide high-speed data transfer (10GBps+), essential for fast networks.

**20-22. Security Measures**

* **UTM (Unified Threat Management)** consolidates security measures like firewalls and antivirus solutions.
* **VPNs** encrypt data to provide secure internet access.
* **Containers** run **above the host OS**, isolating applications for efficiency and security.

**23-24. Cloud Computing and Scaling**

* **SaaS (Software as a Service)**: Provides IT services (e.g., help desks).
* **Horizontal Scaling**: Expanding by adding more servers instead of upgrading a single system.

**25. Operating System Role**

An **OS (Operating System)** is the interface between software and hardware, executing user commands and managing system resources.

**26-27. Windows vs. Linux & Hypervisors**

Both **Windows and Linux** support file management and program execution. A **hypervisor** enables virtualization but requires sufficient memory and CPU power.

**28. Backend Web Development**

**Python** is a backend language, used for handling server logic, database interactions, and API requests. Frontend languages like **HTML, CSS, and Vue.js** focus on UI design.

**29-30. Virtual Environments & Data Registers**

* **Virtual Sandboxes** safely test software, including legacy programs.
* **Memory Data Register (MDR)** temporarily stores data for processing.

**31-32. Virtual Machines & Applications**

A **hypervisor** enables multiple users to share an OS via **VMs (Virtual Machines)**. A **program application** is software designed for specific tasks.

**33. Sandbox vs. Test Environments**

* **Sandbox**: Isolated testing.
* **Test Environment**: Simulates real-world use before deployment.

Both are crucial for software development.

**34. Hadoop and Big Data**

**Hadoop** is used for processing large datasets across distributed systems, essential in big data analytics.

**35. Variables in Programming**

A **variable** stores values dynamically during program execution, unlike classes or modules.

**36-40. Cybersecurity and Malware**

* **Keyloggers** record keystrokes for malicious purposes.
* **Boot Sector Viruses** infect system boot records.
* **Ransomware** encrypts files for ransom.
* **Trojan Horses** disguise themselves as legitimate software.
* **Rootkits** hide malicious activity and grant attackers unauthorized access.

**41. Workstation Operating Systems**

A **workstation operating system** is designed for personal computing, allowing users to run applications and manage system hardware efficiently. It differs from server or network operating systems, which are optimized for handling multiple users or managing network resources. Examples include **Windows 10/11, macOS, and various Linux distributions**. These OSes support productivity software, gaming, and multimedia applications.

**42. Operating System Life Cycle: Extended Support**

Operating systems go through various life cycle stages, including **beta testing, release candidate, mainstream support, and extended support**. The **extended support** phase ensures that critical security patches continue to be released, even after the system stops receiving new features. For example, **Windows 10 will receive security updates until October 2025**.

**43. Graphical User Interface Shell in Linux: GNOME**

Linux offers multiple graphical user interfaces (GUIs) called **desktop environments**. **GNOME** is one of the most popular GUI shells, providing a user-friendly interface with window management, application launching, and system controls. Other alternatives include **KDE Plasma, XFCE, and Cinnamon**.

**44. Windows API System Calls: CreateProcess()**

System calls allow programs to interact with the operating system. In **Windows API**, the CreateProcess() function is used to **create and manage new processes**. This is essential for launching applications, managing background tasks, and handling inter-process communication.

**45. Programming-Language Support Services**

A **system service category** that includes compilers, debuggers, and interpreters is **Programming-Language Support**. These tools are essential for software development, enabling programmers to **write, debug, and execute code efficiently**. Examples include **GCC (GNU Compiler Collection) and Microsoft’s .NET Framework**.

**46. iOS and SpringBoard**

Apple’s **iOS operating system** features **SpringBoard**, a graphical interface responsible for app launching and home screen management. It provides **multi-touch gestures, widgets, and notification controls**, making it a crucial component of the user experience.

**47. RAID for Reliability**

**RAID (Redundant Array of Independent Disks)** is a data storage technology that improves reliability through redundancy. **RAID 1 and RAID 5** are commonly used for **fault tolerance**, ensuring data remains accessible even if a hard drive fails.

**48. Motherboard Buses**

A **bus** is a crucial motherboard component that facilitates communication between different hardware parts. It consists of **address buses, data buses, and control buses**, enabling data transfer between the **CPU, RAM, and peripherals**.

**49. Northbridge for High-Speed Communication**

The **Northbridge** chipset manages high-speed communication between **the CPU, RAM, and GPU**. It ensures efficient data transfer and is often paired with the **Southbridge**, which handles slower peripherals like USB and storage.

**50. Memory Address Register (MAR)**

The **Memory Address Register (MAR)** holds the memory location address currently being accessed by the CPU. It plays a key role in the **fetch-execute cycle**, ensuring that data is retrieved or stored in the correct location.

**51. Data Movement Instructions**

Data movement instructions handle **transferring information between CPU registers, memory, and input/output devices**. These instructions are vital for efficient program execution and memory management.

**52. Hypervisor and Emulation**

A **hypervisor** enables virtualization by **emulating hardware** so that multiple virtual machines (VMs) can run on the same physical system. There are **Type 1 hypervisors** (bare-metal) like **VMware ESXi** and **Type 2 hypervisors** like **VirtualBox**.

**53. Type 1 Hypervisors**

A **Type 1 hypervisor** runs directly on system hardware, unlike Type 2, which requires a host OS. Examples include **Microsoft Hyper-V and XenServer**.

**54. Intrusion Prevention Systems (IPS)**

An **IPS (Intrusion Prevention System)** actively **monitors and blocks malicious activity in real-time**. Unlike an **IDS (Intrusion Detection System), which only alerts**, an IPS can **take automated action** to stop threats.

**55. Platform as a Service (PaaS)**

**PaaS provides cloud-based tools for software development**, including **frameworks, databases, and runtime environments**. Examples include **Google App Engine, AWS Elastic Beanstalk, and Microsoft Azure App Services**.

**56. Debugging with Breakpoints**

A **breakpoint** is a debugging tool that **pauses program execution at a specific line of code**, allowing developers to inspect variables and behavior before proceeding.

**57. Procedural Programming and Modules**

In procedural programming, **modules** help **organize code into reusable, structured segments**, improving code maintainability.

**58. VBScript and Component Object Model (COM)**

**VBScript** is a scripting language based on **COM**, allowing Windows-based automation, web development, and application control.

**59. Patch Panels in Networking**

A **patch panel** is a large hub used in structured cabling to **connect multiple network cables**, facilitating easy management in **data centers and office networks**.

**60. Category 3 Twisted Pair Cables**

**Category 3 (Cat 3) cables** were used in early Ethernet networks and could support **up to 16 Mbps**, making them suitable for telephone wiring.

**61. Network Protocols**

A **protocol** defines the set of rules governing **data transmission, error handling, and communication** in networking. Examples include **TCP/IP, HTTP, and FTP**.

**62. Client-Server Model**

A **client-server model** involves a **centralized application providing resources** to multiple clients. Examples include **web servers and file servers**.