**Day 1: Introduction to Operating Systems**

* Operating systems act as resource managers.
* Single-user systems manage personal computer resources.
* Examples of single-user systems: Windows, macOS, Linux.
* Mobile OS: Android (70% market share) and iOS (28% market share).
* Non-magnetic storage in mobile devices resists static fields.
* Mainframe OS: Enterprise-scale applications.
* Network server systems manage databases and web services.
* Thin client systems rely on server processing.
* Real-Time OS is used in air traffic control and autonomous cars.
* Embedded control systems run low-power IoT devices like fridges.
* Distributed OS allocates tasks across a network.
* Cloud-based OS is stateless and works in virtualized environments.
* Emulator allows one OS to mimic another (e.g., Windows on Mac).
* Cross-platform applications run on multiple OS types.
* Multiprogramming supports multiple concurrent programs.

**Day 2: System Functionality**

* Multitasking allows simultaneous process execution.
* Preemptive multitasking allocates equal resources to processes.
* Cooperative multitasking relies on voluntary process handoffs.
* Hybrid multitasking combines preemptive and cooperative methods.
* Concurrency allows multiple threads to access shared resources.
* Batch processing handles repetitive, high-volume jobs.
* Process Control Block stores OS information about processes.
* SMART monitors HDD and SSD performance.
* HCL lists compatible hardware for OS.
* System calls enable program-to-kernel communication.
* Process control manages CPU and memory for tasks.
* File management organizes and retrieves stored data.
* Device management oversees peripheral connections.
* Communication ensures seamless data transfer between systems.
* Protection enforces file permissions and data security.

**Day 3: Hardware Overview**

* Motherboards connect all components of a computer.
* ATX boards are standard-sized with seven expansion slots.
* Mini ATX and Micro ATX reduce size but limit features.
* Mini ITX is compact and uses a single expansion slot.
* CPU sockets use ZIF mechanisms for easy installation.
* Intel sockets use LGA, while AMD sockets use PGA.
* Northbridge connects the CPU to fast peripherals.
* Southbridge handles slower peripherals like USB devices.
* PCI slots are older technologies for add-on cards.
* PCIe provides high bandwidth for modern components.
* BIOS manages hardware configurations during boot.
* CMOS battery maintains BIOS settings when powered off.
* Port clusters consolidate input/output connections.
* CPU has an ALU for arithmetic operations.
* Control Unit (CU) orchestrates CPU operations.

**Day 4: CPU and Memory**

* Registers are small, fast data storage in CPUs.
* MAR and MDR facilitate communication with RAM.
* Program Counter (PC) tracks the next instruction.
* Pipelining overlaps CPU tasks for efficiency.
* Multithreading executes parallel threads of the same process.
* SMP uses multiple processors for performance.
* Multi-core CPUs integrate multiple processors into one.
* Virtualization enables multiple systems on one CPU.
* RAM stores active data for CPU processing.
* DRAM constantly refreshes data to prevent loss.
* SRAM is faster but more expensive than DRAM.
* Cache levels (L1, L2, L3) reduce data access time.
* Virtual RAM extends memory using storage devices.
* ECC memory detects and corrects errors.
* ROM stores essential startup instructions.

**Day 5: Storage Devices**

* HDDs use spinning disks and magnets to store data.
* SSDs use flash memory for faster access.
* SATA replaced older PATA technology.
* PCIe SSDs support NVMe for high-speed data transfer.
* Hybrid drives combine HDD and SSD features.
* Flash memory is ideal for portable devices.
* RAID 0 provides speed but no redundancy.
* RAID 1 mirrors data for backup purposes.
* RAID 5 balances speed and redundancy.
* RAID 6 adds extra redundancy for drive failures.
* RAID 10 combines RAID 0 and RAID 1 benefits.
* Optical storage includes CDs and DVDs.
* USB drives offer portable, fast storage.
* NAS devices enable shared network storage.
* Cloud storage integrates physical and virtual solutions.

**Day 6: Virtualization and Cloud Computing**

* Virtual machines simulate hardware/software environments.
* Hypervisors manage virtual machine operations.
* Type 1 hypervisors run directly on hardware.
* Type 2 hypervisors operate atop an OS.
* Containers use shared host OS resources.
* Docker and Kubernetes are popular container tools.
* Cloud computing offers scalable IT resources online.
* IaaS provides virtual hardware for users.
* PaaS enables application development on shared platforms.
* SaaS delivers software through the cloud.
* DaaS centralizes desktop management on servers.
* Public clouds serve multiple organizations.
* Private clouds cater to single organizations.
* Hybrid clouds combine public and private models.
* Community clouds share resources among similar entities.

**Day 7: Networking Basics**

* NIC connects devices to networks.
* Switches forward data to intended devices.
* Routers manage connections between networks.
* Access Points enable wireless communication.
* CAT cables connect Ethernet devices.
* OSI Model defines seven network communication layers.
* TCP ensures reliable data transmission.
* UDP prioritizes speed over reliability.
* DNS translates domain names to IP addresses.
* DHCP assigns IP addresses automatically.
* Firewalls block unauthorized traffic.
* IDS monitors network threats.
* IPS intercepts and stops malicious activities.
* VPNs secure data over public networks.
* Patch panels organize LAN cable connections.

**Day 8: Network Topologies and Protocols**

* **Bus Topology**: Devices share a single communication line.
* **Star Topology**: Devices connect through a central hub or switch.
* **Ring Topology**: Devices connect in a closed loop.
* **Mesh Topology**: Every device connects to every other device.
* **Tree Topology**: Combines star and bus topologies hierarchically.
* **Point-to-Point Topology**: Direct link between two devices.
* **LAN**: Limited area networks (homes, offices).
* **WAN**: Wide geographical networks (internet).
* **PAN**: Personal devices (Bluetooth).
* **MAN**: Networks covering cities or metropolitan areas.
* **SAN**: Storage-specific networks for servers.
* **HTTP**: Protocol for loading web pages.
* **TCP/IP**: Ensures reliable data transfer.
* **Syslog**: Centralized event log management.
* **Dynamic Ports**: Temporary ports for client devices.

**Day 9: Network Security**

* **Firewalls**: Allow or block network traffic.
* **Intrusion Detection Systems (IDS)**: Passive threat detection.
* **Intrusion Prevention Systems (IPS)**: Actively block threats.
* **Unified Threat Management (UTM)**: Combines IDS, IPS, and firewall.
* **Authentication**: Validates user identities.
* **Authorization**: Manages user access permissions.
* **Accounting**: Tracks user activity on the network.
* **Encryption**: Secures data from unauthorized access.
* **VPNs**: Establish secure connections over public networks.
* **Phishing**: Social engineering attacks for sensitive information.
* **Ransomware**: Encrypts data until a ransom is paid.
* **Rootkits**: Conceal unauthorized access or malware.
* **Trojans**: Appear legitimate but perform malicious actions.
* **Botnets**: Networks of compromised devices.
* **Man-in-the-Middle (MITM)**: Intercepts communications.

**Day 10: Programming Basics**

* **Text Editors**: Simple code-writing tools (e.g., Notepad).
* **IDEs**: Integrated Development Environments streamline coding.
* **Source Code Editors**: Edit and manage program code.
* **Syntax Highlighting**: Differentiates commands visually.
* **Debugging Tools**: Identify and fix code errors.
* **Breakpoints**: Pause code execution at specific points.
* **Memory Inspectors**: Track memory use during debugging.
* **Compiled Languages**: Pre-compiled for performance (e.g., C++).
* **Interpreted Languages**: Line-by-line execution (e.g., Python).
* **Bootstrapping**: Compilers compiling themselves.
* **Object-Oriented Programming (OOP)**: Code reuse with classes.
* **Technology Stacks**: Set of technologies for development.
* **Front-End Stacks**: UI development (HTML, CSS, JS).
* **Back-End Stacks**: Server-side programming (Python, Node.js).
* **Full-Stack Development**: Combines front-end and back-end.

**Day 11: Technology Stacks**

* **LAMP**: Linux, Apache, MySQL, PHP (backend).
* **WAMP**: Windows version of LAMP.
* **MEAN**: MongoDB, Express, Angular, Node.js.
* **MERN**: MongoDB, Express, React, Node.js.
* **MEVN**: MongoDB, Express, Vue.js, Node.js.
* **Front-End Frameworks**: React, Angular, Vue.js.
* **Back-End Frameworks**: Django, Express, Spring.
* **Web Servers**: Apache, Nginx, IIS.
* **Databases**: MySQL, MongoDB, CouchDB.
* **Event Messaging**: Kafka for real-time processing.
* **Cloud Providers**: AWS, Google Cloud, Azure.
* **Virtualization**: Docker for containers.
* **Mobile Apps**: Android (Java), iOS (Swift).
* **Reliability**: Tested frameworks improve performance.
* **Scalability**: Easily handle growing traffic.

**Day 12: Security Risks**

* **Viruses**: Malicious code replicates and spreads.
* **Worms**: Self-replicating malware spreads via networks.
* **Trojan Horses**: Disguised as legitimate software.
* **Rootkits**: Hide malware activity.
* **Ransomware**: Blocks access until ransom is paid.
* **Phishing**: Tricks users into providing sensitive data.
* **SQL Injection**: Malicious queries for database access.
* **MITM Attacks**: Intercept communication between parties.
* **Privilege Escalation**: Gain unauthorized system access.
* **Insider Threats**: Attacks from within the organization.
* **Botnets**: Controlled devices execute attacks.
* **Armored Viruses**: Evade detection with complex coding.
* **Stealth Viruses**: Mask their presence.
* **Polymorphic Viruses**: Change their code to avoid detection.
* **Macro Viruses**: Use Office macros to execute attacks.

**Day 13: Software Development**

* **Functional Requirements**: Define system functionality.
* **Non-Functional Requirements (NFRs)**: Define performance metrics.
* **Performance**: Speed of software under workloads.
* **Scalability**: Ability to handle growth in users/data.
* **Portability**: Compatibility with different environments.
* **Compatibility**: Works alongside other systems.
* **Reliability**: Minimal failures under predefined conditions.
* **Maintainability**: Ease of fixing or improving software.
* **Availability**: Consistent user access to systems.
* **Security**: Protect against attacks or breaches.
* **Usability**: Intuitive user experience.
* **Application Virtualization**: Run older applications.
* **Hyperconverged Infrastructure**: Fully integrated resources.
* **Containers**: Isolated application environments.
* **Testing Environments**: Safe zones for new code.

**Day 14: Memory and Cache**

* **RAM**: Volatile memory for active data processing.
* **Cache Memory**: Small, fast memory near the CPU.
* **L1 Cache**: Fastest but smallest memory level.
* **L2 Cache**: Larger but slower than L1.
* **L3 Cache**: Shared across CPU cores.
* **Virtual RAM**: Disk storage used as additional RAM.
* **SRAM**: Fast, expensive, static memory.
* **DRAM**: Dynamic memory requiring refresh cycles.
* **DDR Variants**: DDR1 to DDR5 for faster speeds.
* **DIMM**: Dual-inline memory modules for desktops.
* **SO-DIMM**: Compact memory for laptops.
* **Multichannel Memory**: Improves bandwidth.
* **Error Detection**: Parity checks corrupted bits.
* **ECC Memory**: Corrects errors for reliability.
* **ROM Variants**: PROM, EPROM for permanent storage.

**Day 15: Storage and RAID Configurations**

* **HDD**: Mechanical drives using spinning disks.
* **SSD**: Faster flash memory with no moving parts.
* **Hybrid Drives (SSHD)**: Combines SSD speed and HDD capacity.
* **SATA**: Standard connection for modern drives.
* **PCIe SSD**: Supports NVMe for high-speed transfer.
* **mSATA**: Compact SSDs for portable devices.
* **M.2 Drives**: Supports both SATA and PCIe interfaces.
* **RAID 0**: Disk striping for speed but no redundancy.
* **RAID 1**: Disk mirroring for redundancy.
* **RAID 5**: Spreads data and parity across drives.
* **RAID 6**: Double parity for additional redundancy.
* **RAID 10**: Combines RAID 1 and RAID 0 benefits.
* **NAS**: Network storage accessed by multiple users.
* **Optical Discs**: CDs, DVDs for portable storage.
* **Flash Drives**: Small, convenient, portable storage.

**Day 16: Virtualization**

* **Virtual Machines (VMs)**: Simulate hardware environments.
* **Hypervisor Type 1**: Direct hardware management.
* **Hypervisor Type 2**: Runs atop an existing OS.
* **VM Benefits**: Scalability, isolation, and disaster recovery.
* **Docker**: Lightweight containerization technology.
* **Kubernetes**: Manages container clusters.
* **Virtual Sandbox**: Isolated testing environments.
* **VDI**: Virtual Desktop Infrastructure centralizes desktops.
* **Cross-Platform Virtualization**: Run software on multiple OS.
* **Resource Allocation**: Assign CPU, memory to VMs.
* **Process Isolation**: Protects host system resources.
* **Snapshot Feature**: Save VM states for rollback.
* **Cloud Integration**: Deploy VMs in cloud environments.
* **Application Virtualization**: Run old apps in new environments.
* **Security Benefits**: Limit attacker access to VM resources.

**Day 17: Cloud Computing**

* **Definition**: Delivery of computing resources over the internet.
* **Public Cloud**: Shared resources for multiple organizations.
* **Private Cloud**: Dedicated resources for single organizations.
* **Hybrid Cloud**: Combines public and private models.
* **Community Cloud**: Shared by organizations with similar needs.
* **IaaS**: Virtualized infrastructure (e.g., AWS EC2).
* **PaaS**: Platform for developing and deploying applications.
* **SaaS**: Fully hosted applications (e.g., Google Workspace).
* **DaaS**: Centralized desktops provided as a service.
* **High Availability**: Minimal downtime with 99.999% uptime.
* **Elasticity**: Real-time scaling up/down of resources.
* **Metered Service**: Pay only for resources used.
* **Scalability**: Increase or decrease IT capacity.
* **File Synchronization**: Access data from multiple locations.
* **Security Considerations**: Encryption, multitenancy concerns.

**Day 18: Networking and Ports**

* **Switch**: Forwards data to intended devices.
* **Router**: Directs traffic between networks.
* **Access Point**: Enables wireless device connections.
* **Hub**: Broadcasts data to all devices (outdated).
* **Firewall**: Blocks unauthorized network traffic.
* **IPv4**: Four-number address format.
* **IPv6**: Eight sets of hexadecimal digits.
* **Port Numbers**: Logical connections for applications.
* **Common Ports**: FTP (21), SSH (22), HTTP (80).
* **DNS**: Resolves domain names to IP addresses.
* **DHCP**: Dynamically assigns IP addresses.
* **Syslog Protocol**: Centralized logging system.
* **RDP**: Remote desktop connection (port 3389).
* **POP3/IMAP**: Email retrieval protocols.
* **SNMP**: Network device management and alerts.

**Day 19: Advanced Networking**

* **Client-Server Model**: Centralized network structure.
* **Peer-to-Peer (P2P)**: Decentralized resource sharing.
* **Bus Topology**: Single shared network cable.
* **Star Topology**: Central hub connects devices.
* **Ring Topology**: Devices form a closed loop.
* **Mesh Topology**: Devices interconnected for redundancy.
* **Tree Topology**: Hierarchical network flow.
* **Personal Area Network (PAN)**: Small device-centric networks.
* **LAN**: Covers small areas like homes or offices.
* **WAN**: Connects devices across vast distances.
* **SAN**: Storage-specific network infrastructure.
* **TCP/IP Model**: Foundation for internet communication.
* **HTTP/HTTPS**: Protocols for web pages and secure connections.
* **Network Security Layers**: Physical, data link, and application.
* **Unified Threat Management**: Combines multiple security features.

**Day 20: Final Review and Practice**

* Review **Operating Systems**: Key functionalities and types.
* Recap **Hardware Components**: Motherboards, CPUs, RAM.
* Revisit **Storage Devices**: SSD, RAID, and NAS.
* Summarize **Virtualization**: Hypervisors, VMs, containers.
* Highlight **Cloud Computing**: Deployment models and benefits.
* Revise **Programming**: IDEs, OOP, and technology stacks.
* Refresh **Networking Basics**: Topologies and devices.
* Reassess **Security Risks**: Viruses, ransomware, and phishing.
* Recap **Ports and Protocols**: Common ports and their purposes.
* Review **NFRs**: Scalability, reliability, and usability.
* Take **Practice Quizzes**: Address weaker areas.
* Revise **Key Definitions**: Core concepts and terms.
* Focus on **Case Scenarios**: Apply concepts practically.
* Take [D386 Practice Questions](https://wguashkin.github.io/D386-Practice-Questions/)