Chapter 2 – Short Questions with Answers

Q1: Enlist the functions of an operating system?

- **Process Management** – manages running programs (CPU time).
- **Memory Management** – allocates RAM fairly.
- **File Management** – creates, deletes, organizes files.
- **Device Management** – controls input/output devices.
- **Security Management** – protects data and processes.
  □ *Example: Windows handles printing (I/O), running MS Word (process), and storing documents (file system).*

Q2: Describe the functions of an operating system?

- **Resource Allocation:** Distributes CPU, memory, I/O among programs.
- **Security:** Prevents unauthorized access.
- **Communication:** Allows inter-process communication.
- **Error Detection:** Detects hardware/software errors.
  □ *Example: Linux server runs multiple websites by allocating CPU and RAM resources properly.*

Q3: Explain memory management techniques?

- **Partitioning:** Divides memory into fixed or variable parts.
- **Paging:** Breaks memory into equal-size pages.
- **Segmentation:** Divides program logically (code, data, stack).
- **Virtual Memory:** Extends RAM using disk space.
  □ *Example: When running many Chrome tabs, OS uses paging + virtual memory.*

Q4: Describe virtual memory?

- Allows execution of programs larger than RAM.
- Uses **disk storage as extension of RAM.**
- Provides illusion of large memory space.
  □ *Example: Windows creates a "pagefile.sys" on hard disk when RAM is full.*

Q5: Describe hierarchical directory system?

- Directory system arranged in **tree structure.**
- Root directory at top, subdirectories under it.
- Easier file organization and searching.
  □ *Example:* `C:\Users\Ali\Documents\file.docx` *→ Root (C:), Users, Ali, Documents.*

Q6: Describe UNIX file system?

- Based on **hierarchical directory structure.**
- Everything (files, devices) is treated as a **file.**
- Provides security using permissions (Read, Write, Execute).
  □ *Example:* `/home/student/osbook.pdf` *stored under UNIX root.*

Q7: List five services provided by an operating system?

- Process Management
- File Management
- Memory Management
- Device Management
- Error Detection
  □ *Example: Android OS provides these services to apps like WhatsApp or Chrome.*

Q8: Explain how the operating system manages the resources of a computer?

- **CPU:** Scheduling processes (round robin, priority).
- **Memory:** Allocates space, prevents interference.

- **I/O Devices:** Controlled through device drivers.
- **Files:** Organizes data storage.
  □ *Example: In gaming, OS gives CPU to game process, GPU driver for graphics, and audio device driver for sound.*

Q9: Explain disk management system?

- Divides disk into partitions.
- Provides file system formatting (NTFS, FAT32, ext4).
- Manages free space, bad sectors.
  □ *Example: Disk Management Tool in Windows allows resizing partitions.*

Q10: Define system calls and also describe what is the purpose of system calls?

- **Definition:** Interface between **user programs** and **OS services.**
- **Purpose:** To request OS services like file handling, process creation, or I/O operations.
  □ *Example:* `open()` *system call in Linux opens a file.*

Q11: Write down the steps and diagram for system call process?

- User program requests a service (e.g., read file).
- System call transfers control to OS kernel.
- OS performs the requested task.
- OS returns result to program.
  □ *Example: In C,* `printf()` *internally uses* `write()` *system call.*

Q12: Why do you need system calls in operating system?

- Provide safe access to hardware resources.
- Allow program interaction with OS.
- Without system calls, users would directly access hardware (unsafe).
  □ *Example: User program cannot directly write to printer; it uses system calls.*

Q13: Describe Error Detection in operating system?

- OS detects and handles errors during execution.
- Errors may be in memory, CPU, I/O devices, or programs.
- Provides messages or corrective action.
  □ *Example: "Blue Screen" in Windows indicates system error detection.*

Q14: Enumerate OS services and explain the service?

- Process Management
- Memory Management
- File Handling
- Security
- Communication
  □ *Example: Antivirus uses OS services for file scanning and memory checking.*

Q15: Identify system programs that come with the operating system?

- File managers
- Editors (Notepad, vi editor)
- Compilers
- Command interpreters (Shell, CMD)
- Utilities (Disk cleanup, Antivirus)

Q16: Describe the functions of these system programs?

- File manager → Organizes files.
- Editor → Creates/modifies text files.

- Compiler → Converts source code to machine code.
- Shell → Executes user commands.
- Utility → Maintains system health.

Q17: What is Resource Allocation in operating system?

- OS distributes resources like CPU, memory, I/O among processes.
- Ensures fairness and efficiency.
  □ *Example: In a bank server, OS allocates CPU to both ATM transactions and database queries.*

Q18: Which rules for passing parameters for system call?

- Parameters may be passed via:
  - Registers.
  - Stack.
  - Memory blocks.
    □ *Example: In Linux, system calls often use registers for parameters.*

Q19: Define File Manipulation?

- Refers to creating, deleting, reading, writing, and modifying files.
- OS provides system calls for these operations.
  □ *Example: MS Word saves (writes) a file, Notepad opens (reads) a file.*

Q20: Difference between Real-time and Embedded operating system?

- **Real-time OS (RTOS):**
  - Quick response, used in time-critical apps.
  - Example: Air traffic control.
- **Embedded OS:**
  - Designed for devices with specific functions.
  - Example: OS in washing machine, microwave.

Q21: Difference between GUI and CUI?

- **GUI (Graphical User Interface):**
  - Uses windows, icons, menus.
  - Example: Windows, Android.
- **CUI (Command User Interface):**
  - Uses typed commands.
  - Example: MS-DOS, Linux Shell.

Q22: Define Program Loading and Execution?

- **Loading:** Bringing program from secondary storage (disk) into RAM.
- **Execution:** CPU runs program instructions.
  □ *Example: Opening Chrome → OS loads Chrome into RAM → CPU executes it.*

Q23: Illustrate the structure by diagrams?

- **Simple OS (Monolithic):** All services in one layer.
- **Layered OS:** Divided into levels.
- **Microkernel:** Only essential functions in kernel.
  □ *Example: Windows (Hybrid), Linux (Monolithic), MacOS (Microkernel-like).*

Q24: Enlist the design goals of the operating system?

- **Primary Goals:** Convenience for users, efficient use of hardware.
- **Secondary Goals:** Security, reliability, flexibility.
  □ *Example: Windows prioritizes ease of use, Linux prioritizes stability.*

Long Questions (Chapter 2)

Q1: Explain system components and also describe its modern goals of system components?

- **Process Management:** Handles creation, scheduling, and termination of processes.
- **Memory Management:** Allocates and deallocates RAM.
- **File System Management:** Organizes storage.
- **Device Management:** Controls I/O devices.
- **Networking:** Manages communication over networks.
- **Security & Protection:** Prevents unauthorized access.
  □ *Modern Goals:* Reliability, user-friendliness, performance, portability.

Q2: What are system calls and also describe types of system calls?

- Interface between user program and OS kernel.
- **Types:**
  - Process Control (create, terminate process).
  - File Manipulation (open, close, read, write).
  - Device Management (request/release device).
  - Information Maintenance (get system info).
  - Communication (send, receive messages).
    □ *Example: `fork()` in UNIX creates a new process.*

Q3: Write a detailed note on OS services?

- **Process Management** → multitasking.
- **Memory Management** → paging, segmentation.
- **File Management** → access control, directories.
- **Device Management** → drivers.
- **Security** → authentication, permissions.
- **Networking** → communication protocols.
- **Error Detection** → logs, alerts.
  □ *Example: Windows Event Viewer shows error detection services.*

Q4: Explain the general structure and architecture of an operating system?

- **Monolithic OS:** Single large kernel. (Example: Linux)
- **Layered OS:** Organized in layers. (Example: THE OS)
- **Microkernel:** Minimal kernel, services in user space. (Example: Minix, MacOS hybrid)
- **Modular/Hybrid OS:** Combines microkernel + monolithic. (Example: Windows, Solaris)

Q5: Explain system design and implementation in operating system?

- **Design Goals:** Security, efficiency, flexibility.
- **Implementation:**
  - Written mostly in **C language**.
  - Some parts in **assembly** (boot loader, drivers).
- **Steps:**
  1. Define goals (efficiency, reliability).
  2. Choose architecture (monolithic, microkernel).
  3. Implement kernel, device drivers, system calls.
  4. Test and debug.
     □ *Example: Linux kernel is open-source and implemented in C with some assembly parts.*

External / Important Questions with Answers

Q1: Differentiate between Monolithic, Layered, and Microkernel OS structures.

- **Monolithic:** One large kernel, fast but complex (Linux).
- **Layered:** Divided into layers, easier to debug (THE OS).
- **Microkernel:** Only essential functions in kernel, rest in user space (Minix, MacOS hybrid).

Q2: Explain Process Management with example.

- Process = Running program.
- OS schedules processes using algorithms (FCFS, Round Robin).
  □ *Example: Playing music + browsing internet simultaneously.*

Q3: What is File System Management?

- OS organizes, stores, retrieves, and protects files.
- Provides naming, directories, access permissions.
  □ *Example: NTFS in Windows, ext4 in Linux.*

Q4: Define Shell and its role.

- A shell is a command interpreter between user and OS.
- Types: CLI shell (Bash), GUI shell (Windows Explorer).
  □ *Example: `ls` command in Linux shell lists files.*

Q5: What are System Utilities?

- Programs that maintain and manage system.
- Examples: Disk cleanup, backup tool, antivirus, defragmenter.

Q6: Differentiate Multiprogramming and Multitasking.

- **Multiprogramming:** Multiple programs in memory, CPU executes one at a time.
- **Multitasking:** CPU switches quickly between programs, user feels both run together.