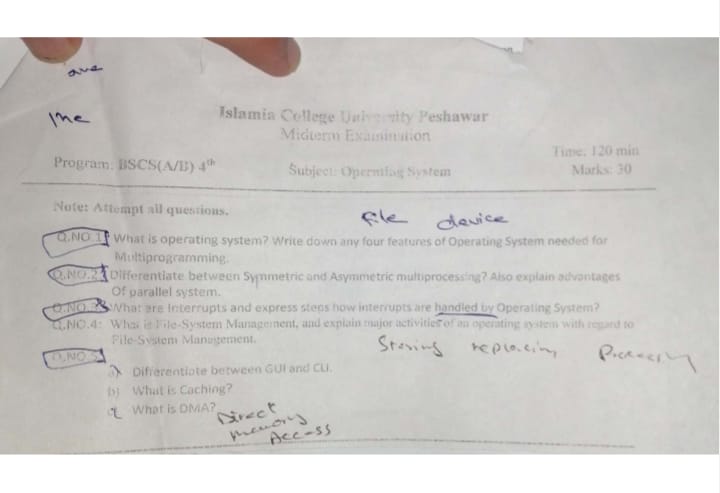
**Q1: What is an Operating System? Write down any four features of an Operating System needed for Multiprogramming.**

**What is an Operating System (OS)?**

An **Operating System (OS)** is a system software that acts as an interface between computer hardware and users. It manages computer resources, executes applications, and ensures smooth functioning. Without an OS, a computer cannot function properly.

**What is Multiprogramming?**

Multiprogramming is a method where multiple programs reside in memory and are executed by the CPU one by one. The OS switches between programs, making efficient use of the CPU. It ensures that while one program is waiting (e.g., for user input), the CPU executes another program.

**Four Features of an OS Needed for Multiprogramming**

1. **CPU Scheduling:**
   * Since multiple programs are in memory, the OS must decide which process should use the CPU at a given time.
   * It does this through scheduling algorithms like **Round Robin, First Come First Serve (FCFS), Shortest Job Next (SJN), etc.**
   * Example: If a user is watching a video while downloading a file, the OS ensures that the CPU efficiently switches between these tasks.
2. **Memory Management:**
   * The OS must allocate memory to different programs while ensuring there is no overlap or wastage.
   * It uses techniques like **paging, segmentation, and virtual memory** to manage memory efficiently.
   * Example: If multiple applications (e.g., browser, text editor, and media player) are open, the OS ensures each gets enough memory.
3. **Process Management:**
   * The OS manages multiple processes by **creating, scheduling, and terminating** them.
   * It ensures that processes do not interfere with each other and that resources are shared efficiently.
   * Example: If a document is printing and a video is playing simultaneously, the OS coordinates their execution.
4. **I/O Management:**
   * Input/output (I/O) devices such as **keyboard, mouse, printer, and hard disk** need to be managed properly.
   * The OS handles data transfer between these devices and the CPU efficiently.
   * Example: A user can type in a document (keyboard input) while music is playing in the background (audio output).

**Q2: Differentiate between Symmetric and Asymmetric Multiprocessing. Also, explain the advantages of parallel systems.**

**What is Multiprocessing?**

Multiprocessing is the use of two or more processors (CPUs) in a single system to **increase efficiency and speed**. It allows multiple processes to run simultaneously, making the system faster.

**Difference Between Symmetric and Asymmetric Multiprocessing**

| **Feature** | **Symmetric Multiprocessing (SMP)** | **Asymmetric Multiprocessing (AMP)** |
| --- | --- | --- |
| **Processor Role** | All processors share work equally. | One main processor controls others. |
| **Efficiency** | More efficient since all CPUs work together. | Less efficient, as one CPU handles most work. |
| **Failure Handling** | If one CPU fails, others continue working. | If the main CPU fails, the system crashes. |
| **Example** | Modern multi-core processors like Intel Core i7. | Some embedded systems and specialized servers. |

**Advantages of Parallel Systems (Multiple CPUs Working Together)**

Great question! Let's break down **symmetric** and **asymmetric processing** in simple and clear terms:

**🧠 Symmetric vs Asymmetric Processing**

These terms refer to how **processors (CPUs)** work together in a multiprocessor system.

**🔁 Symmetric Multiprocessing (SMP)**

**Definition**:  
All processors are **equal** — they share the same memory and have **equal access** to I/O devices. Each processor can run any task, including the operating system.

**How it works**:

* All CPUs run **independently** but share the same OS.
* The **OS scheduler** assigns tasks to any available CPU.
* There is **no master/slave** relationship between CPUs.

**Example use case**: Most **modern computers**, servers, and multi-core CPUs (like quad-core or octa-core processors) use SMP.

**Advantages**:

* ✅ Better load balancing
* ✅ Easier to program (uniform structure)
* ✅ All CPUs contribute to performance

**🔀 Asymmetric Multiprocessing (AMP)**

**Definition**:  
Processors are **not equal** — one processor is designated as the **master**, while others are **slaves**.

**How it works**:

* The **master CPU** controls the system and assigns tasks.
* Slave CPUs typically perform only **specific tasks** assigned by the master.
* Usually, only the master handles the **operating system**.

**Example use case**: Used in older systems or **embedded systems** where one processor manages control and others handle specific duties (like I/O processing or real-time tasks).

**Advantages**:

* ✅ Simpler OS design
* ✅ Can dedicate processors to critical tasks

**🆚 Comparison Table**

| **Feature** | **Symmetric (SMP)** | **Asymmetric (AMP)** |
| --- | --- | --- |
| Processor Role | All processors are equal | One master, others are slaves |
| OS Handling | Shared OS among all CPUs | Only master handles OS |
| Task Assignment | Any CPU can run any task | Master assigns tasks |
| Performance | High scalability and balance | Less scalable |
| Complexity | More complex system design | Simpler system design |
| Usage | Modern PCs, servers | Embedded systems, older computers |

1. **Increased Speed and Performance:**
   * More CPUs mean more tasks can be completed at the same time, reducing execution time.
2. **Better Resource Utilization:**
   * Each processor can handle a different task, making better use of computing resources.
3. **Improved System Reliability:**
   * If one CPU fails, the system can still function with the remaining CPUs.
4. **Scalability:**
   * More CPUs can be added as needed, allowing systems to scale efficiently.

**Q3: What are Interrupts? Explain the steps of how interrupts are handled by the Operating System.**

**What is an Interrupt?**

An **interrupt** is a signal sent to the CPU by hardware or software to **stop its current task and perform a more urgent task**. After handling the interrupt, the CPU resumes its previous task.

**Types of Interrupts:**

1. **Hardware Interrupt:** Generated by hardware devices like keyboard (when a key is pressed) or printer (when printing is completed).
2. **Software Interrupt:** Generated by software programs (e.g., system calls made by an application).

**Steps in Handling an Interrupt:**

1. **Interrupt Occurs:**
   * A device or software sends an interrupt signal to the CPU.
   * Example: A user presses a key on the keyboard.
2. **Interrupt Controller Prioritizes It:**
   * If multiple interrupts occur, the **Interrupt Controller** decides which one to handle first.
3. **CPU Pauses Current Task:**
   * The CPU stops the ongoing process and **saves its state**.
4. **Interrupt Service Routine (ISR) Executes:**
   * The OS calls the appropriate **Interrupt Handler** to manage the interrupt.
5. **Process Resumes:**
   * After handling the interrupt, the CPU **resumes the previous task**.

**Q4: What is File System Management? Explain major activities of an Operating System with regard to File System Management.**

**What is File System Management?**

File System Management refers to the OS's ability to **store, organize, retrieve, and secure files** on storage devices like hard disks, SSDs, or USBs.

**Major Activities of an OS in File System Management**

1. **File Creation and Deletion:**
   * Users can create and delete files.
   * Example: Creating a new Word document or deleting an old file.
2. **Directory Management:**
   * Files are organized into directories (folders).
   * Example: A folder named "Projects" containing different files.
3. **File Access Control:**
   * The OS restricts access based on user permissions (read, write, execute).
   * Example: Some system files can only be modified by administrators.
4. **Storage Management and Allocation:**
   * The OS efficiently allocates disk space to files, preventing **fragmentation**.
5. **Backup and Recovery:**
   * Important files are backed up to prevent data loss.
   * Example: Windows has a "Restore Previous Version" feature.

**Q5(a): Differentiate between GUI and CLI.**

| **Feature** | **GUI (Graphical User Interface)** | **CLI (Command Line Interface)** |
| --- | --- | --- |
| **Ease of Use** | Easy, uses windows and icons. | Harder, requires typing commands. |
| **Speed** | Slower (requires more resources). | Faster (less processing power needed). |
| **Flexibility** | Limited to available options. | More powerful and flexible. |
| **Example** | Windows, macOS, Android. | Linux Terminal, Command Prompt. |

**Q5(b): What is Caching?**

**Definition:**

Caching is the **temporary storage of frequently accessed data** to speed up processing.

**How Caching Works:**

* Frequently used data is stored in **cache memory**, which is faster than RAM and hard drives.
* Example: A web browser stores frequently visited websites so they load faster.

**Advantages of Caching:**

1. Reduces access time.
2. Improves CPU efficiency.
3. Reduces data transfer delays.

**Q5(c): What is DMA (Direct Memory Access)?**

**Definition:**

DMA is a technique where **data is transferred between memory and devices without using the CPU**.

**How DMA Works:**

* Normally, the CPU handles data transfer, which slows it down.
* DMA allows devices like hard drives and network adapters to transfer data directly to RAM.

**Benefits of DMA:**

1. **Faster Data Transfer:** The CPU is free for other tasks.
2. **Efficient for Large Data Transfers:** Used in disk operations and network communication.