OS Interview Questions

1. **Define the types of Operating System?**

Ans :: 1. Batch Operating System:

- Processes jobs in batches without user interaction
- Efficient for repetitive tasks, but lacks interactivity

2. Multi-Programming Operating System:

- Allows multiple programs to run simultaneously
- Improves CPU utilization by switching between programs

3. Multi-Tasking Operating System:

- Enables multiple tasks or processes to run concurrently
- Provides the illusion of simultaneous execution through time-sharing

4. Multi-Processing Operating System:

- Utilizes multiple processors to handle tasks
- Improves overall system performance and reliability

5. Real-Time Operating System:

- Designed for time-critical applications
- Guarantees task completion within specific time constraints

6. Distributed Operating System:

- Manages a group of independent computers
- Appears to users as a single coherent system

7. Network Operating System:

- Manages network resources and provides network services
- Facilitates communication between different computers on a network

8. Mobile Operating System:

- Designed specifically for mobile devices like smartphones and tablets
- Optimized for touch interfaces and mobile hardware

---

2. **Explain DHCP?**

Ans :: DHCP (Dynamic Host Configuration Protocol) is a network protocol used to automatically assign IP addresses and other network configuration parameters to devices on a network.

3. Explain DNS?

Ans :: DNS (Domain Name System) is a hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network. Its main functions include:

1. Translating human-readable domain names (e.g., www.example.com) into IP addresses (e.g., 192.0.2.1) that computers use to identify each other on the network.
2. Providing a distributed database of domain names and their associated IP addresses, ensuring scalability and reliability.

4. Explain paging?

Ans :: Paging is a memory management scheme used in operating systems to manage computer memory.

1. It divides physical memory into fixed-size blocks called frames and logical memory into blocks of the same size called pages.
2. Paging allows non-contiguous allocation of physical memory, which helps in reducing external fragmentation.
3. Paging improves memory utilization and allows for efficient multitasking by sharing physical memory among multiple processes.

5. Explain segmentation?

Ans :: Segmentation is a memory management technique used in operating systems that divides the computer's memory into segments of varying sizes.

1. Each segment is a contiguous block of memory that represents a specific part of a program or data structure (e.g., code segment, data segment, stack segment).
2. Segments can be of different sizes, allowing for more flexible memory allocation compared to fixed-size paging.
3. It provides better protection and sharing of code and data between different processes, as each segment can have its own access rights.

4. Segmentation supports logical organization of memory, making it easier for programmers to conceptualize memory layout.

---

6. Explain memory management?

Ans :: Memory management is a crucial function of operating systems that involves organizing and controlling computer memory.

It encompasses several key aspects:

1. Allocation: Assigning portions of memory to processes and programs as they request it.
2. Tracking: Keeping track of which parts of memory are in use and which are free.
3. Freeing: Releasing memory when it's no longer needed, making it available for other processes.
4. Protection: Ensuring that processes can only access memory allocated to them, preventing interference between different programs.

Memory management techniques include:

- Paging: Dividing memory into fixed-size blocks for efficient allocation.
- Segmentation: Organizing memory into variable-sized segments based on logical divisions of programs.
- Virtual Memory: Using disk space to extend the available RAM, allowing programs to use more memory than physically available.
- Garbage Collection: Automatically freeing memory that is no longer in use in some programming languages.

---

7. Explain the function of OS?

Ans ::

1. Resource Management: Allocates and manages system resources such as CPU time, memory, storage, and I/O devices among different programs and users.
2. Process Management: Creates, schedules, and terminates processes, ensuring efficient multitasking and fair resource allocation.

3. Memory Management: Allocates and deallocates memory space to programs, manages virtual memory, and ensures memory protection between processes.
4. File System Management: Organizes and maintains the file system, handling file creation, deletion, access, and storage.
5. Device Management: Manages device communication through drivers, controlling input/output operations for various hardware devices.
6. User Interface: Provides a means for users to interact with the computer, either through a command-line interface (CLI) or graphical user interface (GUI).
7. Security and Protection: Implements security measures to protect the system from unauthorized access and ensures data integrity.
8. Networking: Manages network connections and protocols, enabling communication between different computers and devices.
9. Error Handling: Detects and responds to various types of errors, ensuring system stability and reliability.

---

8. Explain a kernel? Its architecture and working?

Ans :: A kernel is the core component of an operating system that acts as a bridge between applications and hardware. It manages system resources and provides essential services to other parts of the operating system.

Architecture:

1. Monolithic Kernel: All operating system services run in kernel space. Examples: Linux, Unix.
2. Microkernel: Minimal functionality in kernel space, with most services running in user space. Example: MINIX.
3. Hybrid Kernel: Combines aspects of both monolithic and microkernel architectures. Example: Windows NT.

Working:

1. Process Management: Schedules and manages processes, allocating CPU time.

2. Memory Management: Handles memory allocation, virtual memory, and paging.
3. Device Management: Manages device drivers and hardware interactions.
4. System Calls: Provides an interface for user-space applications to request kernel services.
5. Interrupt Handling: Manages hardware and software interrupts to ensure timely response to events.

The kernel operates in a privileged mode (kernel mode) with full access to hardware resources, ensuring efficient and secure system management.

---

9. Explain a shell script?

Ans :: A shell script is a program written for a Unix shell, which is a command-line interpreter for operating systems. Key points about shell scripts include:

1. It's a text file containing a series of commands that the shell can execute.
2. Shell scripts are used to automate tasks, run multiple commands sequentially, and create custom commands.
3. They can include variables, control structures (like loops and conditionals), and functions.

---

10. Explain a page fault?

Ans :: A page fault is an error in a computer system that occurs when a program tries to access a page in memory that is not currently mapped to physical RAM. Here are some key points about page faults:

- It's part of the virtual memory system in modern operating systems.
- When a page fault occurs, the operating system must locate the required page on disk and load it into memory.

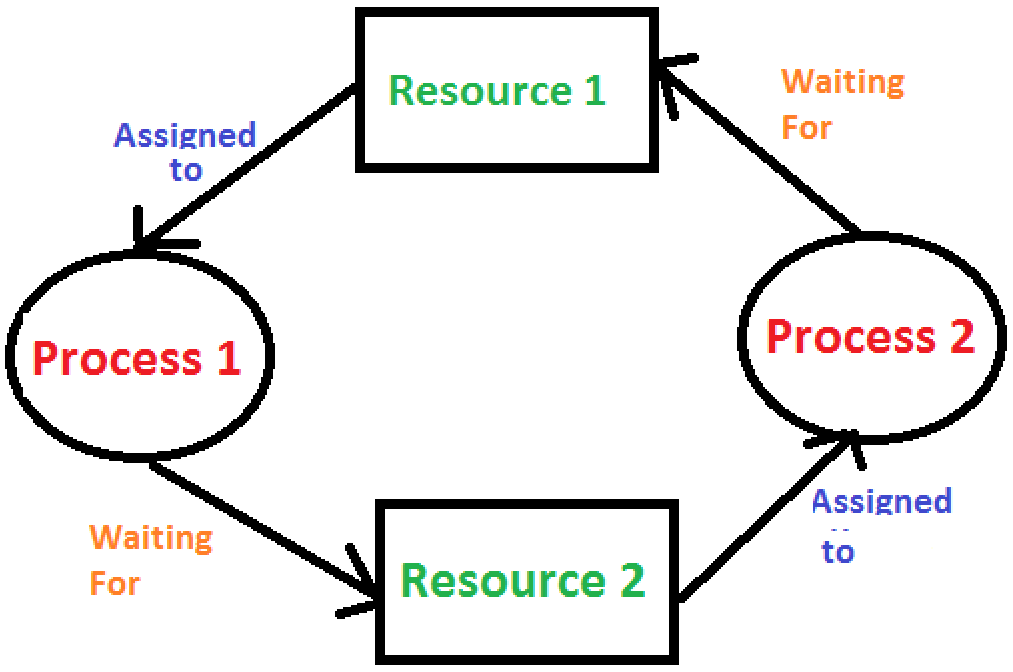
There are two main types of page faults:

- Minor: The page is in memory but not marked as present in the page table.
- Major: The page needs to be loaded from disk.

- Page faults are a normal part of system operation and help in efficiently managing memory resources.
- Excessive page faults can lead to a condition called thrashing, where the system spends more time paging than executing processes.

11. Explain a deadlock?

Ans :: A deadlock is a situation where two or more processes are unable to proceed because each is waiting for the other to release a resource.



12. Define the necessary conditions for deadlock?

Ans :: The four necessary conditions for a deadlock to occur are:

1. Mutual Exclusion: At least one resource must be held in a non-sharable mode, meaning only one process can use the resource at a time.
2. Hold and Wait: A process must be holding at least one resource while waiting to acquire additional resources held by other processes.

3. No Preemption: Resources cannot be forcibly taken away from a process; they must be released voluntarily by the process holding them.
4. Circular Wait: A circular chain of two or more processes, each waiting for a resource held by the next process in the chain.

13. Explain a semaphore?

Ans :: A semaphore is a synchronization primitive used in concurrent programming to control access to shared resources and coordinate the execution of multiple processes or threads. Key points about semaphores include:

1. Definition: It's a variable or abstract data type used to control access to a common resource by multiple processes in a concurrent system.

Types:

- Binary semaphore: Can have only two values (0 or 1), often used for mutual exclusion.
- Counting semaphore: Can have a range of non-negative values, used for resource counting.

14. Explain a mutex?

Ans :: A mutex (short for mutual exclusion) is a synchronization primitive used in concurrent programming to prevent multiple threads from simultaneously accessing a shared resource.

1. Definition: A mutex is a locking mechanism that ensures only one thread can access a particular section of code or resource at a time.
2. Purpose: It's used to avoid race conditions and ensure data integrity in multi-threaded applications.

15. Difference among kernel space and user space.

Ans :: The difference between kernel space and user space is a fundamental concept in operating system architecture. Here are the key distinctions:

1. **Definition:**
   • Kernel Space: The area of virtual memory where the kernel (core of the

operating system) executes and provides its services.
• User Space: The memory area where user processes (applications) run.

2. **Access Rights:**
   • Kernel Space: Has unrestricted access to the system's hardware and memory.
   • User Space: Has limited access to hardware and memory, requiring system calls to interact with kernel services.

3. **Protection Level:**
   • Kernel Space: Operates in a privileged mode (also called kernel mode or supervisor mode).
   • User Space: Operates in a restricted mode (user mode) with limited privileges.

4. **Memory Management:**
   • Kernel Space: Can access both kernel space and user space memory.
   • User Space: Can only access its own memory area, preventing direct interference with other processes or the kernel.

---

16. Write in brief the ping command.

Ans :: The ping command is a network utility used to test the reachability of a host on an Internet Protocol (IP) network and measure the round-trip time for messages sent from the originating host to a destination computer. Key points about the ping command include:

1. Function: It sends ICMP echo request packets to the target host and waits for ICMP echo reply packets.

2. Usage: Typically used for network troubleshooting to verify connectivity and assess network latency.

3. Syntax: In most systems, the basic syntax is "ping [options] host", where host can be an IP address or domain name.

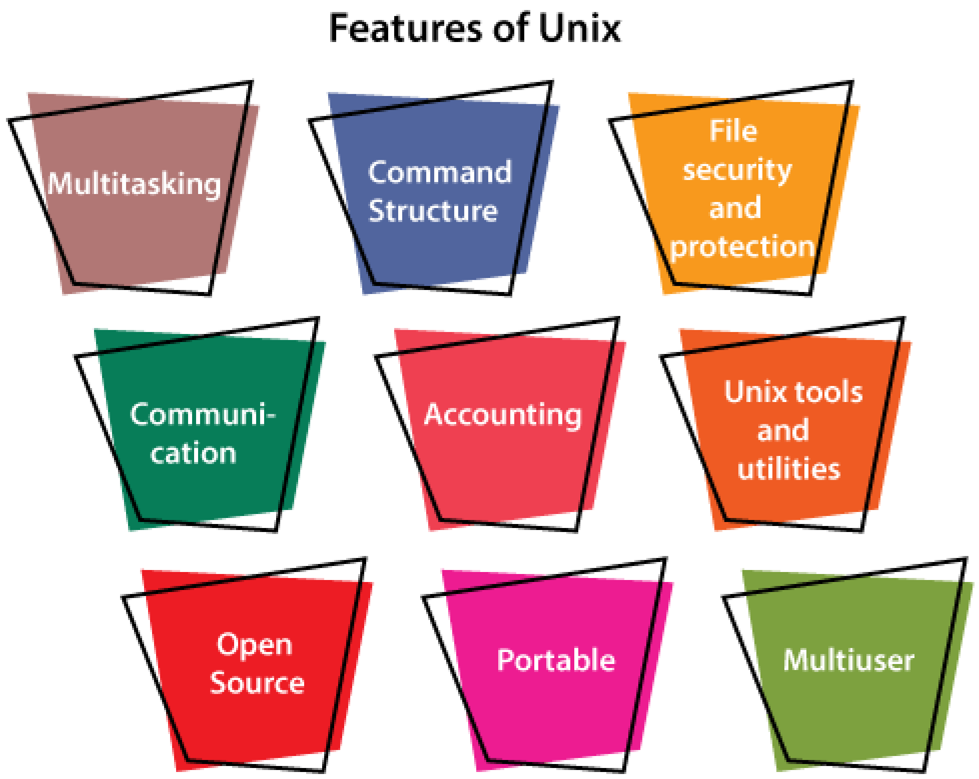
The ping command is available on most operating systems, including Windows, macOS, and Linux, making it a universal tool for basic network diagnostics.

---

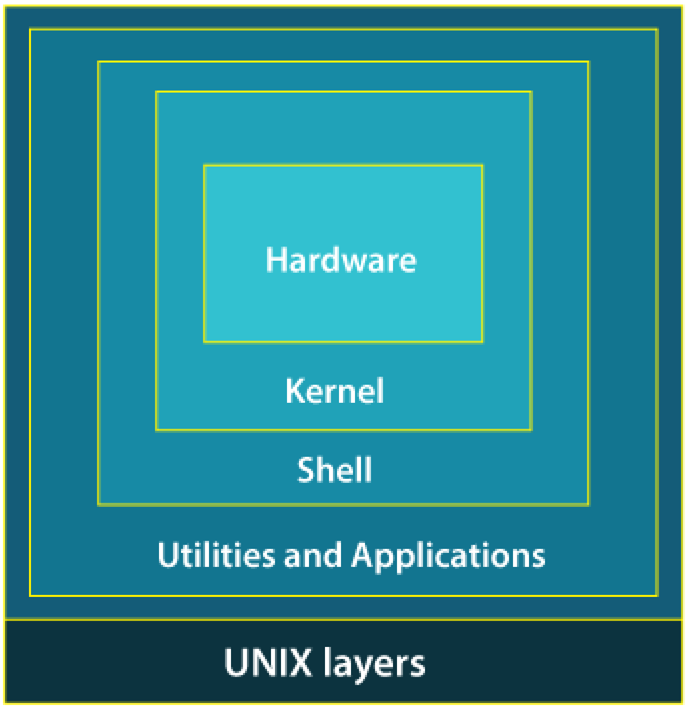
17. Explain UNIX?

Ans :: UNIX is a powerful Operating System initially developed by Ken Thompson, Dennis Ritchie at AT&T Bell laboratories in 1970. It is prevalent among scientific, engineering, and academic institutions due to its most appreciative features like multitasking, flexibility, and many more. In UNIX, the file system is a hierarchical structure of files and directories where users can store and retrieve information using the files.

Features of UNIX Operating System:



The structure of Unix OS Layers are as follows:



18. Explain grep?

Ans :: The `grep` command in Unix/Linux is a powerful tool used for searching and manipulating text patterns within files. Its name is derived from the ed (editor) command g/re/p (globally search for a regular expression and print matching lines), which reflects its core functionality. `grep` is widely used by programmers, system administrators, and users alike for its efficiency and versatility in handling text data. In this article, we will explore the various aspects of the `grep` command.

Syntax of grep Command in Unix/Linux

The basic syntax of the ``grep`` command is as follows:

```
grep [options] pattern [files]
```

Here,

`[ options ]`: These are command-line flags that modify the behavior of `grep`.

`[ pattern ]`: This is the regular expression you want to search for.

`[ file ]`: This is the name of the file(s) you want to search within. You can specify multiple files for simultaneous searching.

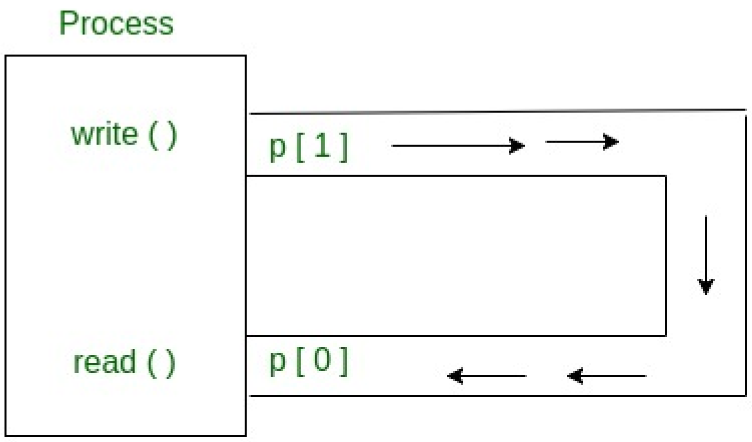
Example

```
cat > geekfile.txt
```

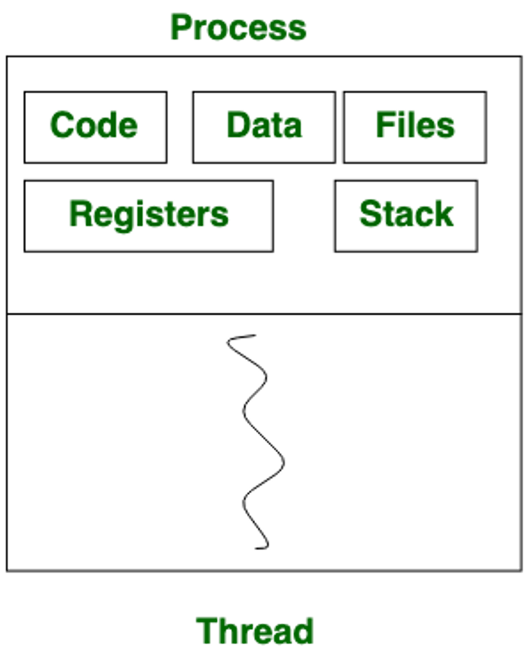
19. Explain pipe?

Ans :: A pipe is a connection between two processes, such that the standard output from one process becomes the standard input of the other process. In UNIX Operating System, Pipes are useful for communication between related processes(inter-process communication).

- Pipe is one-way communication only i.e we can use a pipe such that One process write to the pipe, and the other process reads from the pipe. It opens a pipe, which is an area of main memory that is treated as a ***"virtual file"***.
- The pipe can be used by the creating process, as well as all its child processes, for reading and writing. One process can write to this "virtual file" or pipe and another related process can read from it.



20. Difference among Thread & Process.



Process	Thread
Process means any program is in execution.	Thread means a segment of a process.
The process takes more time to terminate.	The thread takes less time to terminate.
It takes more time for creation.	It takes less time for creation.
It also takes more time for context switching.	It takes less time for context switching.
The process is less efficient in terms of communication.	Thread is more efficient in terms of communication.

21. Explain a scheduling algorithm?

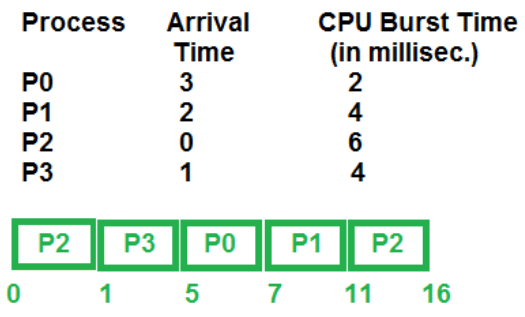
CPU Scheduling is a process that allows one process to use the CPU while another process is delayed (in standby) due to unavailability of any resources such as I / O etc, thus making full use of the CPU. The purpose of CPU Scheduling is to make the system more efficient, faster, and fairer.

CPU scheduling is a key part of how an operating system works. It decides which task (or process) the CPU should work on at any given time. This is important because a CPU can only handle one task at a time, but there are usually many tasks that need to be processed.

22. Explain pre-emptive and non-preemptive scheduling?

Preemptive scheduling is used when a process switches from the running state to the ready state or from the waiting state to the ready state. The resources (mainly CPU cycles) are allocated to the process for a limited amount of time and then taken away, and the process is again placed back in the ready queue if that process still has CPU burst time remaining. That process stays in the ready queue till it gets its next chance to execute.

Algorithms based on preemptive scheduling are **Round Robin (RR)**, **Shortest Remaining Time First (SRTF)**, **Priority (preemptive version)**, etc.

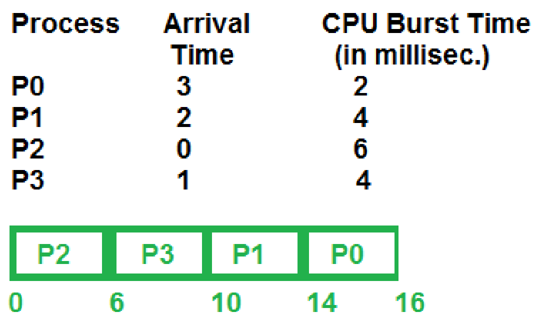


Process	Arrival Time	CPU Burst Time (in millisec.)
P0	3	2
P1	2	4
P2	0	6
P3	1	4

Preemptive Scheduling

Non-preemptive Scheduling is used when a process terminates, or a process switches from running to the waiting state. In this scheduling, once the resources (CPU cycles) are allocated to a process, the process holds the CPU till it gets terminated or reaches a waiting state. In the case of non-preemptive scheduling does not interrupt a process running CPU in the middle of the execution. Instead, it waits till the process completes its CPU burst time, and then it can allocate the CPU to another process.

Algorithms based on non-preemptive scheduling are: **Shortest Job First (SJF basically non preemptive)** and **Priority (nonpreemptive version)**, etc.



Process	Arrival Time	CPU Burst Time (in millisec.)
P0	3	2
P1	2	4
P2	0	6
P3	1	4

Non-Preemtive Scheduling

23. **Define the different scheduling algorithms**

1. First Come First Serve

It is the simplest algorithm to implement. The process with the minimal arrival time will get the CPU first. The lesser the arrival time, the sooner will the process gets the CPU. It is the non-preemptive type of scheduling.

2. Round Robin

In the Round Robin scheduling algorithm, the OS defines a time quantum (slice). All the processes will get executed in the cyclic way. Each of the process will get the CPU for a small amount of time (called time quantum) and then get back to the ready queue to wait for its next turn. It is a preemptive type of scheduling.

ADVERTISEMENT

3. Shortest Job First

The job with the shortest burst time will get the CPU first. The lesser the burst time, the sooner will the process get the CPU. It is the non-preemptive type of scheduling.

4. Shortest remaining time first

It is the preemptive form of SJF. In this algorithm, the OS schedules the Job according to the remaining time of the execution.

5. Priority based scheduling

In this algorithm, the priority will be assigned to each of the processes. The higher the priority, the sooner will the process get the CPU. If the priority of the two processes is same then they will be scheduled according to their arrival time.

6. Highest Response Ratio Next

In this scheduling Algorithm, the process with highest response ratio will be scheduled next. This reduces the starvation in the system.

24. **Explain booting process?**

Booting is the process of starting a computer. It can be initiated by hardware such as a button press or by a software command. After it is switched on, a CPU has no software in its main memory, so some processes must load software into memory before execution. This may be done by hardware or firmware in the CPU or by a separate processor in the computer system.

25. **Explain bias?**

In operating systems (OS), "bias" generally refers to a preference or prioritization that the system or its components show towards certain processes, tasks, or resources. Bias can manifest in various forms within an OS, often affecting the performance, efficiency, and fairness of system operations. Here are some common areas where bias can occur in an OS:

1. Scheduling Bias:

- **CPU Scheduling:** The OS scheduler decides which process gets to use the CPU and for how long. Certain scheduling algorithms can introduce bias. For example:
  - **First-Come, First-Served (FCFS):** Can be biased towards processes that arrive earlier.
  - **Shortest Job Next (SJN):** Biased towards shorter processes, potentially starving longer ones.
  - **Priority Scheduling:** Higher priority processes may get more CPU time, leading to starvation of lower priority processes.

2. Resource Allocation Bias:

- **Memory Management:** The way memory is allocated can favor certain processes, especially in systems using techniques like paging or segmentation.
- **I/O Device Management:** Certain processes may get more frequent or faster access to I/O devices, leading to a bias in I/O operations.

3. System Call Handling Bias:

- **Latency in Handling System Calls:** Some system calls may be handled more efficiently than others, leading to biases in system performance depending on the type of request being made.

26. **Explain the difference among static memory allocation and dynamic memory allocation?**

S.No	Static Memory Allocation	Dynamic Memory Allocation
1	In the static memory allocation, variables get allocated permanently, till the program executes or function call finishes.	In the Dynamic memory allocation, the memory is controlled by the programmer. It gets allocated whenever a malloc() is executed gets deallocated wherever the free() is executed.
2	Static Memory Allocation is done before program execution.	Dynamic Memory Allocation is done during program execution.
3	It uses **stack** for managing the static allocation of memory	It uses heap (not heap data structure) of memory for managing the dynamic allocation of memory
4	It is less efficient	It is more efficient
5	In Static Memory Allocation, there is no memory re-usability	In Dynamic Memory Allocation, there is memory re-usability and memory can be freed when not required
6	In static memory allocation, once the memory is allocated, the memory size can not change.	In dynamic memory allocation, when memory is allocated the memory size can be changed.

7	In this memory allocation scheme, we cannot reuse the unused memory.	This allows reusing the memory. The user can allocate more memory when required. Also, the user can release the memory when the user needs it.
8	In this memory allocation scheme, execution is faster than dynamic memory allocation.	In this memory allocation scheme, execution is slower than static memory allocation.
9	In this memory is allocated at compile time.	In this memory is allocated at run time.
10	In this allocated memory remains from start to end of the program.	In this allocated memory can be released at any time during the program.
11	**Example:** This static memory allocation is generally used for **array**.	**Example:** This dynamic memory allocation is generally used for **linked list**.

27. **UNIX commands like touch, sed, grep.**

1. `touch`

The `touch` command is used to create an empty file or update the timestamp of an existing file.

2. `sed`

The `sed` (Stream Editor) command is used to perform basic text transformations on an input stream (a file or input from a pipeline).

3. `grep`

The `grep` command is used to search for patterns within files. It prints lines that match a specified pattern.

28. **Explain a process and process table? Define different states of process?**

Process

A process is an instance of a program that is being executed. It contains the program code and its current activity. A process is more than just the program code; it includes a program counter, stack, registers, and variables. A program is a passive entity, such as a file containing a list of instructions stored on disk, whereas a process is an active entity, with a program counter specifying the next instruction to execute and a set of associated resources.

Process Table

The process table is a data structure maintained by the operating system to keep track of all the processes. Each entry in the process table, called a process control block (PCB), contains important information about a specific process. This includes:

- **Process ID (PID):** A unique identifier for the process.
- **Process State:** The current state of the process (e.g., running, waiting).
- **Program Counter:** The address of the next instruction to be executed.
- **CPU Registers:** The contents of all the process's registers.
- **Memory Management Information:** Information about the process's memory allocation, such as page tables or segment tables.
- **Accounting Information:** Information like CPU usage, real-time used, and process priorities.
- **I/O Status Information:** Information about I/O devices allocated to the process, open files, etc.

States of a Process

- **New:** The process is being created.
- **Ready:** The process is waiting to be assigned to a processor. It is loaded into memory and is waiting for CPU time.
- **Running:** The process is being executed on the CPU.
- **Waiting (Blocked):** The process cannot continue until some event occurs, such as the completion of an I/O operation or the availability of a resource.
- **Terminated (Exit):** The process has finished execution.

29. **Define the benefits of multithreaded programming?**

Responsiveness – Multithreading in an interactive application may allow a program to continue running even if a part of it is blocked or is performing a lengthy operation, thereby increasing responsiveness to the user.

Resource Sharing –

Processes may share resources only through techniques such as-

- Message Passing
- Shared Memory

Economy – Allocating memory and resources for process creation is a costly job in terms of time and space. Since, threads share memory with the process it belongs, it is more economical to create and context switch threads.

Minimized system resource usage – Threads have a minimal influence on the system's resources. The overhead of creating, maintaining, and managing threads is lower than a general process.

Enhanced Concurrency – Multithreading can enhance the concurrency of a multi-CPU machine. This is because the multithreading allows every thread to be executed in parallel on a distinct processor.

Reduced Context Switching Time – The threads minimize the context switching time as in Thread Context Switching, the virtual memory space remains the same.

30. **Explain Thrashing?**

Thrashing is when the page fault and swapping happens very frequently at a higher rate, and then the operating system has to spend more time swapping these pages. This state in the operating system is known as thrashing. Because of thrashing, the CPU utilization is going to be reduced or negligible.

31. **Explain Belady's Anomaly?**

Bélády's anomaly is the name given to the phenomenon where increasing the number of page frames results in an increase in the number of page faults for a given memory access pattern.

This phenomenon is commonly experienced in the following page replacement algorithms:

- First in first out (FIFO)
- Second chance algorithm
- Random page replacement algorithm

32. **Explain starvation and aging?**

Starvation or indefinite blocking is a phenomenon associated with the Priority scheduling algorithms. A process that is present in the ready state and has low priority keeps waiting for the CPU allocation because some other process with higher priority comes with due respect time. Higher-priority processes can prevent a low-priority process from getting the CPU.

Aging

In Operating systems, Aging is a scheduling technique used to avoid Starvation. Fixed priority scheduling is a scheduling discipline in which tasks queued for utilizing a system resource is assigned each priority. A task with a high priority is allowed to access a specific system resource before a task with a lower priority is allowed to do the same.

Aging is a technique of gradually increasing the priority (by time quantum) of processes that wait in the system for a long time. By doing so, as time passes, the lower priority process becomes a higher priority process.

33. **Explain a trap and trapdoor?**

Trap

A trap, also known as an exception or a software interrupt, is a mechanism used by an operating system to handle exceptional conditions or specific service requests generated during program execution. Traps are initiated by the operating system or the CPU to interrupt the normal flow of execution and switch to a special routine that handles the condition.

Trapdoor

A trap*door*, also known as a backdoor, is a hidden method for bypassing normal authentication or access controls within a computer system, software, or cryptographic system. Trapdoors are typically used by software developers for legitimate purposes such as debugging and testing, but they can also be exploited by malicious actors for unauthorized access.

34. **Explain a daemon?**

A daemon is a background process that runs continuously and performs specific operations or waits for certain events or conditions. Daemons typically start at boot time and do not have a user interface. They are essential for the functioning of many system services and can provide various functionalities such as handling network requests, managing hardware, performing scheduled tasks, and more.

35. **Which application software's executed on OS?**

- **Productivity Software:**
  - **Word Processors:** e.g., Microsoft Word, Google Docs
  - **Spreadsheets:** e.g., Microsoft Excel, Google Sheets
  - **Presentation Software:** e.g., Microsoft PowerPoint, Google Slides
- **Web Browsers:**
  - **Chrome**
  - **Firefox**
  - **Safari**
- **Multimedia Software:**
  - **Media Players:** e.g., VLC, Windows Media Player
  - **Photo Editors:** e.g., Adobe Photoshop, GIMP
  - **Video Editors:** e.g., Adobe Premiere Pro, Final Cut Pro

36. **Define daemon objects and thread objects?**

Daemon Objects

A daemon thread is a thread that runs in the background and is typically used for tasks that should not block the program from exiting. When the main program terminates, daemon threads are automatically killed, ensuring that they do not prevent the program from shutting down.

Thread Objects

A thread object represents an individual thread of execution within a program. Threads allow a program to run multiple operations concurrently in the same process space. This is particularly useful for performing tasks in the background while keeping the main program responsive.

37. **Give commands for finding process ID.**

    ps -ef

38. **How to edit, rename and move file in Linux?**

    Edit a File

    nano filename

    nano filename

    ### Rename a File

    mv old_filename new_filename

    Move a File

    mv filename /path/to/destination/

39. **Give 5 commands in Linux with explanation**

1. ls	Displays information about files in the current directory.
2. pwd	Displays the current working directory.
3. mkdir	Creates a directory.
4. cd	To navigate between different folders.
5. rmdir	Removes empty directories from the directory lists.

40. **Which are deadlock handling situations?**

    a. **Mutual Exclusion:** A resource can be used by only one process at a time. If another process requests for that resource, then the requesting process must be delayed until the resource has been released.

    b. **Hold and wait:** Some processes must be holding some resources in the non-shareable mode and at the same time must be waiting to acquire some more resources, which are currently held by other processes in the non-shareable mode.

c. **No pre-emption:** Resources granted to a process can be released back to the system only as a result of voluntary action of that process after the process has completed its task.

d. **Circular wait:** Deadlocked processes are involved in a circular chain such that each process holds one or more resources being requested by the next process in the chain.