Advanced Operating System Question Bank

1 Marker

▼ Q1. What is heap?

The heap is an area of dynamically-allocated memory that is managed automatically by the operating system or the memory manager library.

▼ Q2. What is stack?

A stack is a special area of computer’s memory which stores temporary variables created by a function. In stack, variables are declared, stored and initialized during runtime. The last data or process which is added in the stack are accessed first (LIFO).

▼ Q3. What is process table?

The process table is a data structure maintained by the operating system to facilitate context switching and scheduling, and other activities discussed later.

▼ Q4. What are pipes?

Conceptually, 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).

▼ Q5. What is Scheduler?

A scheduler is a process manager which take incharge of which process to be taken out from the CPU and which all processes to be inserted into the CPU for execution depending upon the particular strategy.

▼ Q6. What is Scheduling?

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

▼ Q7. What is Fork?

In an operating system, a fork is a Unix or Linux system call to create a new process from an existing running process. The new process is a child

process of the calling parent process.

▼ Q8. What is Demand Paging?

Demand paging is a process in which data is moved from secondary memory to RAM on a demand basis, which means all data is not stored in the **main memory** because the space is limited in RAM.

▼ Q9. What is Page Fault?

If the required page from the CPU is not present in the Main Memory then this condition is called as page miss or page fault.

▼ Q10. What is Virtual Memory?

Virtual memory is a technique in operating system in which to compensate the limited storage of main memory the non priority of task is been transferred from main memory to secondary memory which is storage area.

▼ Q11. What is Caching?

Caching is storing data in a separate disk (very fast speed disk). The data which is to be used many times results in wastage of time if it is in hard disk, but storing the data in cache reduces this time wastage.

▼ Q12. What is Buffer?

Buffer is a temporary storage area, usually a block in memory, in which items are placed while waiting to be transferred from an input device or to an output device.

▼ Q13. What are inodes?

By definition, an inode is an index node. It serves as a unique identifier for a specific piece of metadata on a given filesystem. Each piece of metadata describes what we think of as a file.

▼ Q14. What are exception?

Exceptions are unexpected events that exist somewhere in the system, the processor, or within a program that requires attention of the CPU. Types of exceptions are: processor-detected exceptions, programmed exceptions.

▼ Q15. What are interrupt?

Interrupts are unexpected events that put the normal flow of execution of instructions to a hair which prompts the OS to take immediate action. Types of interrupts: synchronous and asynchronous interrupts.

▼ Q16. What is locality of reference?

Locality of reference refers to a phenomenon in which a computer program tends to access same set of memory locations for a particular time period.

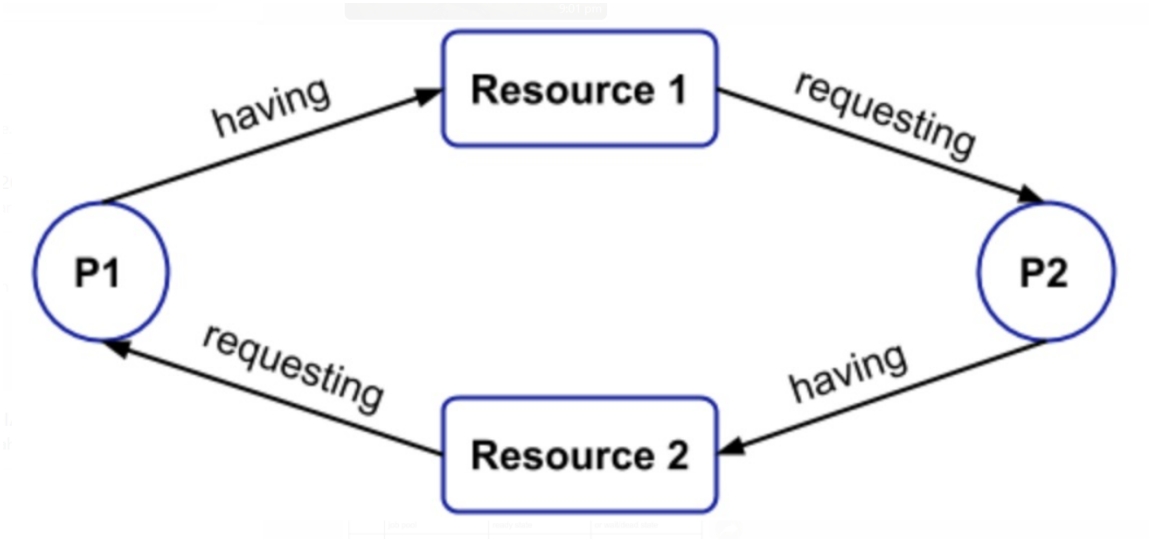
▼ Q17. What is basic difference between program and process?

Program contains a set of instructions designed to complete a specific task. Process is an instance of an executing program.

Questions

▼ Q.1 What is a Deadlock? Necessary conditions for deadlock?

Deadlock is a situation where a process already using a resource is waiting to acquire the resources held by other processes. Thus each process waits for an indefinite amount of time for an event that may or may not occur.



Necessary conditions for a deadlock to occur:

1. **Mutual exclusion**: Only one process at a time can use a resource.
2. **Hold & Wait**: A process holding at least one resource is waiting to acquire additional resources held by other processes.
3. **No preemption**: A resource can be released only voluntarily by the process holding it, after that process has completed its task.
4. **Circular wait**: There exists a set $\{P_0, P_1, ..., P_n\}$ of waiting processes such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by $P_2, ..., P_{n-1}$ is waiting for a resource that is held by P_n & P_n is waiting for a resource that is held by P_0.

▼ Q.2 What is demand paging? data structures of demand paging? Example with diagram.

Demand paging is a method of Virtual memory Management.

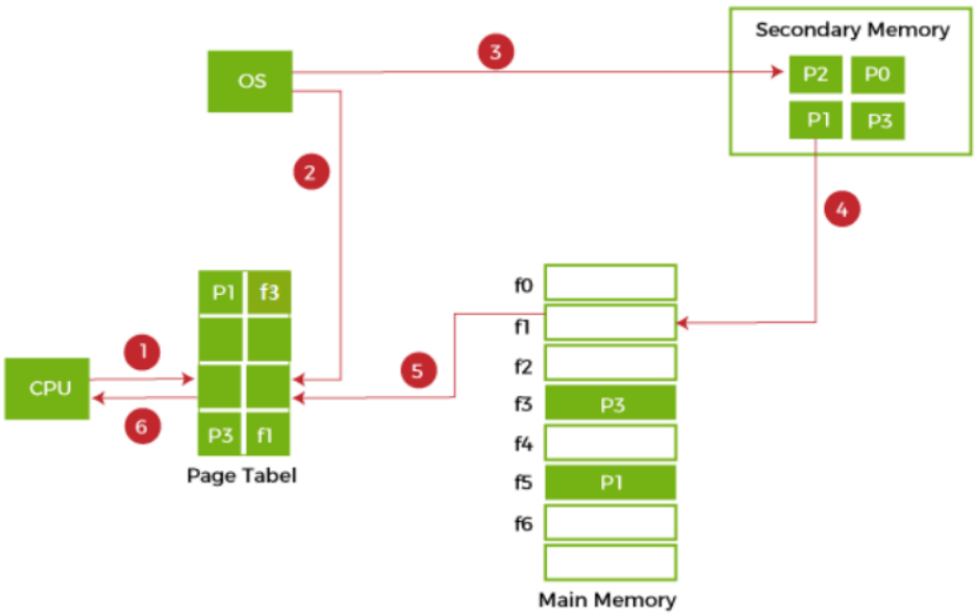
Demand paging is a process in which data is moved from secondary memory to RAM on a demand basis, which means all data is not stored in the main memory because the space is limited in RAM.

So if the CPU demands the process, if that page is not in RAM, then swapping is needed.

This means shifting the existing page from RAM and putting it back in secondary memory and putting the new page in RAM.

Demand paging is an essential feature of modern operating systems, and it helps ensure that all processes have access to the memory they need.

- E.g. - Suppose we have to execute a process P having four pages P_0, P_1, P_2 and P_3. Currently, in the page table, we have pages P_1 and P_3.



1. If the CPU wants to access page P_2 of a process P, it will first search the page in the page table.
2. As the page P2 is not found in the page table so it will be a page fault.
3. The control goes to the OS and context switching takes place, and the OS puts the process in a waiting state.

4. OS loads page P2 from secondary memory to main memory and will update the page table.
5. The control is taken back from the OS, and the execution of the process is resumed.

Data Structures in Demand paging:

1. Page table entries.
2. Disk block descriptors.
3. Page frame data table (pfdata).
4. Swap-use table.

Advantages:

1. Only loads pages that are demanded by the executing process.
2. As there is more space in main memory, more processes can be loaded.

Disadvantages:

1. More complex.
2. Less power.

▼ Q.3 Locality of reference?

Locality of reference refers to the tendency of the computer program to access the same set of memory locations for a particular time period.

The property of Locality of Reference is mainly shown by loops and subroutine calls in a program.

On an abstract level there are two types of localities which are as follows:

1. **Temporal locality**: Temporal locality refers to the reuse of specific data and/or resources within a relatively small time duration.
2. **Spatial locality**: Spatial locality (also termed data locality) refers to the use of data elements within relatively close storage locations.

▼ Q.4 Page Stealer Process?

This is the Kernel process that makes room for the incoming pages, by swapping the memory pages that are not the part of the working set of a process.

Page-Stealer is created by the Kernel at the system initialization and invokes it throughout the lifetime of the system.

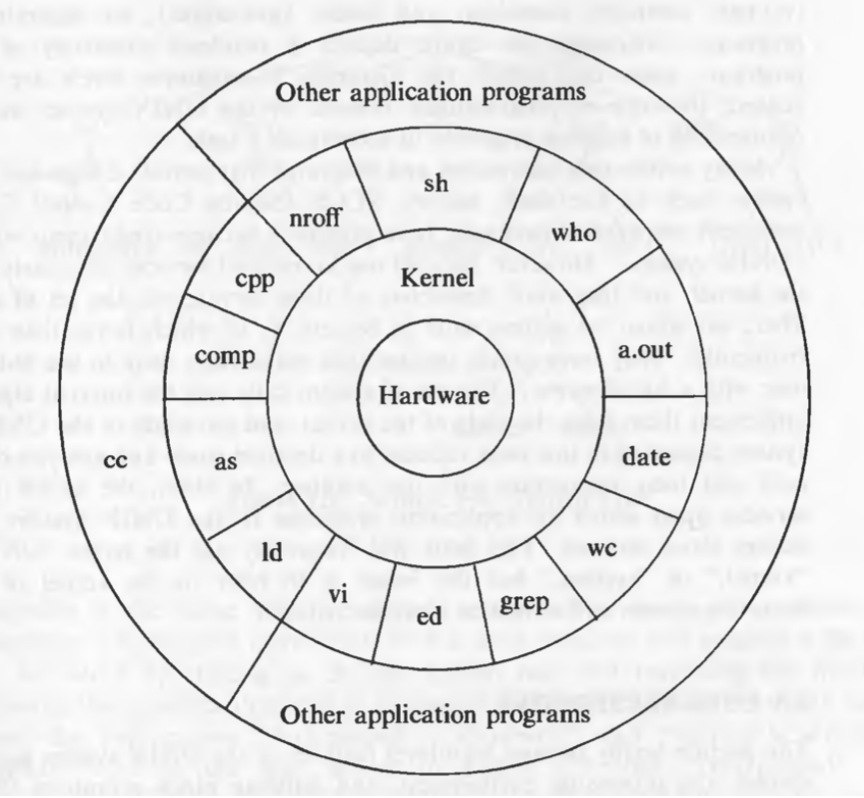
▼ Q.5 Page Fault? Two types of page fault.

A page fault occurs when a program attempts to access data or code that is in its address space, but is not currently located in the system RAM.

Types of page fault:

1. **Minor Page Fault**: A small page fault or soft page fault happens when a page is loaded into memory when a fault occurs.
2. **Major Page Fault**: A major page fault is an exception that occurs when a process attempts to access memory in a way that exceeds its permissions.
3. **Invalid Page Fault**: An invalid page fault happens when a page fault references an address that does not belong to the virtual address space.

▼ Q.6 What is an OS? Why do we need it?



The Unix operating system is a set of programs that act as a link between the computer and the user. It is a stable, multi-user, multi-tasking system for servers, desktops and laptops .

UNIX is made up of 4 main parts:

1. The Kernel:

a. The kernel is the heart of the operating system.
b. It interacts with the hardware to perform various tasks like memory allocation and file storage.

2. The Shell:
   a. The shell is the utility that processes your requests.
   b. When you type in a command at your terminal, the shell interprets the command and calls the program that you want.
3. Commands and Utilities:
   a. There are various commands and utilities which you can make use of.
   b. cp, mv, cat and grep, etc. are few examples of commands and utilities.
4. Files and Directories:
   a. All the data of Unix is organized into files.
   b. All files are then organized into directories.
   c. These directories are further organized into a tree-like structure called the filesystem.

▼ Q.7 What are data region, stack region, and address space?

Data Region: In computing, a data segment (often denoted . data) is a portion of an object file or the corresponding address space of a program that contains initialized static variables, that is, global variables and static local variables.

Stack Region: A stack is a special area of a computer's memory which stores temporary variables created by a function. In stack, variables are declared, stored and initialized during runtime. It is a temporary storage memory.

Address Space: The range of virtual addresses that the operating system assigns to a user or separately running program is called an address space. This is the area of contiguous virtual addresses available for executing instructions and storing data.

▼ Q.8 What is address space?

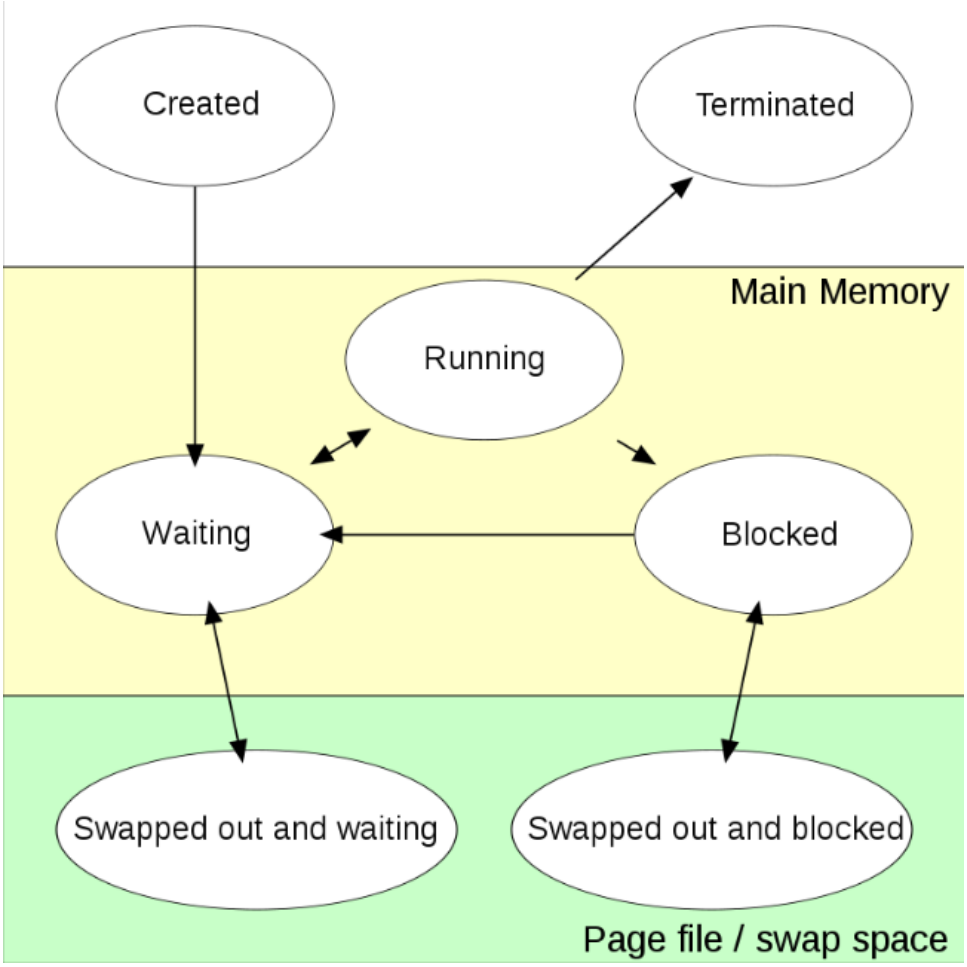
Same as Q.7

▼ Q.9 What is the difference between program and process?

Program	Process

Program	Process
A program is a collection of instructions that are used to complete a specific task.	A process is a program in execution.
A program is a passive entity	A process is an active entity.
It is considered as a static entity.	It is considered as a passive entity.
It doesn't have a control block.	It have its own control block called, Process Control Block
It is a dormant entity which just resides in the secondary memory.	It is an active entity created during execution which resides in the main memory.

▼ Q.10 Process state transition diagram.



Created: A program which is going to be picked up by the OS into the main memory is called a new process.

Ready: Whenever a process is created, it directly enters in the ready state, in which it waits for the CPU to be assigned.

Running: One of the processes from the ready state will be chosen by the OS depending upon the scheduling algorithm

Block/Wait: A process already using a resource waits to acquire other resources held by another process

Swapped out and waiting: In systems that support virtual memory, a process may be swapped out, that is, removed from main memory and placed on external storage by the scheduler. From here the process may be swapped back into the waiting state.

Swapped out and blocked: Processes that are blocked may also be swapped out. In this event the process is both swapped out and blocked, and may be swapped back in again.

▼ Q.11 What is scheduling?

Multiprogramming operating systems rely heavily on process scheduling. It is the process of removing an active task from the processor and replacing it with a new one. It divides a procedure into states such as ready, waiting, or running.

Process scheduling is an essential part of a Multiprogramming operating systems. Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

There are two categories of scheduling:

1. **Non-preemptive:** Here the resource can’t be taken from a process until the process completes execution. The switching of resources occurs when the running process terminates and moves to a waiting state.
2. **Preemptive:** Here the OS allocates the resources to a process for a fixed amount of time. During resource allocation, the process switches from running state to ready state or from waiting state to ready state. This switching occurs as the CPU may give priority to other processes and replace the process with higher priority with the running process.

▼ Q.12 What is the purpose of a fork with syntax?

The only way to create a new process in UNIX is to use the fork system call. The process
which calls fork is called the parent process and the newly created process is called the

child process. Also, the process 0 is the only process which is not created via fork.

The child process is exactly the same as its parent but there is difference in the processes ID's:

1. The process ID of the child process is a unique process ID which is different from the ID's of all other existing processes.
2. The Parent process ID will be the same as that of the process ID of child's parent.

Syntax: $pid = fork()$;

▼ Q.13 What is page fault?

It mainly occurs when any program tries to access the data or the code that is in the address space of the program, but that data is not currently located in the RAM of the system.

- So basically when the page referenced by the CPU is not found in the main memory then the situation is termed as Page Fault.
- Whenever any page fault occurs, then the required page has to be fetched from the secondary memory into the main memory.

The page fault mainly generates an exception, which is used to notify the operating system that it must have to retrieve the "pages" from the virtual memory in order to continue the execution. Once all the data is moved into the physical memory the program continues its execution normally.

▼ Q.14 What is virtual memory?

Virtual Memory is a storage mechanism which offers user an illusion of having a very big main memory. It is done by treating a part of secondary memory as the main memory. In Virtual memory, the user can store processes with a bigger size than the available main memory.

Therefore, instead of loading one long process in the main memory, the OS loads the various parts of more than one process in the main memory. Demand Paging is a popular method of virtual memory management. In demand paging, the pages of a process which are least used, get stored in the secondary memory.

▼ Q.15 What is the file system of UNIX?

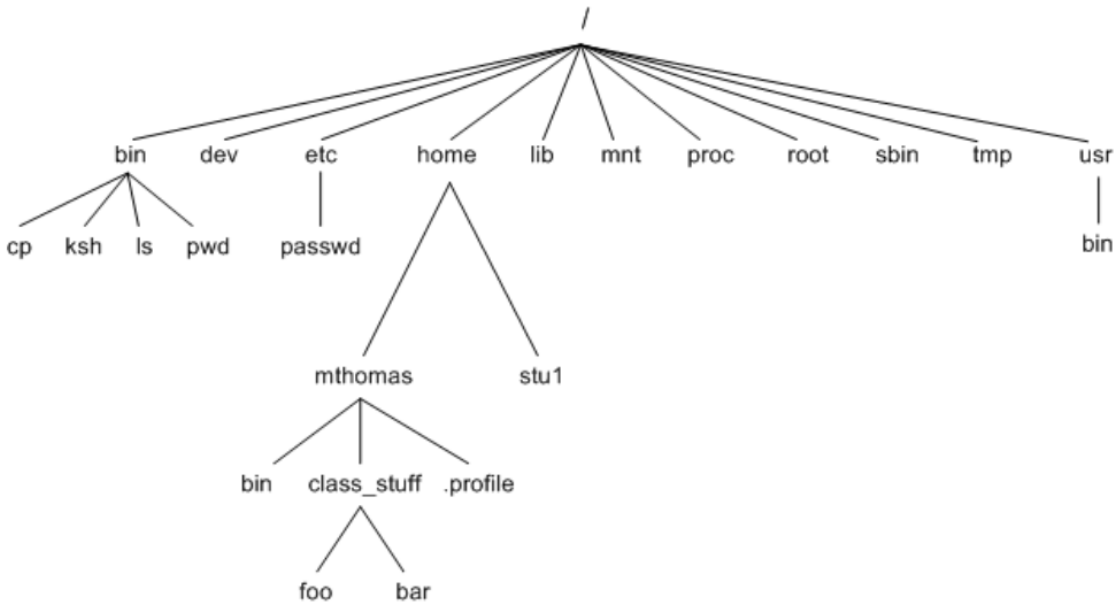
The Unix file system is a methodology for logically organizing and storing large quantities of data such that the system is easy to manage. A **file** can be informally defined as a collection of data, which can be logically viewed as a stream of bytes (i.e. characters). A file is the smallest unit of storage in the Unix file system.

A **file system** consists of files, relationships to other files, as well as the attributes of each file. File attributes are information relating to the file, but do not include the data contained within a file. File attributes for a generic operating system might include:

- a file type
- a file name
- a physical file size
- a file owner
- file protection
- file time stamp

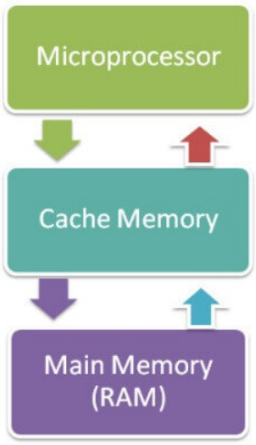
File systems provide tools which allow the manipulation of files, provide a logical organization as well as provide services which map the logical organization of files to physical devices. the Unix file system is essentially composed of files and **directories**. Directories are special files that may contain other files.

The Unix file system has a hierarchical (or tree-like) structure with its highest level directory called root (denoted by /). Immediately below the root level directory are several subdirectories, most of which contain system files. Below this can exist system files, application files, and/or user data files. Similar to the concept of the process parent-child relationship, all files on a Unix system are related to one another. That is, files also have a parent-child existence. Thus, all files (except one) share a common parental link, the top-most file (i.e. /) being the exception.



▼ Q.16 Why do we need cache data structure?

Cache memory is a high-speed memory, which is small in size but faster than the main memory (RAM). The CPU can access it more quickly than the primary memory. So, it is used to synchronize with high-speed CPU and to improve its performance.



Cache memory can only be accessed by CPU. It can be a reserved part of the main memory or a storage device outside the CPU. It holds the data and programs which are frequently used by the CPU. So, it makes sure that the data is instantly available for CPU whenever the CPU needs this data. In other words, if the CPU finds the required data or instructions in the cache memory, it doesn't need to access the primary memory (RAM). Thus, by acting as a buffer between RAM and CPU, it speeds up the system performance.

▼ Q.17 What are interrupts and exceptions?

Exceptions and interrupts are unexpected events which will disrupt the normal flow of execution of instruction.

Exceptions occur during program execution and are so extraordinary that they cannot be handled by the program itself. If you give the processor the command to divide a number by zero, for instance, it will give a divide-by-zero exception, which will cause the computer to either stop the operation or display an error notice.

Interruptions are sudden occurrences that bring to a halt the consistent flow of instructions being carried out by a system. It instructs the operating system to take immediate action on the following steps to take. These unforeseen occurrences are typically associated with an I/O device, which is primarily concerned with communicating with the outside world.

Whenever an exception or interrupt occurs, the hardware starts executing the code that performs an action in response to the exception. The instructions responsible for this action reside in the operating system kernel, and the code that performs this action is called the interrupt handler code.

▼ Q.18 Context of a process.

Each time a process is removed from access to the processor, sufficient information on its current operating state must be stored such that when it is again scheduled to run on the processor it can resume its operation from an identical position. This operational state data is known as its **context** and the act of removing the process's thread of execution from the processor (and replacing it with another) is known as a *process switch* or *context switch.*

The context of a process includes its address space, stack space, virtual address space, register set image (e.g. Program Counter (PC), Stack Pointer (SP), Instruction Register (IR), Program Status Word (PSW) and other general processor registers).

This state information is saved in the process's process control block which is then moved to the appropriate scheduling queue. The new process is moved to the CPU by copying the PCB info into the appropriate locations.

▼ Q.19 What is an OS? Why do we need it? Functioning of an OS.

An Operating System acts as a communication bridge (interface) between the user and computer hardware. The purpose of an operating system is to provide

a platform on which a user can execute programs in a convenient and efficient manner.

An operating system is a piece of software that manages the allocation of computer hardware. The coordination of the hardware must be appropriate to ensure the correct working of the computer system and to prevent user programs from interfering with the proper working of the system.

The operating system is commonly called the system kernel, or just the kernel, emphasizing its isolation from user programs.

Important functions of an operating System:

1. Security
2. Control over system performance
3. Job accounting
4. Error detecting aids
5. Coordination between other software and users
6. Memory Management
7. Processor Management
8. Device Management
9. File Management

Operating System Services:

1. Kernel controls creation, termination or suspension, and communication of the
   process.
2. Process scheduling.
3. Allocating main memory for an executing process.
4. Allocating secondary memory for efficient storage an retrieval of user data.
5. Allowing processes controlled access to peripheral devices such as terminals,
   tape drives, disk drives, and network devices.

▼ Q.20 Why is UNIX different from other OS?

There are various main differences between the UNIX and other. Some of them are as follows:

1. UNIX operating system comes with a Command Line Interface (CLI). In contrast, other operating systems come with a Graphical User Interface (GUI).
2. Multiprocessing is possible in the UNIX OS.
3. UNIX is a free and open-source OS. In contrast to other OS.
4. Unix operating system is known for being very stable to execute.
5. Unix is a flexible operating system that may be installed on various systems, including mainframes, supercomputers, and microcomputers.

▼ Q.21 Compare UNIX with other OS?

Parameters	**UNIX**	Other OS
Basic	It is a command-based operating system.	Menu based operating system.
Licensing	It is an open-source system which can be used to under General Public License.	Maybe a proprietary software.
User Interface	It has a text base interface, making it harder to grasp for newcomers.	Has a Graphical User Interface, making it simpler to use.
Processing	It supports Multiprocessing.	It supports Multithreading.
File System	It uses Unix File System(UFS) that comprises STD.ERR and STD.IO file systems.	Uses File Allocation System (FAT32) and New technology file system(NTFS).
Security	It is more secure as all changes to the system require explicit user permission.	Less secure compared to UNIX.
Hardware	Hardware support is limited in UNIX system. Some hardware might not have drivers built for them.	Drivers are available for almost all the hardware.
Reliability	Unix and its distributions are well known for being very stable to run.	Although Windows has been stable in recent years, it is still to match the stability provided by Unix systems.

▼ Q.22 Features of UNIX.

1. **Multiuser System**: Since all the resources are shared among all the users, it is called a Multiuser system. In this, one user can do multiple tasks. For doing this, a small slice of time is given by a computer to a user to divide the unit into several segments. At a particular instant of time, the CPU is addressing only one user but since the switching time of the switch is so quick, it creates an illusion that multiple users are addressed simultaneously. This is also called context switching where the state of the switch is changed from one state to another.
2. **Portability**
3. **Hierarchical File System**
4. **UNIX Toolkit**: UNIX provides various facilities regarding types of tools like awk, UNIX grep, sed, etc. The general-purpose tool used is a network application, compiler, interpreter, etc. Various server programs are also included for providing remote administrative services.
5. **Program Execution**: To execute UNIX programs, the only thing that a user needs to do is type its name, and preceding name, type ./ for checking if the file is executable or not.
6. **Multitask system**: A user might run multiple tasks concurrently, for instance, editing a file, browsing the net, printing the file, etc.
7. **Pattern Matching**: UNIX provides a complex feature for matching patterns in file names. A special character, meta-char ‘’ is used by the system for matching the file name.
8. **Programming Facility**: The UNIX system has a programming language, the shell that is specially designed for programmers instead of normal users. It has features like loops, variables, and control structures required for programming. All these features are designed for shell scripts for invoking commands in UNIX.
9. **Documentation**: It has a command for the manual which is ‘man’, which is used as a reference command and for configuring files. There’s a vast amount of information available on the Internet apart from online documentation or GitHub.
10. **Machine-independence**
11. **Handling I/O operations**

▼ Q.23 Explain pipes in OS.

1. Pipes allow transfer of data between processes in a First in First out manner.
2. A pipe is just a file system from which the kernel can assign inodes and data blocks for pipes.
3. Pipe is used to combine two or more commands, and in this, the output of one command acts as input to another command, and this command's output may act as input to the next command and so on.
4. It can also be visualized as a temporary connection between two or more *commands/ programs/ processes*.
5. You can make it do so by using the pipe character '|'.
6. Syntax: **command_1 | command_2 | command_3 | .... | command_N**
7. There are 2 types of pipes: Named & Unnamed.
8. Named pipes are created using open().
9. Unnamed pies are created using pipe system call.

▼ Q.24 Write short note on *inode* and what are its characteristics?

1. In Unix based operating system each file is indexed by an Inode. Inode are special disk blocks they are created when the file system is created.
2. The number of Inode limits the total number of files/directories that can be stored in the file system.
3. The Inode contains the following information:
    a. Numeric UID of the owner.
    b. Numeric GUID of the owner.
    c. Size of the file.
    d. File type: regular, directory, device etc.
    e. Date and Time of Last modification of the file data.
    f. Date and Time of Last access of file data.
    g. Date and Time of Last change of the I-node.
4. When a file is created on a system, a file name and inode number is assigned to it.
5. Generally, to access a file, a user uses the file name but internally file name is first mapped with respective inode number stored in a table.

6. inode doesn't contain the file name. Reason for this is to maintain hardlinks for the files.

▼ Q.25 Define Process in detail and what are different data structures associated with it
and what are their purpose?

A process is a program in execution. It is an active and dynamic activity. It have its own control block called Process Control Block and, it requires additional resources for execution. A process has limited lifespan.

Data structure used in process:

1. Process Control Block (PCB):
    a. A process control block (PCB) is a data structure used by computer operating systems to store all the information about a process.
    b. It is also known as a process descriptor.
    c. When a process is created (initialized or installed), the operating system creates a corresponding process control block.
2. Process table Entry:
    a. The process table contains fields that must always be accessible to the kernel.
    b. The kernel contains a process table with an entry that describes the state of every active process in the system.
    c. Kernel has a process table that keeps track of all active processes.
    d. Each entry in the process table contains pointers to the text, data, stack and the U Area of a process.
3. U Area:
    a. The uarea contains fields that need to be accessible only to the running process.
    b. Therefore, the kernel allocates space for the uarea only when creating a process.
    c. It does not need uareas for process table entries that do not have processes.
    d. The uarea contains additional information that controls the operation of a process.

▼ Q.26 Write difference between long-term, short-term and medium-term scheduler.

Long-term Schedular	Short-term Schedular	Medium-term Schedular
It is a job schedular.	It is a CPU schedular.	It is a process swapping schedular.
It takes process from the job pool.	It takes process from the ready state.	It takes process from running or wait/dead state.
Its speed is lesser than short-term schedular.	It is fastest among the two other schedulers.	Its speed is in between long-term and short-term.
It controls the degree of multiprogramming.	It has less control over the degree of multiprogramming.	It reduces the degree of multiprogramming.

▼ Q.27 Short note on Operating System services.

The operating system gives several services to utility programmers and users. Following are the services provided by an operating system:

1. **Program execution:** To execute a program, several tasks need to be performed. Both the instructions and data must be loaded into the main memory. In addition, input-output devices and files should be initialized, and other resources must be prepared.
2. **Control I/O devices:** As there are numerous types of I/O devices within the computer system, and each I/O device calls for its own precise set of instructions for the operation. The Operating System hides that info with the aid of presenting a uniform interface.
3. **Program creation:** The Operating system offers the structures and tools, including editors and debuggers, to help the programmer create, modify, and debugging programs.
4. **Error Detection and Response:** An Error in a device may also cause malfunctioning of the entire device. To avoid error, the operating system monitors the system for detecting errors and takes suitable action with at least impact on running applications.
5. **Accounting:** An Operating device collects utilization records for numerous assets and tracks the overall performance parameters and responsive time to enhance overall performance.

6. **Security and Protection:** Operating device affords safety to the statistics and packages of a person and protects any interference from unauthorized users.
7. **File management:** Computers keep data and information on secondary storage devices like magnetic tape, magnetic disk, optical disk, etc.
8. **Communication:** The operating system manages the exchange of data and programs among different computers connected over a network. This communication is accomplished using message passing and shared memory.

▼ Q.28 Write formal steps for context switching of a process?

The steps involved in context switching are as follows:

1. Save the context of the process that is currently running on the CPU. Update the process control block and other important fields.
2. Move the process control block of the above process into the relevant queue such as the ready queue, I/O queue etc.
3. Select a new process for execution.
4. Update the process control block of the selected process. This includes updating the process state to running.
5. Update the memory management data structures as required.
6. Restore the context of the process that was previously running when it is loaded again on the processor. This is done by loading the previous values of the process control block and registers.

▼ Q.29 Write a short note on ialloc and *ifree* algorithm?

Algorithm *ialloc* is used to assign an inode to a newly created file:

```
/*  Algorithm: ialloc
 *  Input: file system
 *  Output: locked inode
 */

{
  while (not done)
  {
    if (super block locked)
    {
      sleep (event: super block becomes free);
      continue;
    }
```

```
    if (indoe list in super block is empty)
    {
      lock super block;
      get remembered inode for free inode search;
      search disk for free inodes until super block full, or no more free inodes
 (algorithm: bread and brelse);
      unlock super block;
      wake up (event: super block becomes free);
      if (no free inodes found on disk)
        return (no inode);
      set remembered inode for next free inode search;
    }
    // there are inodes in super block inode list
    get inode number form super block inode list;
    get inode (algorithm: iget);
    if (inode not free after all)
    {
      write inode to disk;
      release inode (algorithm: iput);
      continue;
    }
    // inode is free
    initialize inode;
    write inode to disk;
    decrement file system free inode count;
    return inode;
  }
}
```

Algorithm for freeing inodes (*ifree*):

```
/*  Algorithm: ifree
 *  Input: file system inode number
 *  Output: none
 */

{
  increment file system free inode count;
  if (super block locked)
    return;
  if (inode list full)
  {
    if (inode number less than remembered inode for search)
      set remembered inode for search = input inode number;
  }
  else
    store inode number in inode list;
  return;
}
```