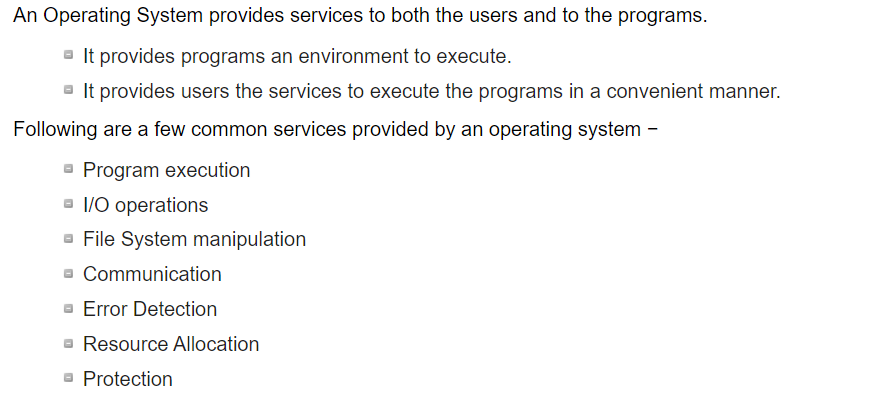
**UNIT** 1:

**1)WHAT ARE OF SERVICE OF OPERATING SYSTEM?**

**A)**



**Program execution:**

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

**I/O Operation:**

An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

**File system manipulation:**

A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

**Communication:**

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

**Error handling:**

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling

**Resource Management:**

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management

**Protection:**

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities..

**2) DEFINE SYSTEM CALLS? EXPLAIN The TYPES OF SYSTEM CALLS?**

**A)**

The interface between a process and an operating system is provided by system calls. In general, system calls are available as assembly language instructions.

System calls are usually made when a process in user mode requires access to a resource. Then it requests the kernel to provide the resource via a system call.

**Here are the types of system calls** −

### **Process Control:**

These system calls deal with processes such as process creation, process termination etc.

### **File Management:**

These system calls are responsible for file manipulation such as creating a file, reading a file, writing into a file etc.

### **Device Management:**

These system calls are responsible for device manipulation such as reading from device buffers, writing into device buffers etc.

### **Information Maintenance:**

These system calls handle information and its transfer between the operating system and the user program.

### **Communication:**

These system calls are useful for interprocess communication. They also deal with creating and deleting a communication connection.

**3)EXPLAIN THE STRUCTURE OF AN OPERATING SYSTEM?**

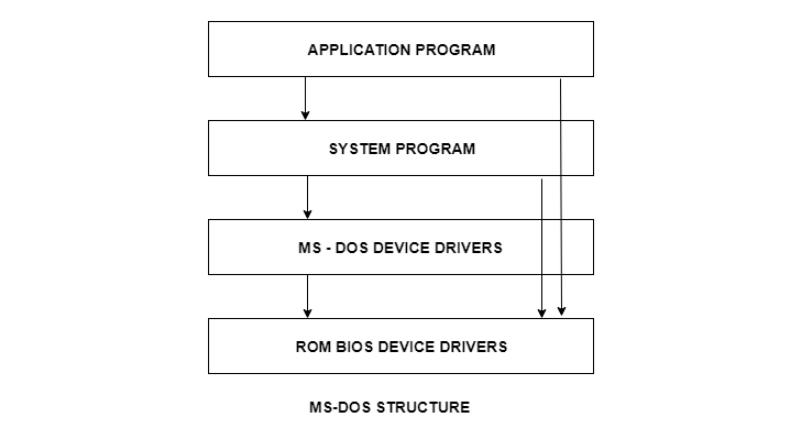
**A)**

An operating system is a construct that allows the user application programs to interact with the system hardware.

## Simple Structure

There are many operating systems that have a rather simple structure. These started as small systems and rapidly expanded much further than their scope. A common example of this is MS-DOS. It was designed simply for a niche amount for people. There was no indication that it would become so popular.

An image to illustrate the structure of MS-DOS is as follows −



It is better that operating systems have a modular structure, unlike MS-DOS. That would lead to greater control over the computer system and its various applications. The modular structure would also allow the programmers to hide information as required and implement internal routines as they see fit without changing the outer specifications.

**UNIT 2:**

**1)DEFINE PROCESS? EXPLAIN The STATES OF PROCESS?**

A)

## Process:

A process is basically a program in execution. The execution of a process must progress in a sequential fashion.

The names of the states are not standardized although the process may be in one of the following states during execution.

### **1. New**

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

### **2. Ready**

Whenever a process is created, it directly enters in the ready state, in which, it waits for the CPU to be assigned. The OS picks the new processes from the secondary memory and put all of them in the main memory.

The processes which are ready for the execution and reside in the main memory are called ready state processes. There can be many processes present in the ready state.

### **3. Running**

One of the processes from the ready state will be chosen by the OS depending upon the scheduling algorithm. Hence, if we have only one CPU in our system, the number of running processes for a particular time will always be one. If we have n processors in the system then we can have n processes running simultaneously.

### **4. Block or wait**

From the Running state, a process can make the transition to the block or wait state depending upon the scheduling algorithm or the intrinsic behavior of the process.

When a process waits for a certain resource to be assigned or for the input from the user then the OS move this process to the block or wait state and assigns the CPU to the other processes.

### **5. Completion or termination**

When a process finishes its execution, it comes in the termination state. All the context of the process (Process Control Block) will also be deleted the process will be terminated by the Operating system.

### **6. Suspend ready**

A process in the ready state, which is moved to secondary memory from the main memory due to lack of the resources (mainly primary memory) is called in the suspend ready state.

If the main memory is full and a higher priority process comes for the execution then the OS have to make the room for the process in the main memory by throwing the lower priority process out into the secondary memory. The suspend ready processes remain in the secondary memory until the main memory gets available.

### **7. Suspend wait**

Instead of removing the process from the ready queue, it's better to remove the blocked process which is waiting for some resources in the main memory. Since it is already waiting for some resource to get available hence it is better if it waits in the secondary memory and make room for the higher priority process. These processes complete their execution once the main memory gets available and their wait is finished.

## Operations on the Process

### **1. Creation**

Once the process is created, it will be ready and come into the ready queue (main memory) and will be ready for the execution.

### **2. Scheduling**

Out of the many processes present in the ready queue, the Operating system chooses one process and start executing it. Selecting the process which is to be executed next, is known as scheduling.

### **3. Execution**

Once the process is scheduled for the execution, the processor starts executing it. Process may come to the blocked or wait state during the execution then in that case the processor starts executing the other processes.

### **4. Deletion/killing**

Once the purpose of the process gets over then the OS will kill the process. The Context of the process (PCB) will be deleted and the process gets terminated by the Operating system.

**2)WHAT IS A SCHEDULER? EXPLAIN VARIOUS TYPES OF SCHEDULERS AND THEIR ROLES?**

**A)**

Process Scheduling handles the selection of a process for the processor on the basis of a scheduling algorithm and also the removal of a process from the processor. It is an important part of multiprogramming operating system.

There are many scheduling queues that are used in process scheduling. When the processes enter the system, they are put into the job queue. The processes that are ready to execute in the main memory are kept in the ready queue. The processes that are waiting for the I/O device are kept in the I/O device queue.

The different schedulers that are used for process scheduling are −

## Long Term Scheduler

The job scheduler or long-term scheduler selects processes from the storage pool in the secondary memory and loads them into the ready queue in the main memory for execution.

The long-term scheduler controls the degree of multiprogramming. It must select a careful mixture of I/O bound and CPU bound processes to yield optimum system throughput. If it selects too many CPU bound processes then the I/O devices are idle and if it selects too many I/O bound processes then the processor has nothing to do.

The job of the long-term scheduler is very important and directly affects the system for a long time.

## Short Term Scheduler

The short-term scheduler selects one of the processes from the ready queue and schedules them for execution. A scheduling algorithm is used to decide which process will be scheduled for execution next.

The short-term scheduler executes much more frequently than the long-term scheduler as a process may execute only for a few milliseconds.

The choices of the short term scheduler are very important. If it selects a process with a long burst time, then all the processes after that will have to wait for a long time in the ready queue. This is known as starvation and it may happen if a wrong decision is made by the short-term scheduler.

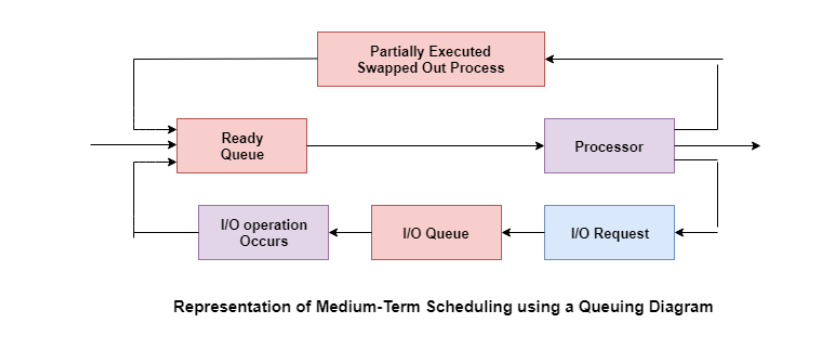
A diagram that demonstrates long-term and short-term schedulers is given as follows −

## Medium Term Scheduler

The medium-term scheduler swaps out a process from main memory. It can again swap in the process later from the point it stopped executing. This can also be called as suspending and resuming the process.

This is helpful in reducing the degree of multiprogramming. Swapping is also useful to improve the mix of I/O bound and CPU bound processes in the memory.

A diagram that demonstrates medium-term scheduling is given as follows −



**3) EXPLAIN THE SIGNIFICANCE OF PROCESS CONTROL BLOCK AND DESCRIBE ITS TYPICAL ELEMENTS.**

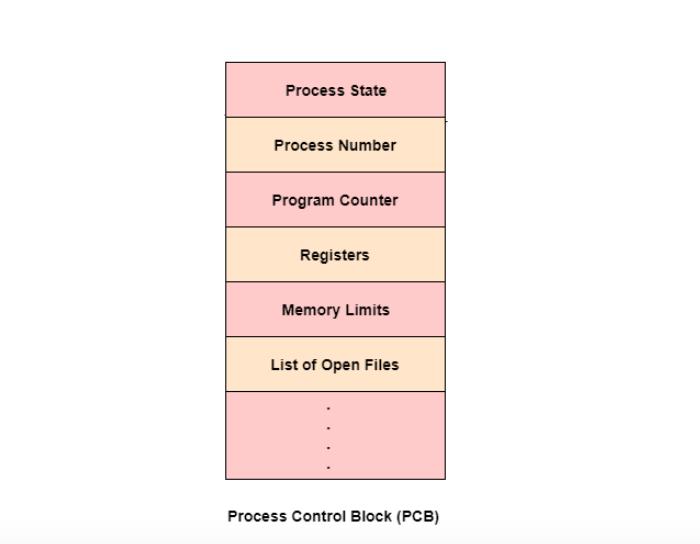
**A)**

Process Control Block is a data structure that contains information of the process related to it. The process control block is also known as a task control block, entry of the process table, etc.

It is very important for process management as the data structuring for processes is done in terms of the PCB. It also defines the current state of the operating system.

## Structure of the Process Control Block

The process control stores many data items that are needed for efficient process management. Some of these data items are explained with the help of the given diagram −



The following are the data items −

### Process State

This specifies the process state i.e. new, ready, running, waiting or terminated.

### Process Number

This shows the number of the particular process.

### Program Counter

This contains the address of the next instruction that needs to be executed in the process.

### Registers

This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.

### List of Open Files

These are the different files that are associated with the process

### CPU Scheduling Information

The process priority, pointers to scheduling queues etc. is the CPU scheduling information that is contained in the PCB. This may also include any other scheduling parameters.

### Memory Management Information

The memory management information includes the page tables or the segment tables depending on the memory system used. It also contains the value of the base registers, limit registers etc.

### I/O Status Information

This information includes the list of I/O devices used by the process, the list of files etc.

### Accounting information

The time limits, account numbers, amount of CPU used, process numbers etc. are all a part of the PCB accounting information.

### Location of the Process Control Block

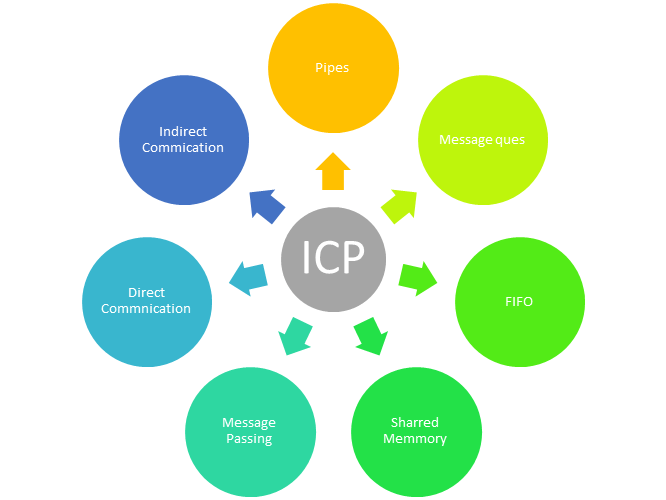
The process control block is kept in a memory area that is protected from the normal user access. This is done because it contains important process information. Some of the operating systems place the PCB at the beginning of the kernel stack for the process as it is a safe location.

**4)**  **WHAT IS THE NEED OF INTER PROCESS COMMUNICATION?**

**A)**

**Inter process communication (IPC)** is used for exchanging data between multiple threads in one or more processes or programs. The Processes may be running on single or multiple computers connected by a network. The full form of IPC is Inter-process communication.

Here, are few important methods for interprocess communication:



Inter-Process Communication Approaches

### Pipes

Pipe is widely used for communication between two related processes. This is a half-duplex method, so the first process communicates with the second process. However, in order to achieve a full-duplex, another pipe is needed.

### Message Passing:

It is a mechanism for a process to communicate and synchronize. Using message passing, the process communicates with each other without resorting to shared variables.

IPC mechanism provides two operations:

* Send (message)- message size fixed or variable
* Received (message)

### Message Queues:

A message queue is a linked list of messages stored within the kernel. It is identified by a message queue identifier. This method offers communication between single or multiple processes with full-duplex capacity.

### Direct Communication:

In this type of inter-process communication process, should name each other explicitly. In this method, a link is established between one pair of communicating processes, and between each pair, only one link exists.

### Indirect Communication:

Indirect communication establishes like only when processes share a common mailbox each pair of processes sharing several communication links. A link can communicate with many processes. The link may be bi-directional or unidirectional.

### Shared Memory:

Shared memory is a memory shared between two or more processes that are established using shared memory between all the processes. This type of memory requires to protected from each other by synchronizing access across all the processes.

### FIFO:

Communication between two unrelated processes. It is a full-duplex method, which means that the first process can communicate with the second process, and the opposite can also happen.

**5)** **DEFINE THREAD? EXPLAIN MULTITHREADING MODELS?**

**A)**

Multithreading allows the execution of multiple parts of a program at the same time. These parts are known as threads and are lightweight processes available within the process. Therefore, multithreading leads to maximum utilization of the CPU by multitasking.

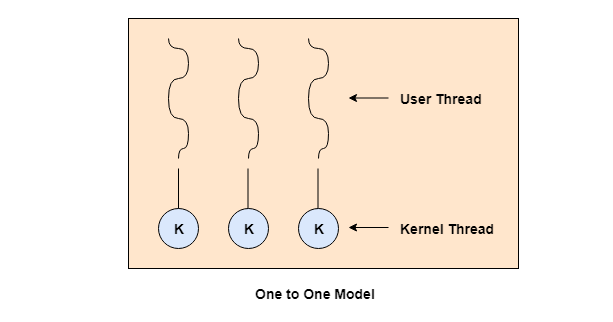
The main models for multithreading are one to one model, many to one model and many to many model. Details about these are given as follows −

## One to One Model

The one to one model maps each of the user threads to a kernel thread. This means that many threads can run in parallel on multiprocessors and other threads can run when one thread makes a blocking system call.

A disadvantage of the one to one model is that the creation of a user thread requires a corresponding kernel thread. Since a lot of kernel threads burden the system, there is restriction on the number of threads in the system.

A diagram that demonstrates the one to one model is given as follows −

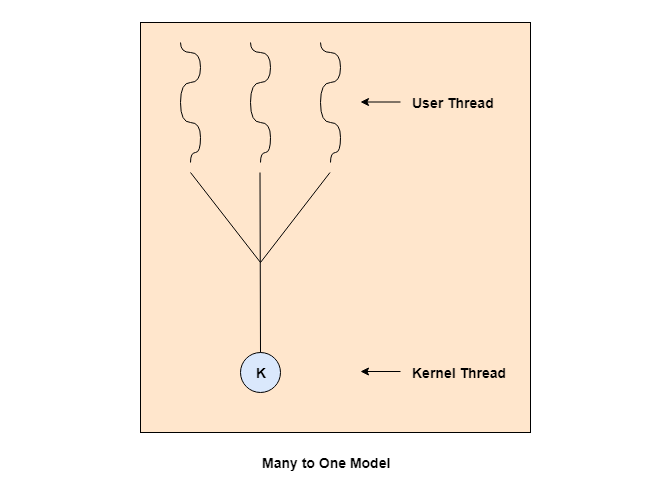


## Many to One Model

The many to one model maps many of the user threads to a single kernel thread. This model is quite efficient as the user space manages the thread management.

A disadvantage of the many to one model is that a thread blocking system call blocks the entire process. Also, multiple threads cannot run in parallel as only one thread can access the kernel at a time.

A diagram that demonstrates the many to one model is given as follows −

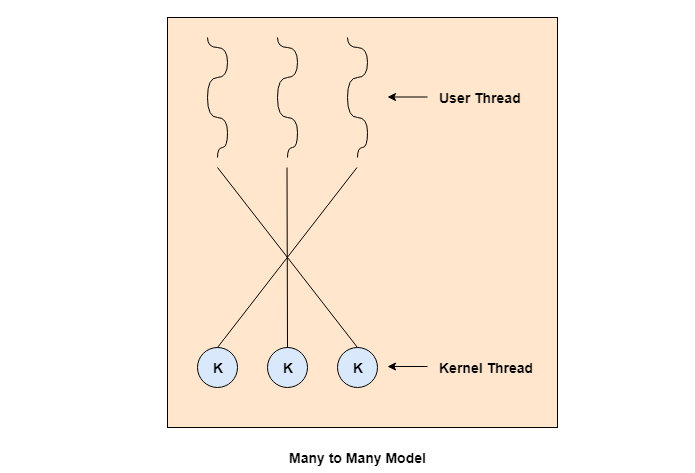


## Many to Many Model

The many to many model maps many of the user threads to a equal number or lesser kernel threads. The number of kernel threads depends on the application or machine.

The many to many does not have the disadvantages of the one to one model or the many to one model. There can be as many user threads as required and their corresponding kernel threads can run in parallel on a multiprocessor.

A diagram that demonstrates the many to many model is given as follows −



**UNIT 3:**

**1)What** **are the conditions for Resource Deadlock?**

**A)**

A deadlock situation can arise if and only if the following four conditions hold simultaneously in a system-

**Mutual Exclusion**: At least one resource is held in a non-sharable mode that is only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.

**Hold and Wait**:There must exist a process that is holding at least one resource and is waiting to acquire additional resources that are currently being held by other processes.

**No Preemption**: Resouces cannot be preempted; that is, a resource can only be released voluntarily by the process holding it, after the process has completed its task.

**Circular Wait**: There must exist a set {p0, p1,.....pn} of waiting processes such that p0 is waiting for a resource which is held by p1, p1 is waiting for a resource which is held by p2,..., pn-1 is waiting for a resource which is held by pn and pn is waiting for a resource which is held by p0.

**UNIT 4:**

1. **Explain the following file concepts: a) File attributes b) File operations c) File types d) Internal file structure**
2. A file is a collection of correlated information which is recorded on secondary or non-volatile storage like magnetic disks, optical disks, and tapes. It is a method of data collection that is used as a medium for giving input and receiving output from that program.

## File Attributes

A file has a name and data. Moreover, it also stores meta information like file creation date and time, current size, last modified date, etc. All this information is called the attributes of a file system.

Here, are some important File attributes used in OS:

* **Name:** It is the only information stored in a human-readable form.
* **Identifier**: Every file is identified by a unique tag number within a file system known as an identifier.
* **Location:** Points to file location on device.
* **Type:** This attribute is required for systems that support various types of files.
* **Size**. Attribute used to display the current file size.
* **Protection**. This attribute assigns and controls the access rights of reading, writing, and executing the file.
* **Time, date and security:** It is used for protection, security, and also used for monitoring

## File Type

It refers to the ability of the operating system to differentiate various types of files like text files, binary, and source files. However, Operating systems like MS\_DOS and UNIX has the following type of files:

### Character Special File

It is a hardware file that reads or writes data character by character, like mouse, printer, and more.

### Ordinary files

* These types of files stores user information.
* It may be text, executable programs, and databases.
* It allows the user to perform operations like add, delete, and modify.

### Directory Files

* Directory contains files and other related information about those files. Its basically a folder to hold and organize multiple files.

### Special Files

* These files are also called device files. It represents physical devices like printers, disks, networks, flash drive, etc.

**Functions of File**

* Create file, find space on disk, and make an entry in the directory.
* Write to file, requires positioning within the file
* Read from file involves positioning within the file
* Delete directory entry, regain disk space.
* Reposition: move read/write position.

**File structure**

A File Structure needs to be predefined format in such a way that an operating system understands. It has an exclusively defined structure, which is based on its type.

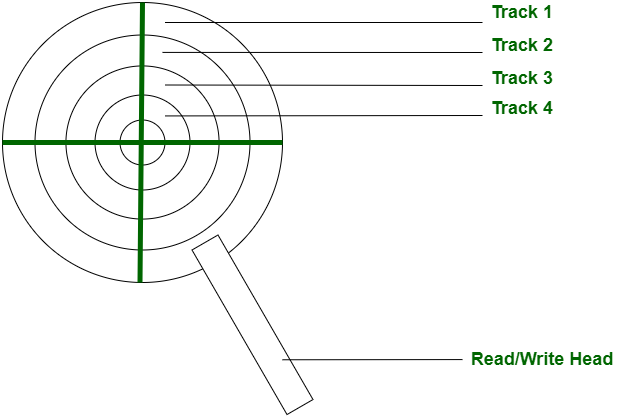
Three types of files structure in OS:

* A text file: It is a series of characters that is organized in lines.
* An object file: It is a series of bytes that is organized into blocks.
* A source file: It is a series of functions and processes.

1. **Explain the following in detail with respect to disk?**
2. **Seek time b) Latency c) Access time d) Transfer time**

**A) Seek Time:**  
A [disk](https://www.geeksforgeeks.org/hard-disk-drive-hdd-secondary-memory/) is divided into many circular tracks. Seek Time is defined as the time required by the read/write head to move from one track to another.

Example,  
Consider the following diagram, the read/write head is currently on track 1.



**Disk Access Time:**  
Disk Access Time is defined as the total time required by the computer to process a read/write request and then retrieve the required data from the disk storage.

Disk Access Time is divided into 2 parts:

1. Access Time
2. Data Transfer Time

Disk Access Time = Access Time + Data Transfer Time

**1. Access Time:**  
Access Time is defined as the setup time before the actual data transfer takes place.  
For example, the read/write head is on track 1 but we need to read data from another track or segment. Thus, the read/write head will move to the data block location before the actual transfer can take place. This delay is called Access Time.

Access Time is calculated by summation of the following:

**(a).** Seek Time

**(b).** Rotational Latency

**(c).** Command Processing Time

**(d).** Settle Time

These are explained as following below in brief.

* **(a). Seek Time –**  
  It is the time required by the read/write head to move from the current track to the requested track.
* Seek Time

= (Number of tracks/cylinders crossed) \* (Time to cross one track/cylinder)

* **(b). Rotational Latency –**  
  It is the time required by the read/write head to move from the current sector to the requested sector.
* Rotational Latency

= (Angle by which disk is rotated) / (Angular Frequency)

* **(c). Command Processing Time –**  
  It is the time required by the disk device to process the command and establish a connection between the various components of the disk device to read/write data. It is due to the internal circuitry.
* **(d). Settle Time –**  
  Settle Time is the time required by read/write head to stop vibrating.

**Note:** Command Processing Time and Settle Time are not normally mentioned in numerical question. We take them as zero.

**2. Data Transfer Time:**  
Data Transfer Time is defined as the time required to transfer data between the system and the disk.  
Data Transfer Time is of two types:

**(a).** Internal Transfer Rate

**(b).** External Transfer Rate

These are explained as following below in brief.

* **(a). Internal Transfer Rate –**  
  It is defined as the time required to move data between the disk surface and hard disk cache.
* **(b). External Transfer Rate –**  
  It is defined as the time required to move data between the hard disk cache and the system.

**Latency:**

Latency(or rotational delay)is the time it takes for the beginning of the required sector to reach the head

**3)Explain**

**a) Paging b) Page table structure**

**c) Translation look-aside buffer d) Segmentation**

**A)**

**4) Explain Contiguous Memory Allocation.**

**A)**

**5) Explain Demand Paging.**

**A)**

**UNIT 5:**

1. **What is RAID? What are different RAID levels? Explain them.**

**A)**

**2)** **Explain Mass Storage Disk Structure ?**

**A)**