COURSE CODE & NAME DCA1201 – OPERATING SYSTEM

SET – 1

1.)

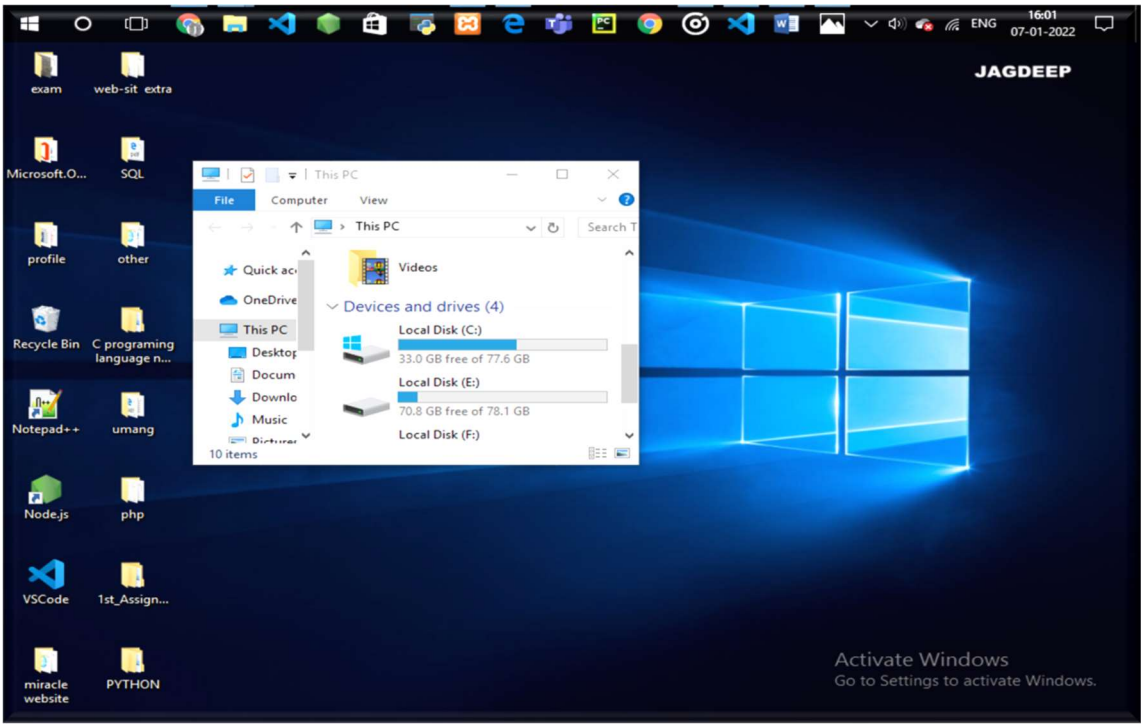
a.) Question :- Discuss the three Operating System Structures?

Answer :- An **operating system** is the **most important software** that runs on a computer. It manages the computer's **memory** and **processes**, as well as all of its **software** and **hardware**. It also allows you to **communicate** with the computer without knowing how to speak the computer's language. **Without an operating system, a computer is useless.**

Operating systems usually come **pre-loaded** on any computer you buy. Most people use the operating system that comes with their computer, but it's possible to upgrade or even change operating systems. The three most common operating systems for personal computers are **Microsoft Windows**, **Mac OS X**, and **Linux**.

⇨ Microsoft Windows

Modern operating systems use a **graphical user interface**, or **GUI** (pronounced **gooey**). A GUI lets you use your mouse to click **icons**, **buttons**, and **menus**, and everything is clearly displayed on the screen using a combination of **graphics** and **text**.



Each operating system's GUI has a different look and feel, so if you switch to a different operating system it may seem unfamiliar at first. However, modern operating systems are designed to be **easy to use**, and most of the basic principles are the same.

⇨ Linux

Linux (pronounced **LINN-ux**) is a family of **open-source** operating systems, which means they can be modified and distributed by anyone around the world. This is different from **proprietary software** like Windows, which can only be modified by the company that owns it. The advantages of Linux are that it is **free**, and there are many different **distributions**—or versions—you can choose from.

According to StatCounter Global Stats, Linux users account for less than **2%** of global operating systems. However, most **servers** run Linux because it's relatively easy to customize.

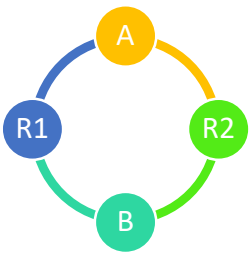
⇨ Mac OS X

The operating systems we've been talking about so far were designed to run on **desktop** and **laptop** computers. **Mobile devices** such as **phones**, **tablet computers**, and **MP3 players** are different from desktop and laptop computers, so they run operating systems that are designed specifically for mobile devices. Examples of mobile operating systems include **Apple iOS** and **Google Android**. In the screenshot below, you can see iOS running on an iPad.

Operating systems for mobile devices generally aren't as fully featured as those made for desktop and laptop computers, and they aren't able to run all of the same software. However, you can still do a lot of things with them, like watch movies, browse the Web, manage your calendar, and play games.

b.) Question :- What is Deadlock avoidance? Discuss Banker's algorithm for the same?

Answer:- There was a time where operating systems were only able to execute a single process at a time, thus giving full system resource and attention to that one single process. Nowadays operating systems can handle multiple tasks at once, but sometimes they have to deal with a problem known as a **deadlock**. A deadlock occurs when there is at least one process which is waiting for resources to be released by another process in order to finish a task correctly.



In this graph, Process A is waiting for Resource R2 to be released by Process B to finish a task. At the same time, Process B is waiting for Resource R1 to be released by Process A to finish a task. The technical term for what is happening here is known as **starvation**. Process B cannot begin until Process A finishes, which would then prevent the whole system from moving forward if deadlocked. This is something operating systems have to deal with, and many techniques can be implemented to stop or prevent a deadlock from occurring altogether.

Prevention & Avoidance

A deadlock can occur if and only if all the following conditions in a system are fulfilled simultaneously. These conditions are called the **Coffman conditions**:

- **Mutual exclusion** states that each resource can be assigned to only one process at a time
- **Circular wait** means that a process is holding a resource and requires more of the resources which are being held by other processes
- **Resource holding** is when one or more processes can hold and wait for other resources to become available for use
- **No preemption** means the resources that have already been granted access in advance and cannot be taken away at that time.

If you are aware of these four conditions, then you can follow them and hopefully avoid a deadlock from occurring altogether. There can also be a slight variation in a deadlock situation in which there are two or more processes that are in a constantly changing state, which is known as a **livelock**. The critical difference here is that the processes at play have not actually stopped at all but have instead just become too busy to respond to each other.

- ⇨ **Banker's Algorithm** is used majorly in the banking system to avoid deadlock. It helps you to identify whether a loan will be given or not. This algorithm is used to test for safely simulating the allocation for determining the maximum amount available for all resources. It also checks for all the possible activities before determining whether allocation should be continued or not. For example, there are X number of account holders of a specific bank, and the total amount of money of their accounts is G. When the bank processes a car loan, the software system subtracts the amount of loan granted for purchasing a car from the total money (G+ Fixed deposit + Monthly Income Scheme + Gold, etc.) that the bank has. It also checks that the difference is more than or not G. It only processes the car loan when the bank has sufficient money even if all account holders withdraw the money G simultaneously.
- ⇨ **Banker's Algorithm Notations** :- Here is an important notation used in Banker's algorithm
- ⇨ X: Indicates the total number of processes of the system.

⇨ Y: Indicates the total number of resources present in the system.

Available

[I: Y] indicate which resource is available.

Max

[I:X,I: Y]: Expression of the maximum number of resources of type j or process i

Allocation

[I:X,I:Y]. Indicate where process you have received a resource of type j

Need

Express how many more resources can be allocated in the future.

2.)

b.) Question :- What do you understand by Virtual Environment & Machine Aggregation?

Answer:- Abstract

Virtual environment displays arose from vehicle simulation and teleoperations technology of the 1960s. They are interactive, head-referenced computer displays that give users the illusion of displacement to another location. Different terms have been applied to the illusion. Some, like the oxymoronic "artificial reality" and "virtual reality", suggest much higher performance than current technology can generally provide. Others, like "cyberspace" are puzzling neologisms. Expressions like "virtual worlds" and "virtual environment" seem preferable because they are linguistically conservative, relating to well-established terms like virtual image. In fact, we can define virtual environments as interactive, virtual image displays enhanced by special processing and by nonvisual display modalities, such as auditory and haptic, to convince users that they are immersed in a synthetic space. Why are these displays useful? Who uses them? How are they developed? The article addresses these and other questions related to this emerging technology.
Virtual machines (VM) have become a fixture of many business networks, thanks to their flexibility and cost-effectiveness. But what's a VM, and why are they so useful? This article aims to answer some of your questions about VMs: what's a virtual machine, who makes them, and why they're so useful. I'll also address your concerns about the VM management process, which at first might seem overwhelming but is much easier with a dedicated software tool.

There are many reasons why your company might consider using virtual machines. VMs allow for reduced overhead, with multiple systems operating from the same console at the same time. VMs also provide a safety net for your data, as they can be used to enable rapid disaster recovery and automatic backups. For large and growing businesses, the scalability of virtual environments can be crucial to accommodate the growing pains of a constantly expanding IT environment.

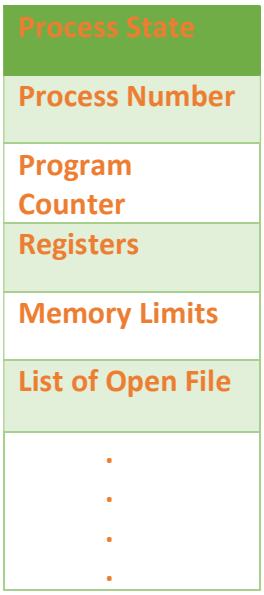
b.) Question :- What is a Process Control Block? What information is stored in it?

Answer :- 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 −



Process control Block (PCB)

Data (information) is stored in computers as Files.

At the core of the computer is the central processing unit or CPU, the source of control that runs all programs and instructions. In order to function, computers use two types of memory: primary and secondary. The main storage is the primary memory, and data and programs are stored in secondary memory.

Data is stored as lots of binary numbers, by magnetism, electronics or optics. ... The computer's operating system, for example, contains instructions for organizing data into files and folders, managing temporary data storage, and sending data to application programs and devices such as printers.

Magnetic storage is commonly used on the hard disc drives found on most computers. Information is stored using positive and negative magnetic charges to correspond with the 1s and 0s noted

above. Optical discs like CDs and DVDs store information as binary code that can be read by an optical sensor in a disc drive.

3.) Question :- Discuss the CPU scheduling algorithms?

Answer :- CPU Scheduling algorithm is an algorithm which is used to assign system resources to processes in a computing system. Consider the case where you are using two apps namely a game like Fortnite and a desktop application like Evernote. Both with require the use of a graphics processor and but only one can use it at a time. It is the CPU scheduling algorithms which manages which process will use a given resource at a time. The focus of such algorithms is to maximize CPU resources usage and minimize waiting time for each process.

The different CPU algorithms are:

- First Come First Serve
- Shortest Job First
- Shortest Remaining Time First
- Round Robin Scheduling
- Priority Scheduling
- Multilevel Queue Scheduling
- Multilevel Feedback Queue Scheduling

Introduction

We will go through some basic information regarding CPU scheduling and some keywords that are frequently used in CPU scheduling algorithms so that we can understand the different algorithms better.

When the CPU is free, and its resources are available, then the CPU must select a process from the ready queue and allocate resources for its execution. Selection is performed with the help of CPU schedulers where the CPU scheduler selects a process from the list of available processes (processes which are present in the ready queue).

We need algorithms for this task to ensure the computing device is able to use all its resources to the fullest.

Objectives of CPU Scheduling:

1. Maximum CPU utilization
2. Fair allocation of CPU

3. Maximum throughput (number of processes executing per second)
4. Minimum turn around time (time taken to finish execution)
5. Minimum waiting time (time for which process waits in ready queue)
6. Minimum Response Time (time when process produces first response)

Key terms to understand different algorithms:

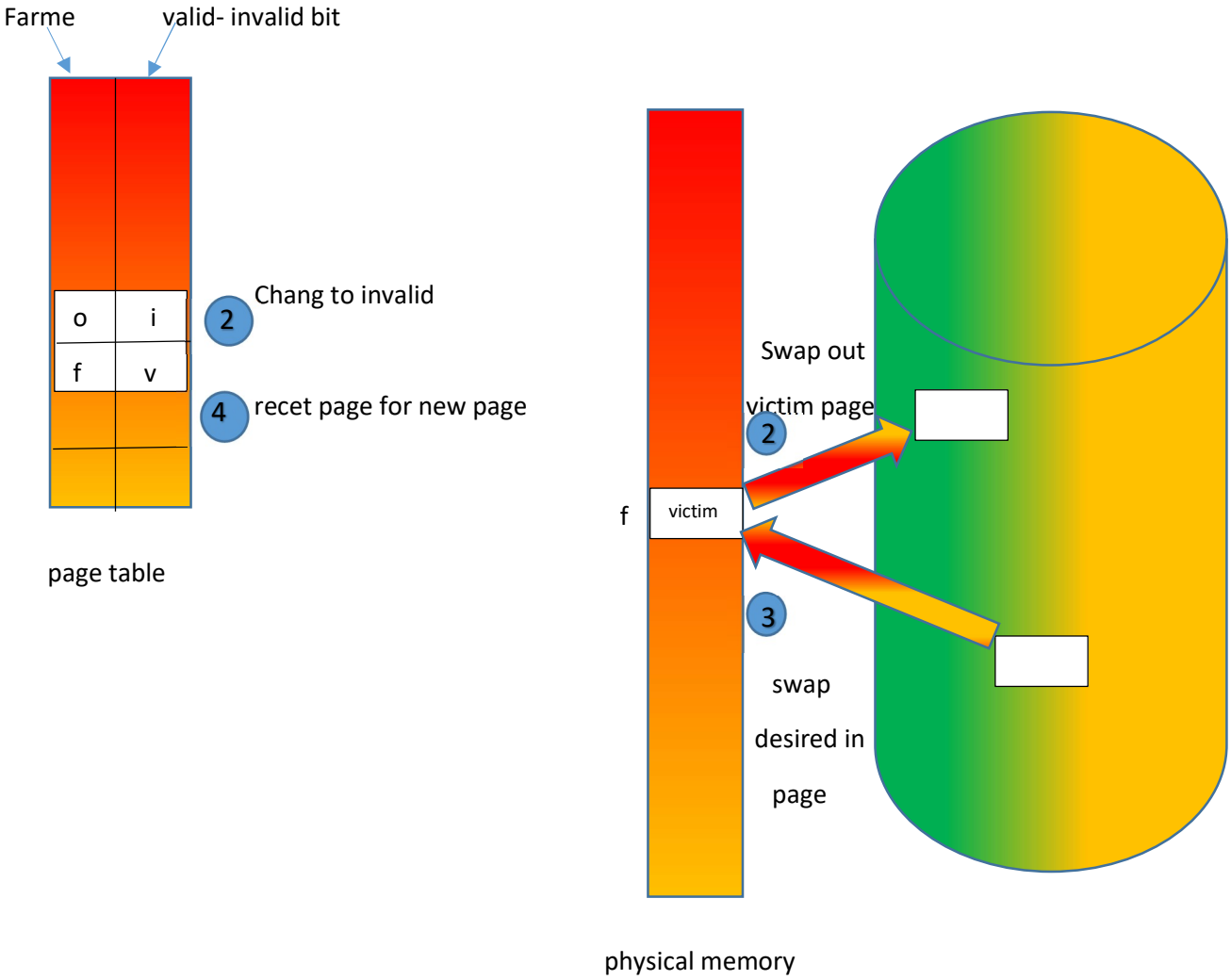
- **Arrival Time :** Time at which any process arrives in ready queue.
- **Burst Time :** Time required by CPU for execution of a process. It is also called as *Running Time* or *Execution Time*.
- **Completion Time :** Time at which process completes execution.
- **Turn Around Time :** Time difference between Completion Time and Arrival Time (*Completion Time - Arrival Time*)
- **Waiting Time :** Time difference between Turn Around Time and Burst Time (*Turn Around Time - Burst Time*)
- **Response Time :** Time after which any process gets CPU after entering the ready queue.
- **Preemptive Scheduling :** It is used if there is process switching from running state to ready state or from ready state to waiting state.
- Resources allocated to a process are for a limited time.
- Process can be interruped in between.
- In case, high priority process arrives, low priority processes may starve.
- It is flexible in nature.
- **Non-Preemptive Scheduling :** It is used if any process terminates or there is process switching from running to waiting state.
- Resources allocated to a process arehold untill it terminates or it switches to waiting state.
- Process can't be interrupted in between.
- In case, process with high burst time is running, other processes may starve.
- It is rigid in nature.

SET – II

4.)

a.) Question :- What is Page replacement? Discuss it's FIFO algorithm with an example?

Answer :- The page replacement approach is introduced when no frame is available free, requiring the user to perform a search operation that is no longer used and has been freed. Users can free a frame by writing its contents to swap space and changing the page table (and every other table as well) to indicate that the page is no longer in memory



Users will now be able to use the free frame to hold the page for which the process has failed. The user changes the page-fault administration routine to include page replacement:

1. First find the location of the desired page on the disk.

2. Then find a free frame:
    1. If there is a free frame available, use it.
    2. If there is no free frame available, use the page-replacement algorithm to select a victim frame.
    3. Write the victim frame to the disk; change the page and frame tables as needed.
3. Read the desired page into the recently free frame; change page and frame tables.
4. Restart the user process.

Note that, if no frame is free, two-page transfers are required: one outside and another one inside. This situation probably doubles the page-fault service time and extends the compelling access time when needed. A user can reduce this overhead by using an adjusted bit named dirty bit. At this point when this scheme is used, there is a corresponding adjusted bit of hardware on each page or frame. The adjusted bit for a page is set by the hardware whenever any word or byte in the page is written into, indicating that the page has been adjusted. At the point when we select a page for replacement, the user inspects its changed bit. If the bit is set, we know that the page has been modified since it was read in from the disk. In this case, we must write the page to the disk. If the modified bit is not set, however, the page has not been modified since it was read into memory. For this situation, we need not write the memory page to the disk: it is already there. This strategy additionally applies only to pages (e.g., pages of binary code). Such pages cannot be changed; therefore, they can be disposed of when required. This scheme can essentially reduce the time required to support a page fault because it reduces the I/O time in half if the page is not adjusted.

In other words page replacement is a process of swapping out a currently existing page from the frame of the main memory and replacing it with the required page. To achieve this task users require page replacement algorithms that decide which memory page is to be replaced. The process of replacement is sometimes called swap out or write to disk. Page replacement is performed when the requested page is not found in the main memory (page fault).

- ⇨ FIFO:- First In, First Out (FIFO) is part of an accounting method where assets which are acquired first are sold of first. The method FIFO considers the inventory as consisting of items bought in the end. The method of FIFO is contrary to another method LIFO in which goods purchased at last are sold first.

In general w.r.t. the FIFO method, the older costs or those lower are assigned as costs to the goods sold under inflationary market conditions. This results in an increase in the net income of the company. The balance quantity of inventory will include goods which are bought last or purchased in the recent past.

The assignment of costs happens at the time of a sale. The assignment happens in the order in which the goods are purchased or in the order in which the goods are manufactured. The FIFO method requires that what comes in first goes out first.

For example, if a batch of 1,000 items gets manufactured in the first week of a month, and another batch of 1,000 in the second week, then the batch produced first gets sold first.

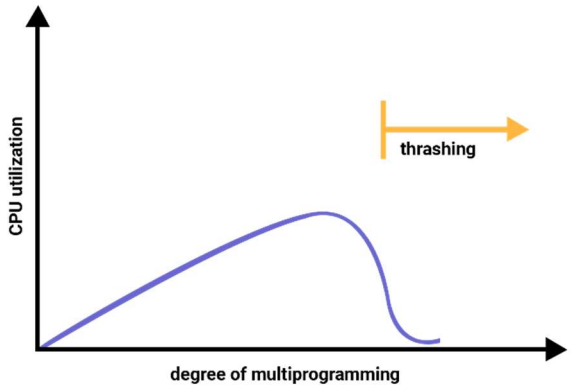
The logic behind the FIFO method is to avoid obsolescence of inventory. Hence, a company sells or assigns the cost of the first batch of production to the first sale. Accordingly, the oldest goods get disposed of first while the new ones remain in stock. An entity must choose a method carefully and should follow it consistently.

b.) Question :- What is Thrashing? What are its causes?

Answer :- In case, if the page fault and swapping happens very frequently at a higher rate, then the operating system has to spend more time swapping these pages. This state in the operating system is termed thrashing. Because of thrashing the CPU utilization is going to be reduced.

Let's understand by an example, if any process does not have the number of frames that it needs to support pages in active use then it will quickly page fault. And at this point, the process must replace some pages. As all the pages of the process are actively in use, it must replace a page that will be needed again right away. Consequently, the process will quickly fault again, and again, and again, replacing pages that it must bring back in immediately. This high paging activity by a process is called thrashing.

During thrashing, the CPU spends less time on some actual productive work spend more time swapping.



Causes of Thrashing

Thrashing affects the performance of execution in the Operating system. Also, thrashing results in severe performance problems in the Operating system.

When the utilization of CPU is low, then the process scheduling mechanism tries to load many processes into the memory at the same time due to which degree of Multiprogramming can be increased. Now in this situation, there are more processes in the memory as compared to the available number of frames in the memory. Allocation of the limited amount of frames to each process.

Whenever any process with high priority arrives in the memory and if the frame is not freely available at that time then the other process that has occupied the frame is residing in the frame will move to secondary storage and after that this free frame will be allocated to higher priority process.

We can also say that as soon as the memory fills up, the process starts spending a lot of time for the required pages to be swapped in. Again the utilization of the CPU becomes low because most of the processes are waiting for pages.

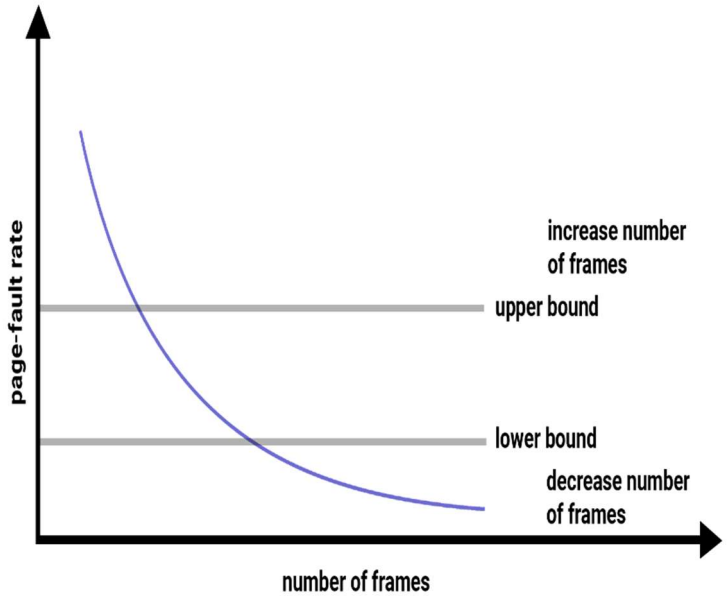
Thus a high degree of multiprogramming and lack of frames are two main causes of thrashing in the Operating system.

Effect of Thrashing

At the time, when thrashing starts then the operating system tries to apply either the **Global page replacement** Algorithm or the **Local page replacement** algorithm.

Global Page Replacement

The Global Page replacement has access to bring any page, whenever thrashing found it tries to bring more pages. Actually, due to this, no process can get enough frames and as a result, the thrashing will increase more and more.



5.) a.) Question :- Discuss the different File Access Methods ?

Answer :- A file is basically a sequence of bytes organized into blocks that are understandable by any machines. In other words, the collection of related

information that is stored in a secondary storage device is also called a file. The file is a collection of logically related entities. According to the users view a file is the smallest allots space of the logical secondary storage. The object that stores data, information, settings or commands used with a computer program on a computer is called file. In graphical user interface (GUI) such as Microsoft Windows, files display as icons that relate to the program that opens the file.

Attributes of file

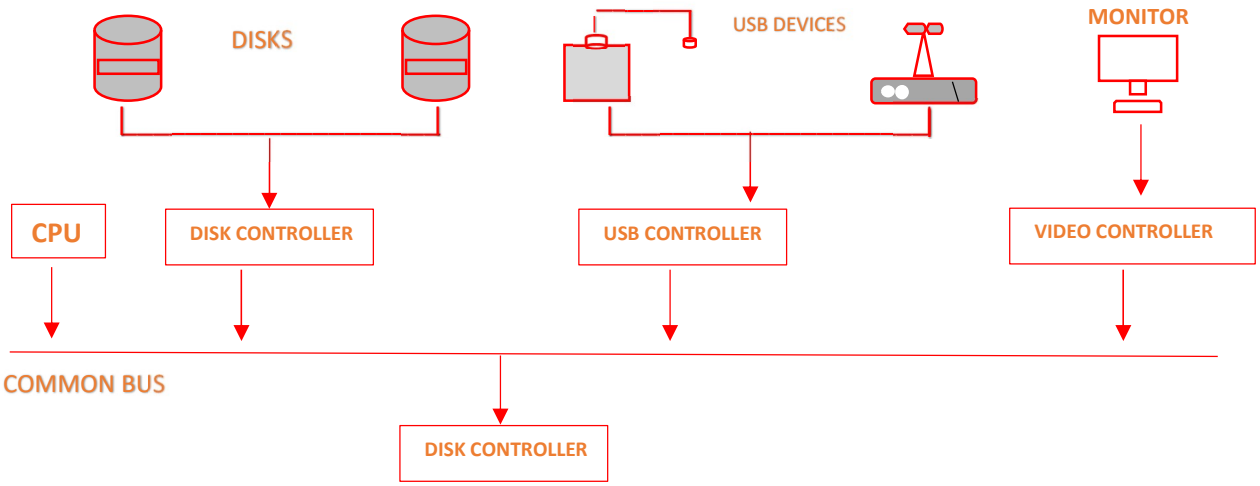
A file is referred by its name, a name or string is a collection of character. There are many systems which differentiate the upper and lower case of the alphabet. When a file is saved by any name it becomes independents for the user and file name should be unique.

- **Name** - The name of any file is the information which is in readable form for the users.
- **Identifier** - The identifier is a unique tag number which identifies the file within the file system. It is not possible for the human to read the identifier or tag number.
- **Type** - type is needed for the system that supports different types of file.
- **Size** - In this attribute, the current or the maximum size of the file is included(in bytes, words, blocks).
- **Location** - This attribute of the file is the pointer to the file and the location of the file where the file is stored on that device.
- **Protection** - this information determines the control and assigns the power of reading, writing and executing. It also defines that who can do the reading, writing.
- **Time date and user identification** - This type of data is useful for the protection, security and usage monitoring. It also defines the last creation, last modification and last use of the file.

b.) Question :- What are I/O Control Strategies?

Answer :-

Operating System has a certain portion of code that is dedicated to managing Input/Output in order to improve the reliability and the performance of the system. A computer system contains CPUs and more than one device controllers connected to a common bus channel, generally referred to as the device driver. These device drivers provide an interface to I/O devices for communicating with the system hardware promoting ease of communication providing access to shared memory.



I/O Requests in operating systems :
I/O Requests are managed by Device Drivers in collaboration with some system programs inside the I/O device. The requests are served by OS using three simple segments :

- **I/O Traffic Controller :** Keeps track of the status of all devices, control units, and communication channels.
- **I/O scheduler :** Executes the policies used by OS to allocate and access the device, control units, and communication channels.
- **I/O device handler :** Serves the device interrupts and heads the transfer of data.

I/O Scheduling in operating systems :
Scheduling is used for efficient usage of computer resources avoiding deadlock and serving all processes waiting in the queue. To know more about CPU Scheduling refer to CPU Scheduling in Operating Systems.

I/O Traffic Controller has 3 main tasks :

- The primary task is to check if there's at least one path available.
- If there exists more than one path, it must decide which one to select.
- If all paths are occupied, its task is to analyze which path will be available at the earliest.

I/O Scheduler functions similar to Process scheduler, it allocates the devices, control units, and communication channels. However, under heavy load of I/O requests, Scheduler must decide what request should be served first and for that we multiple queues to be managed by OS.
The major difference between **Process scheduler**< and **I/O scheduler** is that I/O requests are not pre-empted: Once the channel program has started, it's allowed to continue to completion. Although it is feasible because programs are relatively short (50 to 100 ms). Some modern OS allows I/O Scheduler to serve and higher priority requests. In simpler words, If an I/O request has higher priority then they are served before other I/O requests with lower priority. I/O scheduler works in coordination with the I/O traffic controller to keep track of which path is being served for the current I/O request.

6.) Question :- Discuss about Distributed processing and parallel processing. State the similarities and differences amongst them?

Answer :- Parallel computing is a model that divides a task into multiple sub-tasks and executes them simultaneously to increase the speed and efficiency.

Here, a problem is broken down into multiple parts. Each part is then broke down into a number of instructions.

These parts are allocated to different processors which execute them simultaneously. This increases the speed of execution of programs as a whole.

Distributed computing is different than parallel computing even though the principle is the same.

Distributed computing is a field that studies distributed systems. Distributed systems are systems that have multiple computers located in different locations.

These computers in a distributed system work on the same program. The program is divided into different tasks and allocated to different computers.

The computers communicate with the help of message passing. Upon completion of computing, the result is collated and presented to the user.

Distributed Computing vs. Parallel Computing: A Quick Comparison

Having covered the concepts, let's dive into the differences between them:

Number of Computer Systems Involved

Parallel computing generally requires one computer with multiple processors. Multiple processors within the same computer system execute instructions simultaneously.

All the processors work towards completing the same task. Thus they have to share resources and data.

In distributed computing, several computer systems are involved. Here multiple autonomous computer systems work on the divided tasks.

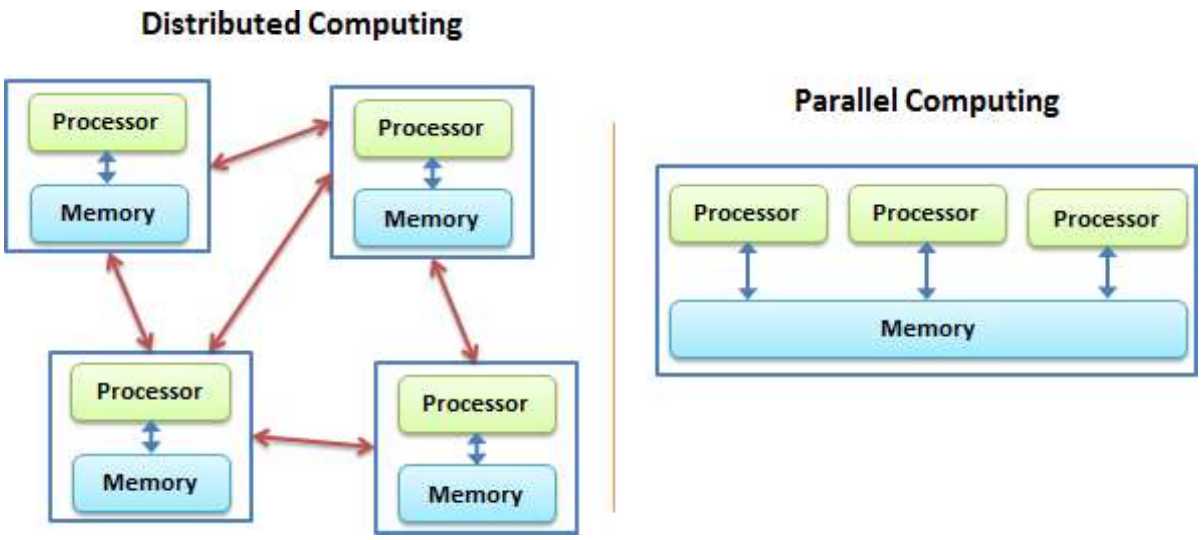
These computer systems can be located at different geographical locations as well.

Dependency Between Processes

In parallel computing, the tasks to be solved are divided into multiple smaller parts. These smaller tasks are assigned to multiple processors.

Here the outcome of one task might be the input of another. This increases dependency between the processors. We can also say, parallel computing environments are tightly coupled.

Some distributed systems might be loosely coupled, while others might be tightly coupled.


Distributed Computing
Processor
Memory
Processor
Memory
Processor
Memory
Processor
Memory
Parallel Computing
Processor
Processor
Processor
Memory