Operating Systems

- **Manthan Godha**

Functions of 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.

Important functions of an operating System:

1. **Security –**
   The operating system uses password protection to protect user data and similar other techniques. it also prevents unauthorized access to programs and user data.

2. **Control over system performance –**
   Monitors overall system health to help improve performance. records the response time between service requests and system response to having a complete view of the system health. This can help improve performance by providing important information needed to troubleshoot problems.

3. **Job accounting –**
   Operating system Keeps track of time and resources used by various tasks and users, this information can be used to track resource usage for a particular user or group of users.

4. **Error detecting aids –**
   The operating system constantly monitors the system to detect errors and avoid the malfunctioning of a computer system.

5. **Coordination between other software and users –**
   Operating systems also coordinate and assign interpreters, compilers, assemblers, and other software to the various users of the computer systems.

6. **Memory Management –**
   The operating system manages the Primary Memory or Main Memory. Main memory is made up of a large array of bytes or words where each byte or word is assigned a certain address. Main memory is fast storage and it can be accessed directly by the CPU. For a program to be executed, it should be first loaded in the main memory. An Operating System performs the following activities for memory management:
   It keeps track of primary memory, i.e., which bytes of memory are used by which user program. The memory addresses that have already been

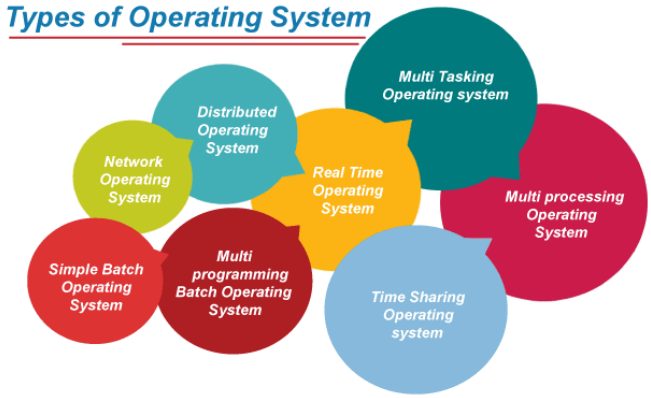
allocated and the memory addresses of the memory that has not yet been used. In multiprogramming, the OS decides the order in which processes are granted access to memory, and for how long. It Allocates the memory to a process when the process requests it and deallocates the memory when the process has terminated or is performing an I/O operation.

7. **Processor Management –**
   In a multi-programming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has. This function of OS is called process scheduling. An Operating System performs the following activities for processor management.
   Keeps track of the status of processes. The program which performs this task is known as a traffic controller. Allocates the CPU that is a processor to a process. De-allocates processor when a process is no more required.

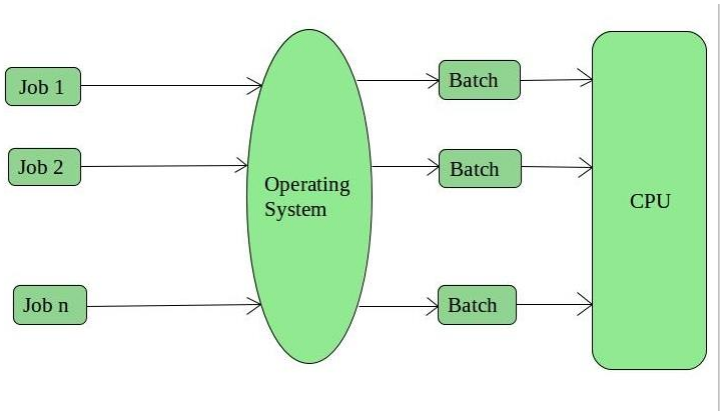
8. **Device Management –**
   An OS manages device communication via their respective drivers. It performs the following activities for device management. Keeps track of all devices connected to the system. designates a program responsible for every device known as the Input/Output controller. Decides which process gets access to a certain device and for how long. Allocates devices in an effective and efficient way. Deallocates devices when they are no longer required.

9. **File Management –**
   A file system is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files. An Operating System carries out the following file management activities. It keeps track of where information is stored, user access settings and status of every file, and more… These facilities are collectively known as the file system.

Types of OS?



Batch Operating System – This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having the same requirement and group them into batches. It is the responsibility of the operator to sort jobs with similar needs.



Advantages of Batch Operating System:

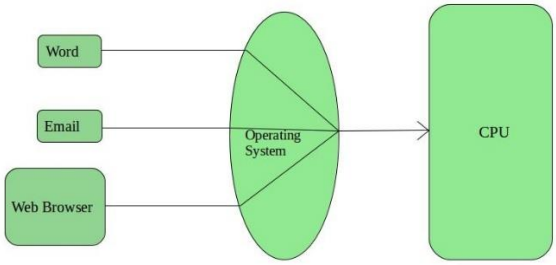
- It is very difficult to guess or know the time required for any job to complete. Processors of the batch systems know how long the job would be when it is in queue
- Multiple users can share the batch systems
- The idle time for the batch system is very less
- It is easy to manage large work repeatedly in batch systems

Disadvantages of Batch Operating System:

- The computer operators should be well known with batch systems
- Batch systems are hard to debug
- It is sometimes costly
- The other jobs will have to wait for an unknown time if any job fails.

Examples of Batch based Operating System: Payroll System, Bank Statements, etc.

Time-Sharing Operating Systems – Each task is given some time to execute so that all the tasks work smoothly. Each user gets the time of CPU as they use a single system. These systems are also known as Multitasking Systems. The task can be from a single user or different users also. The time that each task gets to execute is called quantum. After this time interval is over OS switches over to the next task.



Advantages of Time-Sharing OS:

- Each task gets an equal opportunity
- Fewer chances of duplication of software

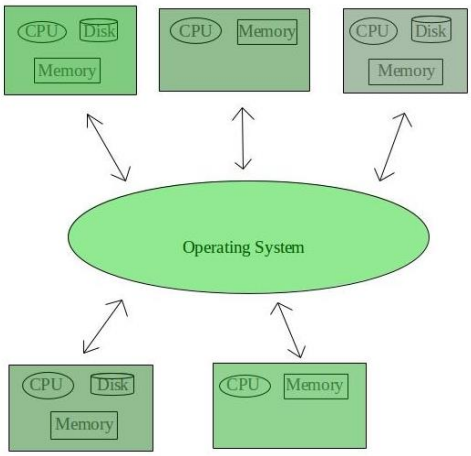
- CPU idle time can be reduced

Disadvantages of Time-Sharing OS:

- Reliability problem
- One must have to take care of the security and integrity of user programs and data
- Data communication problem

Examples of Time-Sharing OSs are: Multics, Unix, etc.

Distributed Operating System – These types of the operating system is a recent advancement in the world of computer technology and are being widely accepted all over the world and, that too, with a great pace. Various autonomous interconnected computers communicate with each other using a shared communication network. Independent systems possess their own memory unit and CPU. These are referred to as **loosely coupled systems** or distributed systems. These system's processors differ in size and function. The major benefit of working with these types of the operating system is that it is always possible that one user can access the files or software which are not actually present on his system but some other system connected within this network i.e., remote access is enabled within the devices connected in that network.



Advantages of Distributed Operating System:

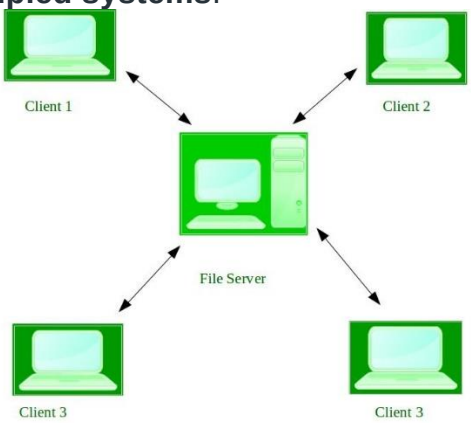
- Failure of one will not affect the other network communication, as all systems are independent from each other
- Electronic mail increases the data exchange speed
- Since resources are being shared, computation is highly fast and durable
- Load on host computer reduces
- These systems are easily scalable as many systems can be easily added to the network
- Delay in data processing reduces

Disadvantages of Distributed Operating System:

- Failure of the main network will stop the entire communication
- To establish distributed systems the language which is used are not well defined yet
- These types of systems are not readily available as they are very expensive. Not only that the underlying software is highly complex and not understood well yet

Examples of Distributed Operating System are- LOCUS, etc.

Network Operating System –
These systems run on a server and provide the capability to manage data, users, groups, security, applications, and other networking functions. These types of operating systems allow shared access of files, printers, security, applications, and other networking functions over a small private network. One more important aspect of Network Operating Systems is that all the users are well aware of the underlying configuration, of all other users within the network, their individual connections, etc. and that's why these computers are popularly known as **tightly coupled systems**.



Advantages of Network Operating System:

- Highly stable centralized servers
- Security concerns are handled through servers
- New technologies and hardware up-gradation are easily integrated into the system
- Server access is possible remotely from different locations and types of systems

Disadvantages of Network Operating System:

- Servers are costly
- User has to depend on a central location for most operations
- Maintenance and updates are required regularly

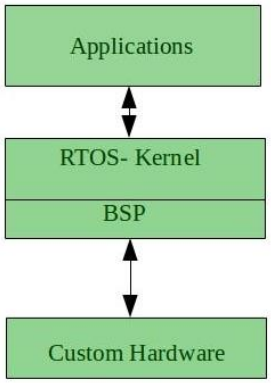
Examples of Network Operating System are: Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD, etc.

Real-Time Operating System – These types of OSs serve real-time systems. The time interval required to process and respond to inputs is very small. This time interval is called **response time**.
Real-time systems are used when there are time requirements that are very strict like missile systems, air traffic control systems, robots, etc.
Two types of Real-Time Operating System which are as follows:

- **Hard Real-Time Systems:**
  These OSs are meant for applications where time constraints are very strict and even the shortest possible delay is not acceptable. These systems are built for saving life like automatic parachutes or airbags

which are required to be readily available in case of any accident. Virtual memory is rarely found in these systems.

- **Soft Real-Time Systems:**
  These OSs are for applications where for time-constraint is less strict.



Advantages of RTOS:

- **Maximum Consumption:** Maximum utilization of devices and system, thus more output from all the resources
- **Task Shifting:** The time assigned for shifting tasks in these systems are very less. For example, in older systems, it takes about 10 microseconds in shifting one task to another, and in the latest systems, it takes 3 microseconds.
- **Focus on Application:** Focus on running applications and less importance to applications which are in the queue.
- Real-time **operating system in** the **embedded system:** Since the size of programs are small, RTOS can also be used in embedded systems like in transport and others.
- **Error Free:** These types of systems are error-free.
- **Memory Allocation:** Memory allocation is best managed in these types of systems.

Disadvantages of RTOS:

- **Limited Tasks:** Very few tasks run at the same time and their concentration is very less on few applications to avoid errors.
- **Use heavy system resources:** Sometimes the system resources are not so good and they are expensive as well.
- **Complex Algorithms:** The algorithms are very complex and difficult for the designer to write on.
- **Device driver and interrupt signals:** It needs specific device drivers and interrupts signals to respond earliest to interrupts.
- **Thread Priority:** It is not good to set thread priority as these systems are very less prone to switching tasks.

Examples of Real-Time Operating Systems are: Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

Demand Paging?

According to the concept of Virtual Memory, in order to execute some process, only a part of the process needs to be present in the main memory which means that only a few pages will only be present in the main memory at any time.

However, deciding, which pages need to be kept in the main memory and which need to be kept in the secondary memory, is going to be difficult because we cannot say in advance that a process will require a particular page at particular time.

Therefore, to overcome this problem, there is a concept called Demand Paging is introduced. It suggests keeping all pages of the frames in the secondary memory until they are required. In other words, it says that do not load any page in the main memory until it is required.

Whenever any page is referred for the first time in the main memory, then that page will be found in the secondary memory.

What is a Page Fault?

If the referred page is not present in the main memory then there will be a miss and the concept is called Page miss or page fault.

The CPU has to access the missed page from the secondary memory. If the number of page fault is very high then the effective access time of the system will become very high.

What is Thrashing?

If the number of page faults is equal to the number of referred pages or the number of page faults are so high so that the CPU remains busy in just reading the pages from the secondary memory then the effective access time will be the time taken by the CPU to read one word from the secondary memory and it will be so high. The concept is called thrashing.

Process Synchronization?

When two or more process cooperates with each other, their order of execution must be preserved otherwise there can be conflicts in their execution and inappropriate outputs can be produced.

A cooperative process is the one which can affect the execution of other process or can be affected by the execution of other process. Such processes need to be synchronized so that their order of execution can be guaranteed.

The procedure involved in preserving the appropriate order of execution of cooperative processes is known as Process Synchronization.

A Race Condition typically occurs when two or more threads try to read, write and possibly make the decisions based on the memory that they are accessing concurrently.

The regions of a program that try to access shared resources and may cause race conditions are called critical section. To avoid race condition among the processes, we need to assure that only one process at a time can execute within the critical section.

Semaphores?

Semaphores are a synchronization mechanism used to coordinate the activities of multiple processes in a computer system. They are used to enforce mutual exclusion, avoid race conditions and implement synchronization between processes.

Semaphores provide two operations: wait (P) and signal (V). The wait operation decrements the value of the semaphore, and the signal operation increments the value of the semaphore. When the value of the semaphore is zero, any process that performs a wait operation will be blocked until another process performs a signal operation.

Semaphores are used to implement critical sections, which are regions of code that must be executed by only one process at a time. By using semaphores, processes can coordinate access to shared resources, such as shared memory or I/O devices.

1. **Binary Semaphore –**
   This is also known as mutex lock. It can have only two values – 0 and 1. Its value is initialized to 1. It is used to implement the solution of critical section problems with multiple processes.
2. **Counting Semaphore –**
   Its value can range over an unrestricted domain. It is used to control access to a resource that has multiple instances.

Threads in OS?

What is a Thread?
Within a program, a thread is a separate execution path. It is a lightweight process that the operating system can schedule and run concurrently with other threads. The operating system creates and manages threads, and they share the same memory and resources as the program that created them. This enables multiple threads to collaborate and work efficiently within a single program.

Why Multithreading?
A thread is also known as lightweight process. The idea is to achieve parallelism by dividing a process into multiple threads. For example, in a browser, multiple tabs can be different threads. MS Word uses multiple threads: one thread to format the text, another thread to process inputs, etc.

The primary difference is that threads within the same process run in a shared memory space, while processes run in separate memory spaces. Threads are not independent of one another like processes are, and as a result threads share with other threads their code section, data section, and OS resources (like open files and signals). But, like process, a thread has its own program counter (PC), register set, and stack space.

Advantages of Thread over Process

1. ***Responsiveness:*** If the process is divided into multiple threads, if one thread completes its execution, then its output can be immediately returned.
2. ***Faster context switch:*** Context switch time between threads is lower compared to process context switch. Process context switching requires more overhead from the CPU.
3. ***Effective utilization of multiprocessor system:*** If we have multiple threads in a single process, then we can schedule multiple threads on multiple processor. This will make process execution faster.
4. ***Resource sharing:*** Resources like code, data, and files can be shared among all threads within a process. Note: stack and registers can't be shared among the threads. Each thread has its own stack and registers.
5. ***Communication:*** Communication between multiple threads is easier, as the threads shares common address space. while in process we have to follow some specific communication technique for communication between two process.
6. ***Enhanced throughput of the system:*** If a process is divided into multiple threads, and each thread function is considered as one job, then the number of jobs completed per unit of time is increased, thus increasing the throughput of the system.

There are two types of threads:

- User Level Thread
- Kernel Level Thread

FCFS?

First come first serve (FCFS) scheduling algorithm simply schedules the jobs according to their arrival time. The job which comes first in the ready queue will get the CPU first. The lesser the arrival time of the job, the sooner will the job get the CPU. FCFS scheduling may cause the problem of starvation if the burst time of the first process is the longest among all the jobs.

Advantages of FCFS

- Simple
- Easy
- First come, First serv

Disadvantages of FCFS

1. The scheduling method is non preemptive, the process will run to the completion.
2. Due to the non-preemptive nature of the algorithm, the problem of starvation may occur.
3. Although it is easy to implement, but it is poor in performance since the average waiting time is higher as compare to other scheduling algorithms.

Round Robin?

Round Robin scheduling algorithm is one of the most popular scheduling algorithm which can actually be implemented in most of the operating systems. This is the **preemptive version** of first come first serve scheduling. The Algorithm focuses on Time Sharing. In this algorithm, every process gets executed in a **cyclic way**. A certain time slice is defined in the system which is called time **quantum**. Each process present in the ready queue is assigned the CPU for that time quantum, if the execution of the process is completed during that time then the process will **terminate** else the process will go back to the **ready queue** and waits for the next turn to complete the execution.

Advantages

1. It can be actually implementable in the system because it is not depending on the burst time.
2. It doesn't suffer from the problem of starvation or convoy effect.
3. All the jobs get a fare allocation of CPU.

Disadvantages

1. The higher the time quantum, the higher the response time in the system.
2. The lower the time quantum, the higher the context switching overhead in the system.
3. Deciding a perfect time quantum is really a very difficult task in the system.

Shortest Job First?

In SJF scheduling, the process with the lowest burst time, among the list of available processes in the ready queue, is going to be scheduled next.

However, it is very difficult to predict the burst time needed for a process hence this algorithm is very difficult to implement in the system.

Advantages of SJF

1. Maximum throughput
2. Minimum average waiting and turnaround time

Disadvantages of SJF

1. May suffer with the problem of starvation
2. It is not implementable because the exact Burst time for a process can't be known in advance.

Diff bet paging and segmentation?

S.NO	Paging	Segmentation
1.	In paging, the program is divided into fixed or mounted size pages.	In segmentation, the program is divided into variable size sections.
2.	For the paging operating system is accountable.	For segmentation compiler is accountable.
3.	Page size is determined by hardware.	Here, the section size is given by the user.
4.	It is faster in comparison to segmentation.	Segmentation is slow.
5.	Paging could result in internal fragmentation.	Segmentation could result in external fragmentation.
6.	In paging, the logical address is split into a page number and page offset.	Here, the logical address is split into section number and section offset.
7.	Paging comprises a page table that encloses the base address of every page.	While segmentation also comprises the segment table which encloses segment number and segment offset.

S.NO	Paging	Segmentation
8.	The page table is employed to keep up the page data.	Section Table maintains the section data.
9.	In paging, the operating system must maintain a free frame list.	In segmentation, the operating system maintains a list of holes in the main memory.
10.	Paging is invisible to the user.	Segmentation is visible to the user.
11.	In paging, the processor needs the page number, and offset to calculate the absolute address.	In segmentation, the processor uses segment number, and offset to calculate the full address.
12.	It is hard to allow sharing of procedures between processes.	Facilitates sharing of procedures between the processes.
13	In paging, a programmer cannot efficiently handle data structure.	It can efficiently handle data structures.
14.	This protection is hard to apply.	Easy to apply for protection in segmentation.
15.	The size of the page needs always be equal to the size of frames.	There is no constraint on the size of segments.
16.	A page is referred to as a physical unit of information.	A segment is referred to as a logical unit of information.
17.	Paging results in a less efficient system.	Segmentation results in a more efficient system.

Diff bet multitasking multiprocessing?

S No.	Multi-tasking	Multiprocessing

1.	**The execution of more than one task simultaneously is known as multitasking.**	**The availability of more than one processor per system, that can execute several set of instructions in parallel is known as multiprocessing.**
2.	**The number of CPU is one.**	**The number of CPUs is more than one.**
3.	**It takes moderate amount of time.**	**It takes less time for job processing.**
4.	**In this, one by one job is being executed at a time.**	**In this, more than one process can be executed at a time.**
5.	**It is economical.**	**It is less economical.**
6.	**The number of users is more than one.**	**The number of users is can be one or more than one.**
7.	**Throughput is moderate.**	**Throughput is maximum.**
8.	**Its efficiency is moderate.**	**Its efficiency is maximum.**
9.	**It is of two types: Single user multitasking and Multiple user multitasking.**	**It is of two types: Symmetric Multiprocessing and Asymmetric Multiprocessing.**

Multiprogramming multithreading?

S.No.	Multiprogramming	Multithreading
1.	The concurrent application of more than one program in the main memory is called as multiprogramming.	The process is divided into several different sub-processes called as threads, which has its own path of execution. This concept is called as multithreading.
2.	It takes more time to process the jobs.	It takes moderate amount of time for job processing.

3.	In this, one process is executed at a time.	In this, various components of the same process are being executed at a time.
4.	The number of users is one at a time.	The number of users usually one.
5.	Throughput is less.	Throughput is Moderate.
6.	Its efficiency is Less.	Its efficiency is moderate.
7.	It is economical.	It is economical.
8.	The number of CPU is one.	The number of CPU can be one or more than one.

Sockets in OS?

Sockets allow communication between two different processes on the same or different machines.

A Unix Socket is used in a client-server application framework. A server is a process that performs some functions on request from a client. Most of the application-level protocols like FTP, SMTP, and POP3 make use of sockets to establish connection between client and server and then for exchanging data.

Starvation?

S.NO	Deadlock	Starvation
1.	All processes keep waiting for each other to complete and none get executed	High priority processes keep executing and low priority processes are blocked
2.	Resources are blocked by the processes	Resources are continuously utilized by high priority processes
3.	Necessary conditions Mutual Exclusion, Hold and Wait, No preemption, Circular Wait	Priorities are assigned to the processes
4.	Also known as Circular wait	Also known as lived lock
5.	It can be prevented by avoiding the necessary conditions for deadlock	It can be prevented by Aging

Kernel?

Kernel is central component of an operating system that manages operations of computer and hardware. It basically manages operations of memory and CPU time. It is core component of an operating system. Kernel acts as a bridge between applications and data processing performed at hardware level using inter-process communication and system calls.
Kernel loads first into memory when an operating system is loaded and remains into memory until operating system is shut down again. It is responsible for various tasks such as disk management, task management, and memory management.

Kernel has a process table that keeps track of all active processes
• Process table contains a per process region table whose entry points to entries in region table.

Kernel loads an executable file into memory during 'exec' system call'.

Objectives of Kernel :

- To establish communication between user level application and hardware.
- To decide state of incoming processes.
- To control disk management.
- To control memory management.
- To control task management.

1. Monolithic Kernel –
It is one of types of kernel where all operating system services operate in kernel space. It has dependencies between systems components. It has huge lines of code which is complex.

Example: Unix, Linux, Open VMS, XTS-400 etc.
Advantage:
It has good performance.
Disadvantage:
It has dependencies between system component and lines of code in millions.

2. Micro Kernel –
It is kernel types which has minimalist approach. It has virtual memory and thread scheduling. It is more stable with less services in kernel space. It puts rest in user space.
Example : Mach, L4, AmigaOS, Minix, K42 etc.

Advantage :
It is more stable.
Disadvantage :
There are lots of system calls and context switches.

Diff between Process and Thread?

S.NO	Process	Thread
1.	Process means any program is in execution.	Thread means a segment of a process.
2.	The process takes more time to terminate.	The thread takes less time to terminate.
3.	It takes more time for creation.	It takes less time for creation.
4.	It also takes more time for context switching.	It takes less time for context switching.
5.	The process is less efficient in terms of communication.	Thread is more efficient in terms of communication.
6.	Multiprogramming holds the concepts of multi-process.	We don't need multi programs in action for multiple threads because a single process consists of multiple threads.
7.	The process is isolated.	Threads share memory.
8.	The process is called the heavyweight process.	A Thread is lightweight as each thread in a process shares code, data, and resources.
9.	Process switching uses an interface in an operating system.	Thread switching does not require calling an operating system and causes an interrupt to the kernel.
10.	If one process is blocked then it will not affect the execution of other processes	If a user-level thread is blocked, then all other user-level threads are blocked.
11.	The process has its own Process Control Block, Stack, and Address Space.	Thread has Parents' PCB, its own Thread Control Block, and Stack and common Address space.
12.	Changes to the parent process do not affect child processes.	Since all threads of the same process share address space and other resources so any changes to the main thread may

S.NO	Process	Thread
		affect the behavior of the other threads of the process.
13.	A system call is involved in it.	No system call is involved, it is created using APIs.
14.	The process does not share data with each other.	Threads share data with each other.

Spooling in OS?

Spooling is a process in which data is temporarily held to be used and executed by a device, program, or system. Data is sent to and stored in memory or other volatile storage until the program or computer requests it for execution.

SPOOL is an acronym for ***simultaneous peripheral operations online***. Generally, the spool is maintained on the computer's physical memory, buffers, or the I/O device-specific interrupts. The spool is processed in ascending order, working based on a FIFO (first-in, first-out) algorithm.

Spooling refers to putting data of various I/O jobs in a buffer. This buffer is a special area in memory or hard disk which is accessible to I/O devices. An operating system does the following activities related to the distributed environment:

- Handles I/O device data spooling as devices have different data access rates.
- Maintains the spooling buffer, which provides a waiting station where data can rest while the slower device catches up.
- Maintains parallel computation because of the spooling process as a computer can perform I/O in parallel order. It becomes possible to have the computer read data from a tape, write data to disk, and write out to a tape printer while it is doing its computing task.

Critical Section?

Critical Section is the part of a program which tries to access shared resources. That resource may be any resource in a computer like a memory location, Data structure, CPU or any IO device.

The critical section cannot be executed by more than one process at the same time; operating system faces the difficulties in allowing and disallowing the processes from entering the critical section.

The critical section problem is used to design a set of protocols which can ensure that the Race condition among the processes will never arise.

In order to synchronize the cooperative processes, our main task is to solve the critical section problem. We need to provide a solution in such a way that the following conditions can be satisfied.