Operating System(OS)

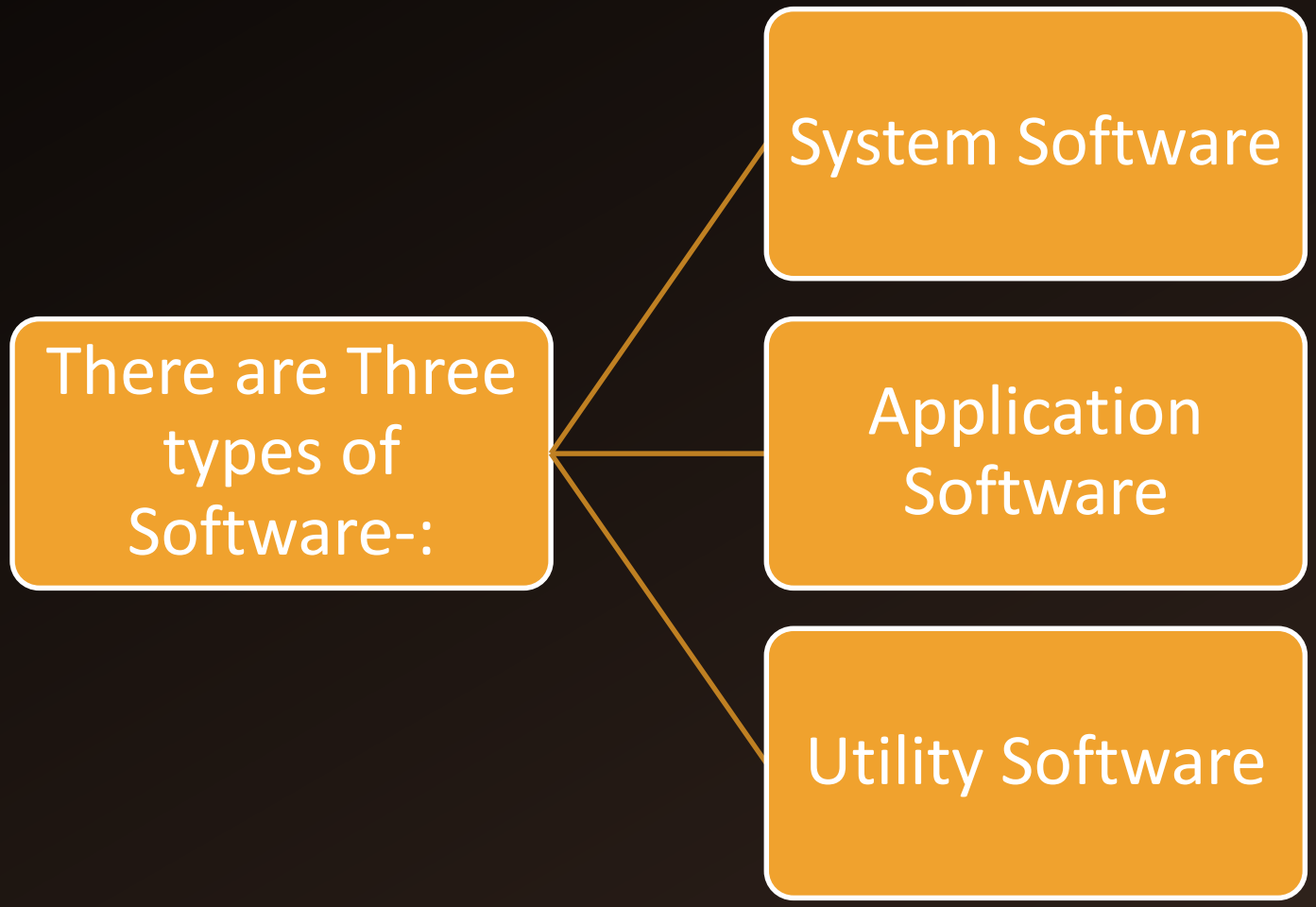
BCA IV SEM OS

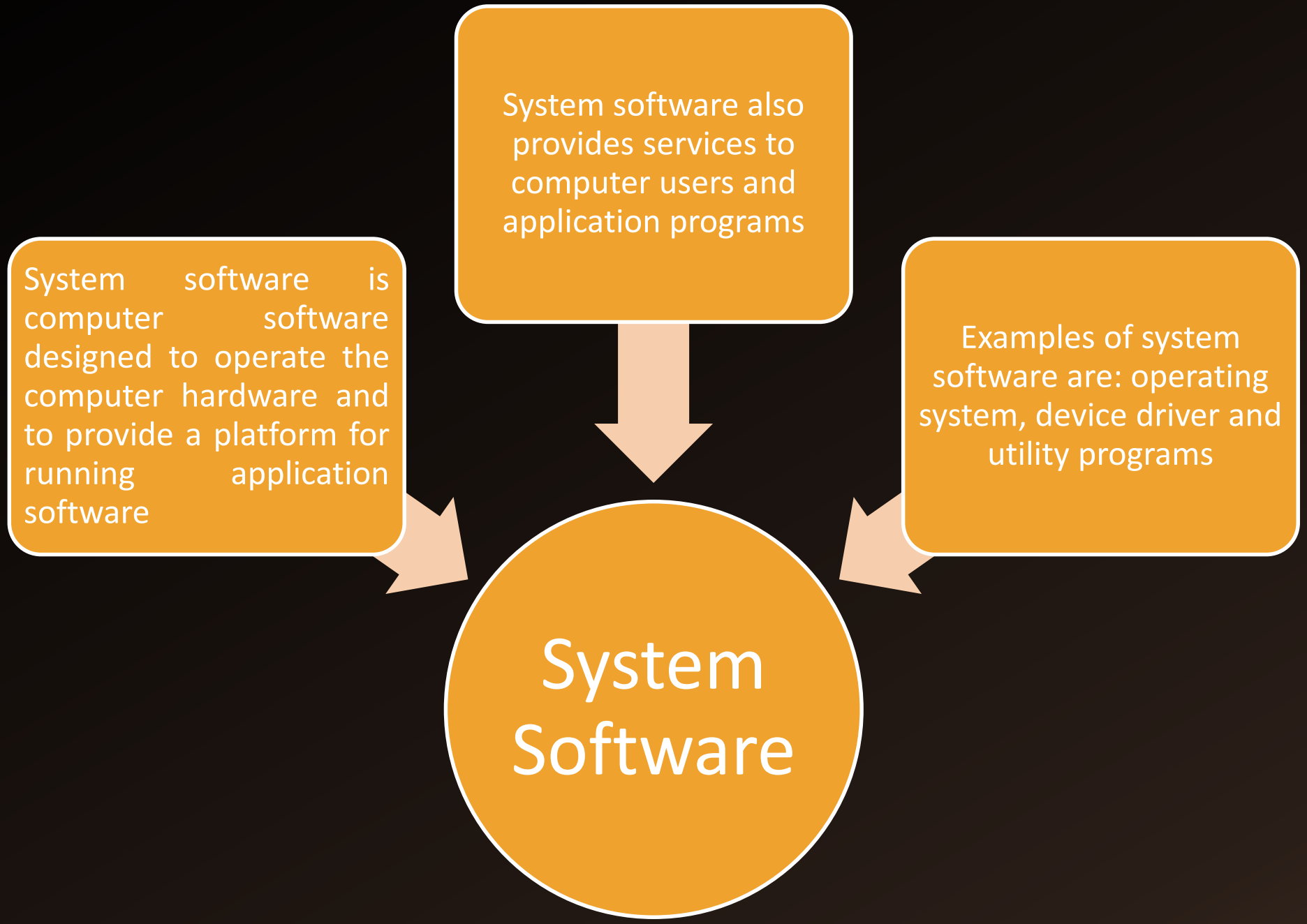
SOFTWARE

AS OPERATING SYSTEM

BASICS OF SOFTWARE

Software is a set of instructions, data or programs used to operate computers and execute specific tasks.




System software also provides services to computer users and application programs
System software is computer software designed to operate the computer hardware and to provide a platform for running application software
Examples of system software are: operating system, device driver and utility programs
System Software

Application software

- Application software (app for short) is a program or group of programs designed for end users.
- Examples of an application include a word processor, a spreadsheet, an accounting application, a web browser, an email client, a media player, a file viewer, simulators, a console Game or a Photo editor.

UTILITY SOFTWARE

is system software Utility Software designed to help analyze, configure, optimize or maintain a computer.

Types of Utility Software:

- Anti Virus Programs
- File managers
- Disk Cleaners

Unit I

(INTRODUCTION TO OPERATING SYSTEM AND ITS CHARACTERISTICS)

Introduction to OS

Understand

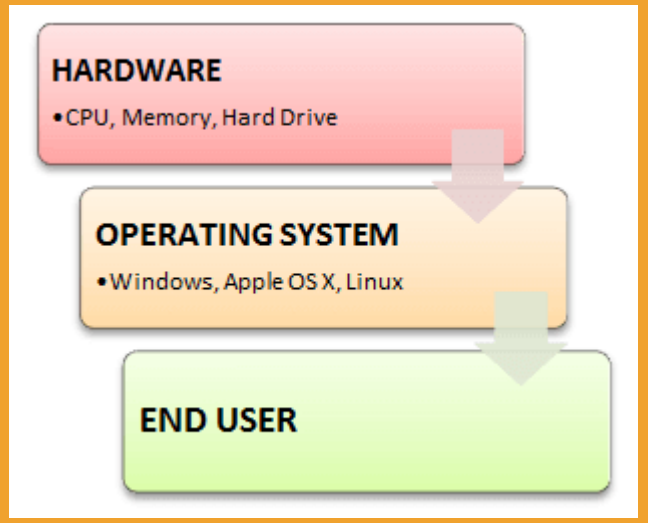
- In the Computer System (comprises of Hardware and software), Hardware can only understand machine code (in the form of 0 and 1) which doesn't make any sense to a naive user.
- We need a system which can act as an intermediary and manage all the processes and resources present in the system.

Definition

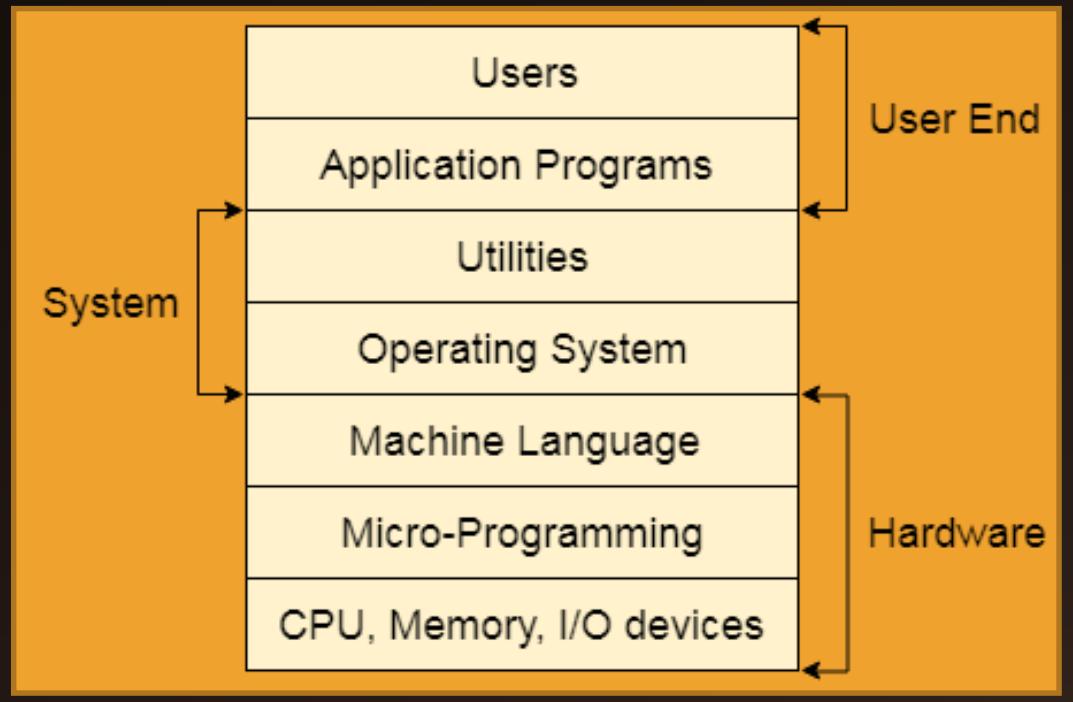
- An Operating System can be defined as an interface between user and hardware. It is responsible for the execution of all the processes, Resource Allocation, CPU management, File Management and many other tasks.
- The purpose of an operating system is to provide an environment in which a user can execute programs in convenient and efficient manner.

components and structure of operating system

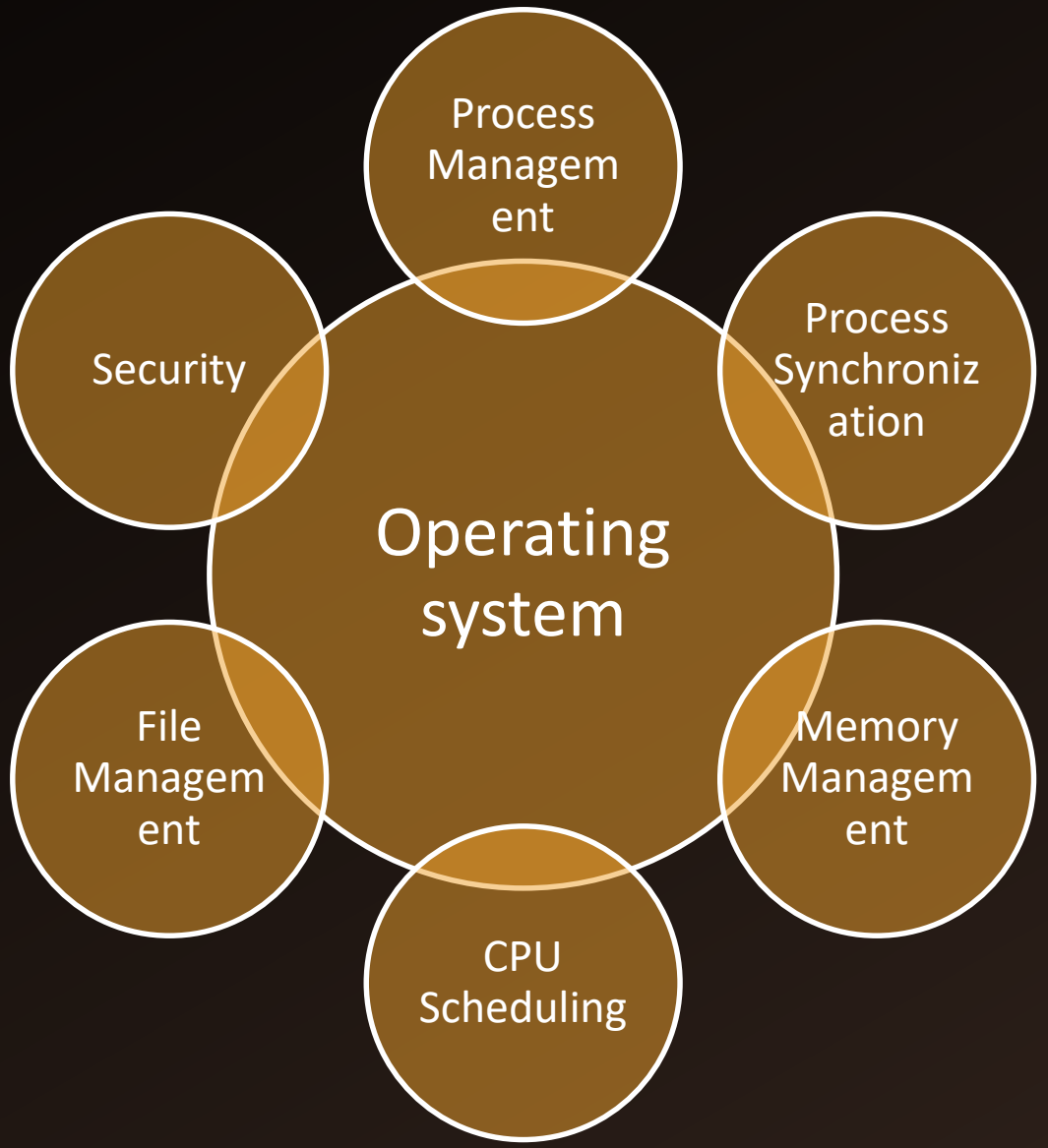
COMPONENTS



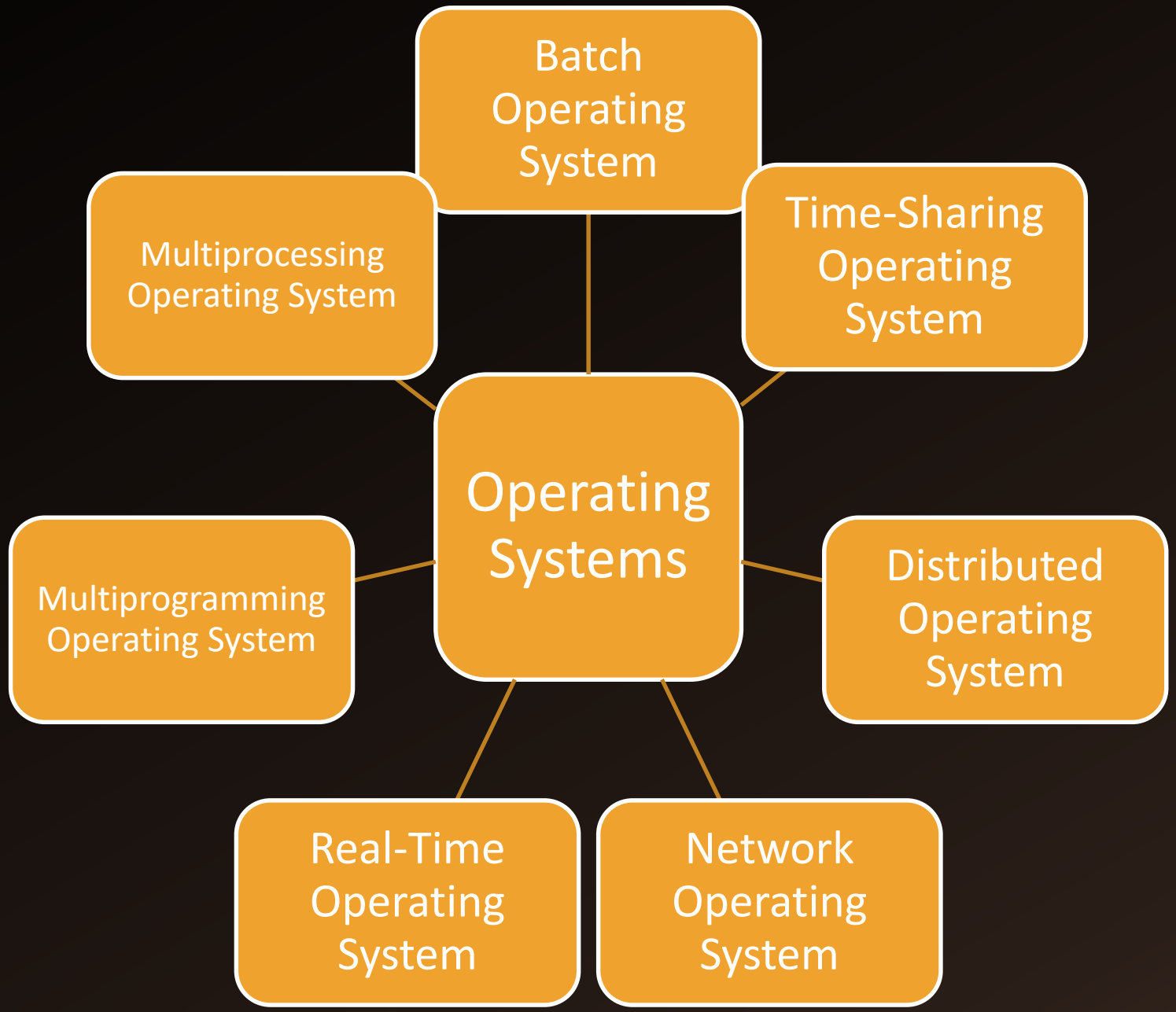
STRUCTURE



What does an Operating system do?



Types of Operating Systems (OS)



BATCH OPERATING SYSTEM:-

In a Batch Operating System, the similar jobs are grouped together into batches with the help of some operator and these batches are executed one by one.

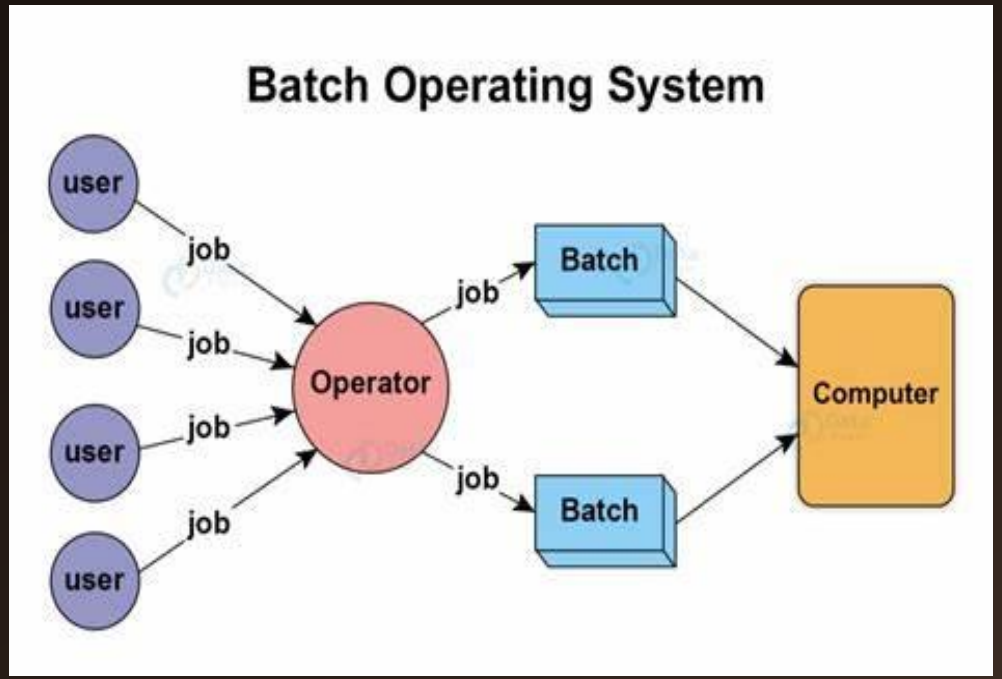
In the 1970s, Batch processing was very popular. In this technique, similar types of jobs were batched together and executed in time. People were used to having a single computer which was called a mainframe.

In Batch operating system, access is given to more than one person; they submit their respective jobs to the system for the execution.

The system put all of the jobs in a queue on the basis of first come first serve and then executes the jobs one by one. The users collect their respective output when all the jobs get executed.

Advantages of Batch OS→ The use of a resident monitor improves computer efficiency as it eliminates CPU time between two jobs.

Disadvantages of Batch OS→ Batch processing take more time .



TIME-SHARING OPERATING SYSTEM:-

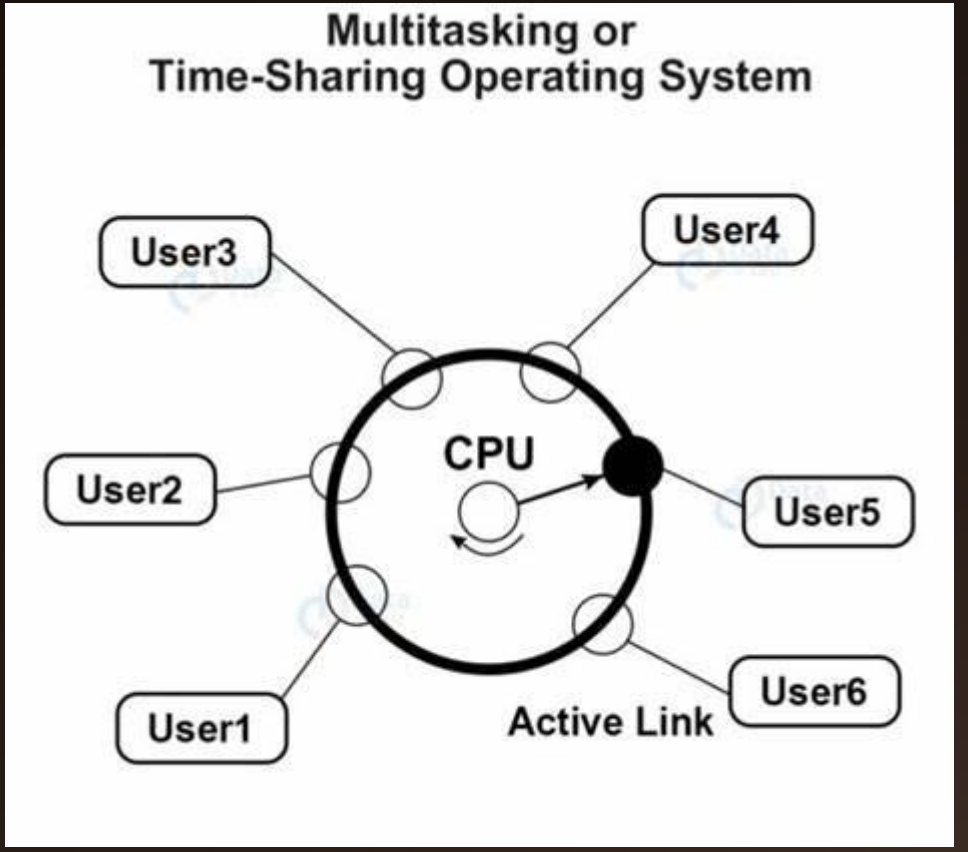
In Time-Sharing Operating System Each task is given some time to execute so that all the tasks work smoothly.

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.

Advantages of Time Sharing Operating System → The time-sharing operating system provides effective utilization and sharing of resources. This system reduces CPU idle and response time.

Disadvantages of Time Sharing Operating System → Data transmission rates are very high in comparison to other methods. Security and integrity of user programs loaded in memory and data need to be maintained as many users access the system at the same time.



MULTIPROGRAMMING OPERATING SYSTEM

Multiprogramming is an extension to batch processing where the CPU is always kept busy. Each process needs two types of system time: CPU time and IO time.

In a multiprogramming environment, when a process does its I/O, The CPU can start the execution of other processes. Therefore, multiprogramming improves the efficiency of the system.

Advantages of Multiprogramming OS→

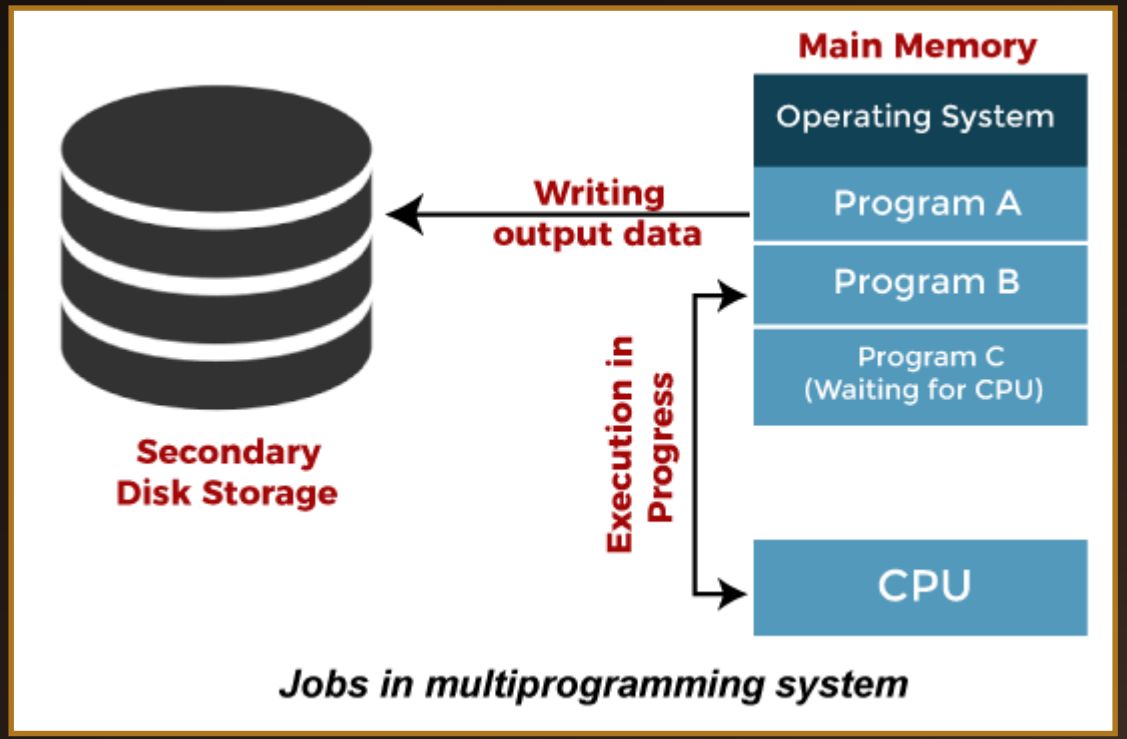
o Throughout the system, it increased as the CPU always had one program to execute.

o Response time can also be reduced.

Disadvantages of Multiprogramming OS→

o Multiprogramming systems provide an environment in which various systems resources are used efficiently, but they

do not provide any user interaction with the computer system.



Jobs in multiprogramming system

MULTIPROCESSING OPERATING SYSTEM

In Multiprocessing, Parallel computing is achieved. There are more than one processors present in the system which can execute more than one process at the same time. This will increase the throughput of the system.

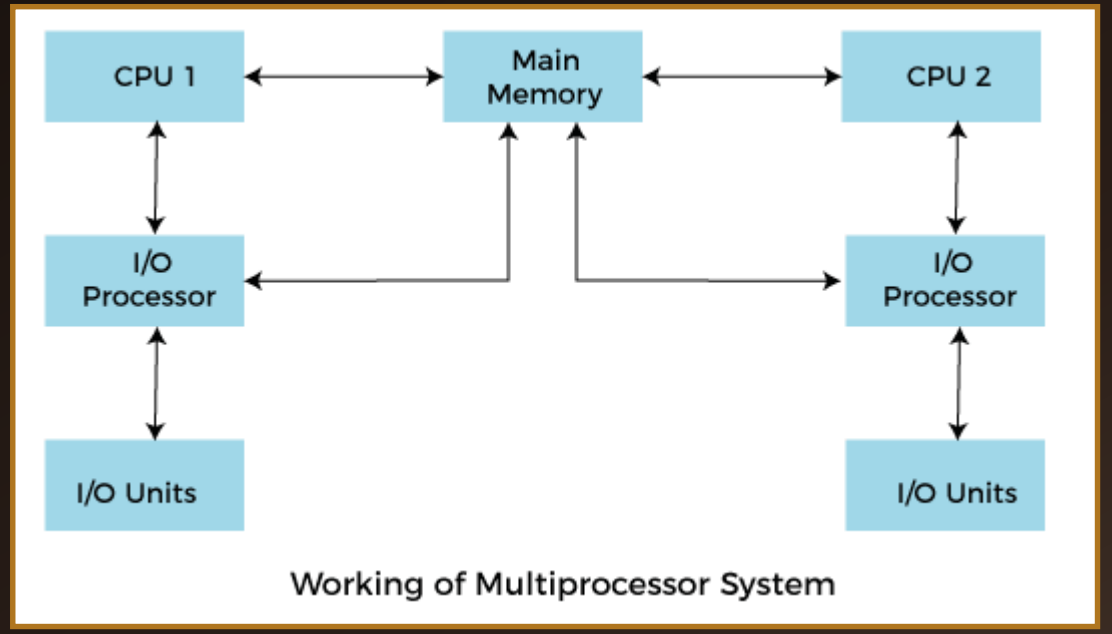
Parallel computing is achieved. More than one processor present in the system can execute more than one process simultaneously, which will increase the throughput of the system.

Advantages of Multiprocessing operating system→

- Increased reliability: Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails, the task can be given to another processor for completion.
- Increased throughout: As several processors increase, more work can be done in less.

Disadvantages of Multiprocessing operating System→

- Multiprocessing operating system is more complex and sophisticated as it takes care of multiple CPUs simultaneously.



Working of Multiprocessor System

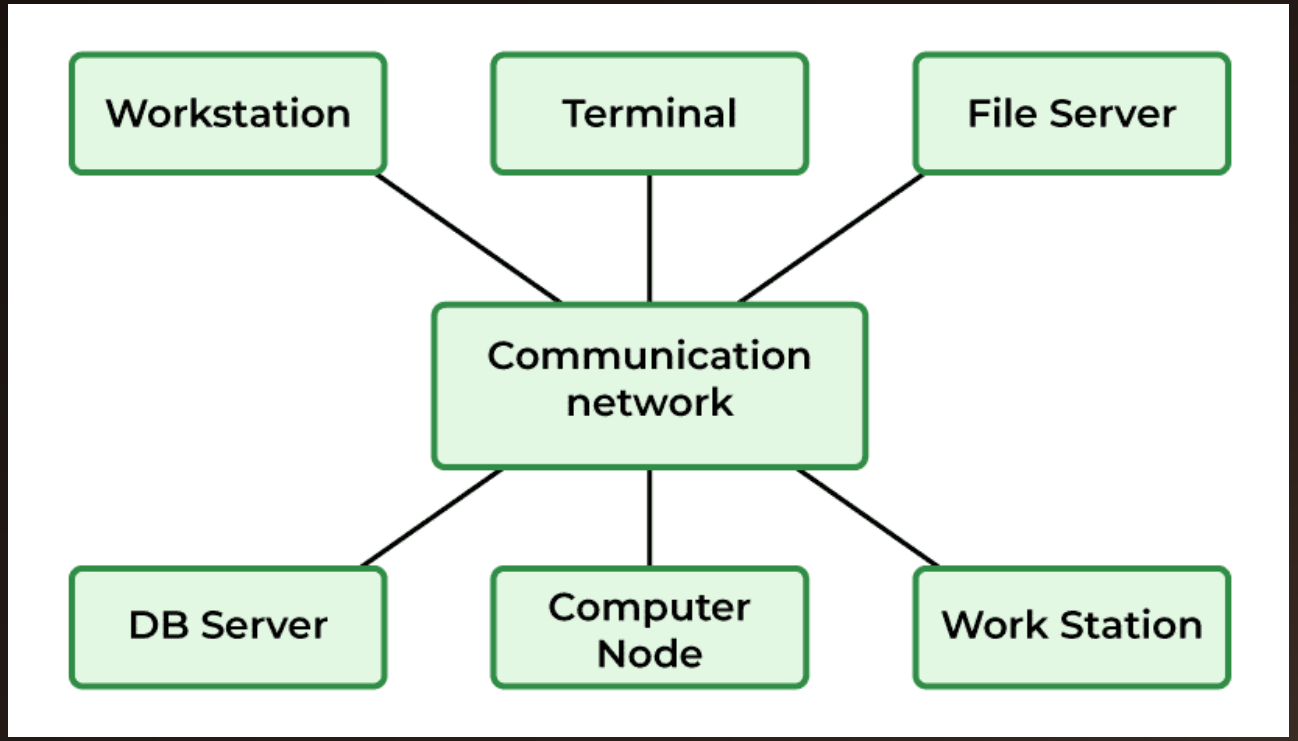
DISTRIBUTED OPERATING SYSTEM

In a Distributed Operating System, e have various systems and all these systems have their own CPU, main memory, secondary memory, and resources.

These systems are connected to each other using a shared communication network. Here, each system can perform its task individually.

Advantages of Distributed Operating System →The distributed operating system provides sharing of resources.
This type of system is fault-tolerant.

Disadvantages of Distributed Operating System →Protocol overhead can dominate computation cost.



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.

Advantages of Network Operating System→

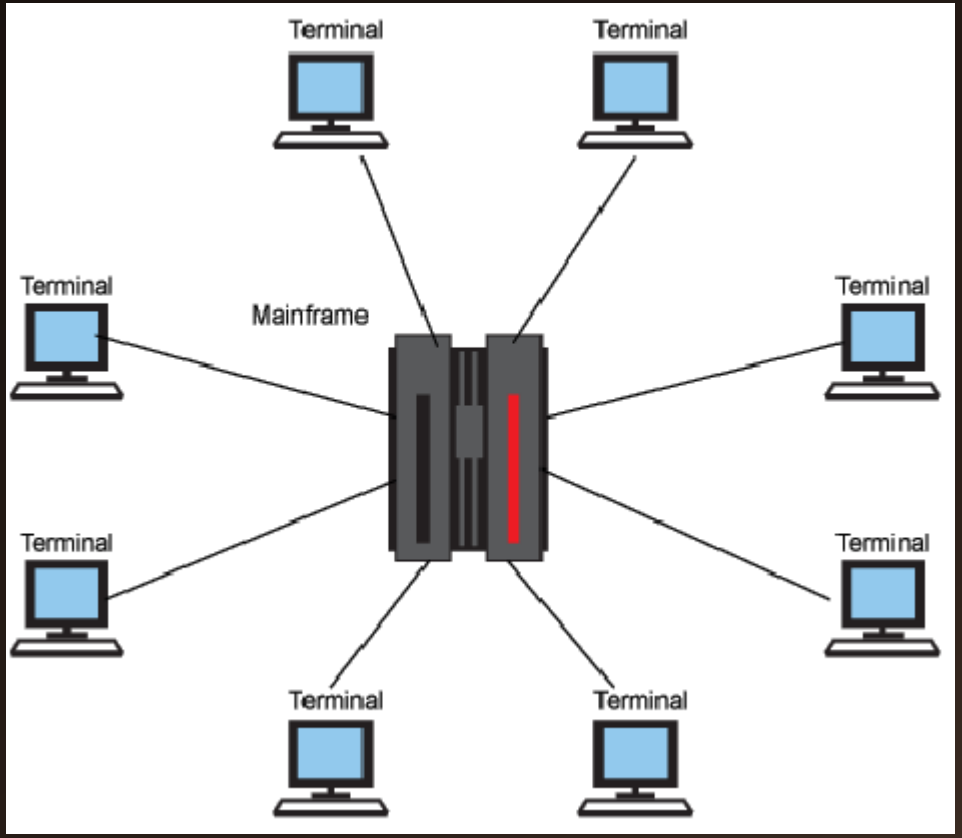
o In this type of operating system, network traffic reduces due to the division between clients and the server.

o This type of system is less expensive to set up and maintain.

Disadvantages of Network Operating System→

o In this type of operating system, the failure of any node in a system affects the whole system.

o Security and performance are important issues. So trained network administrators are required for network administration.



REAL-TIME OPERATING SYSTEM:-

In Real Time Operating Systems, each job carries a certain deadline within which the Job is supposed to be completed, otherwise the huge loss will be there or even if the result is produced then it will be completely useless.

There are two types of Real-Time Operating System→ Hard Real-Time Operating Systems AND Soft Real-Time Operating Systems

Advantages of Real-time operating system:

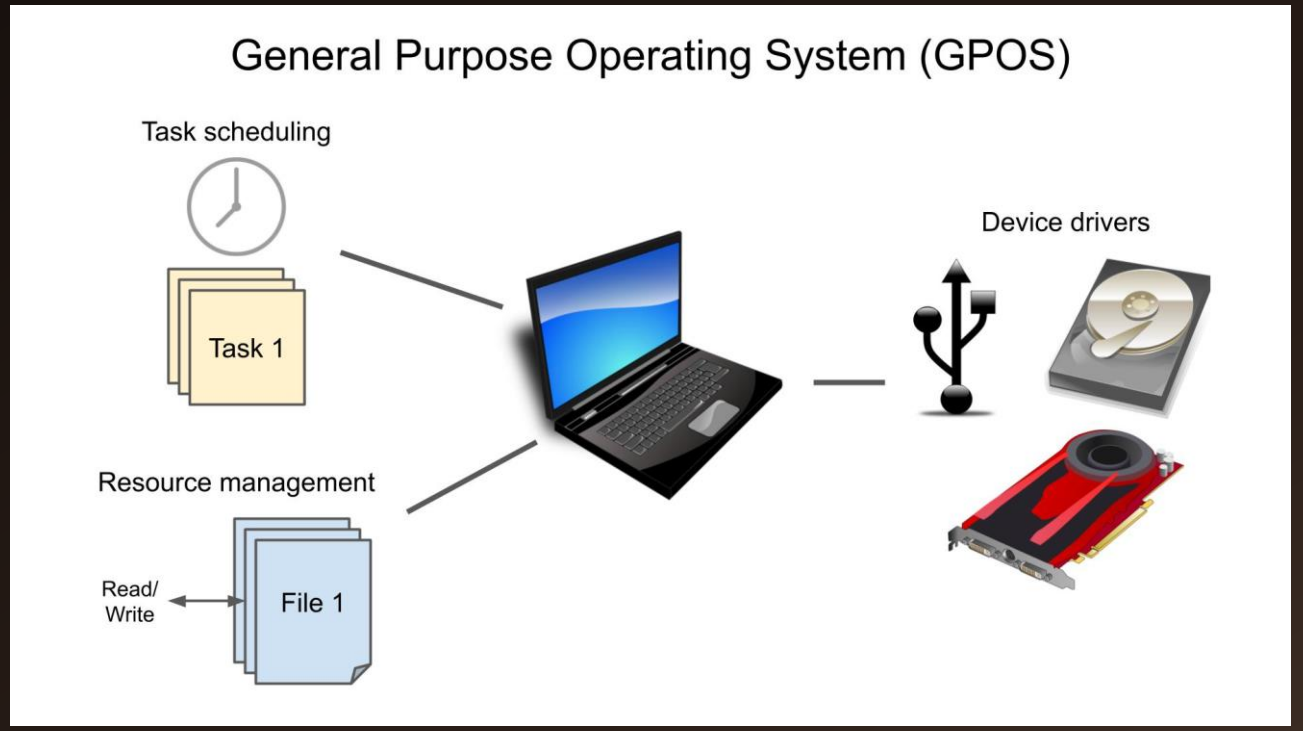
o Easy to layout, develop and execute real-time applications under the real-time operating system.

o In a Real-time operating system, the maximum utilization of devices and systems.

Disadvantages of Real-time operating system:

o Real-time operating systems are very costly to develop.

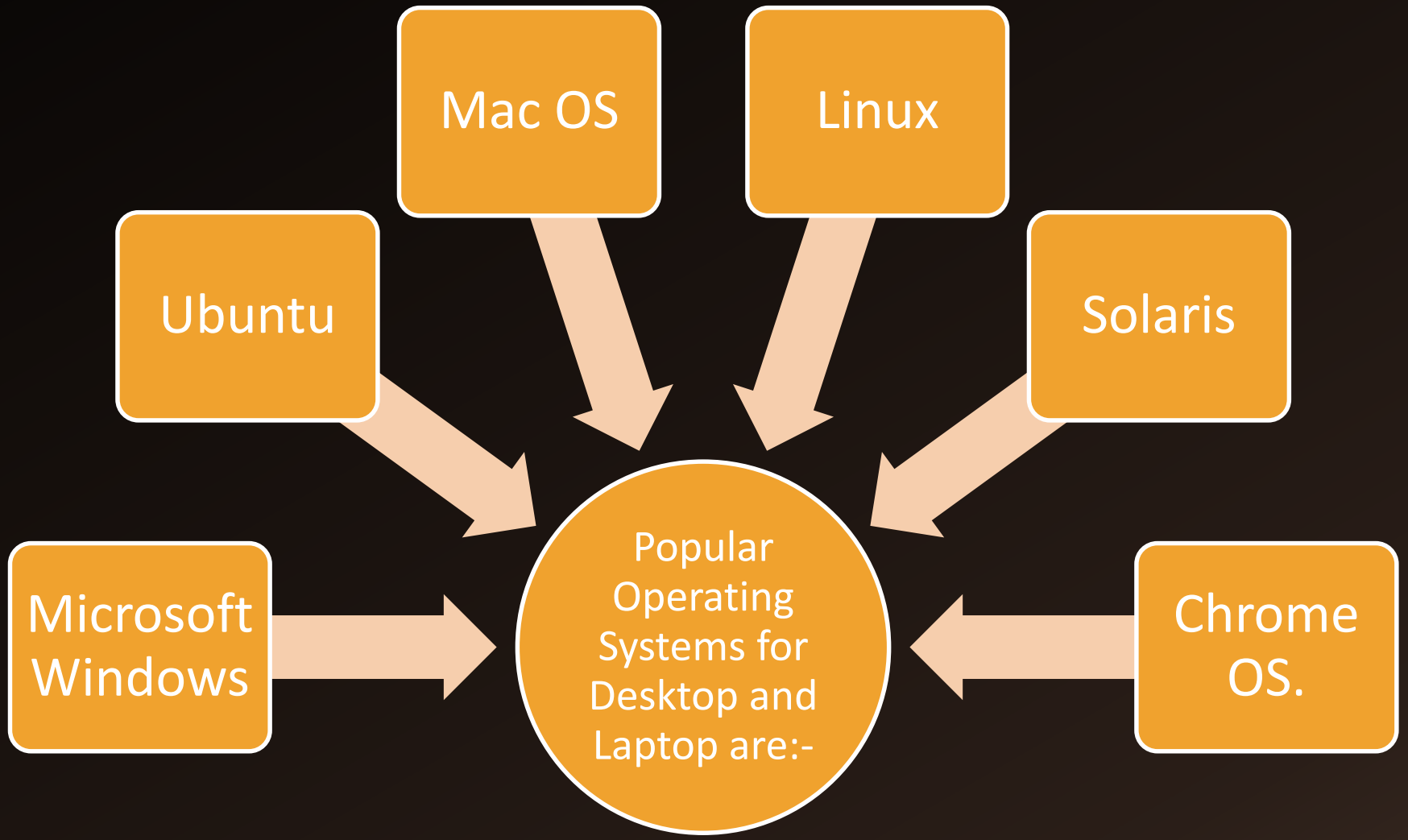
o Real-time operating systems are very complex and can consume critical CPU cycles.



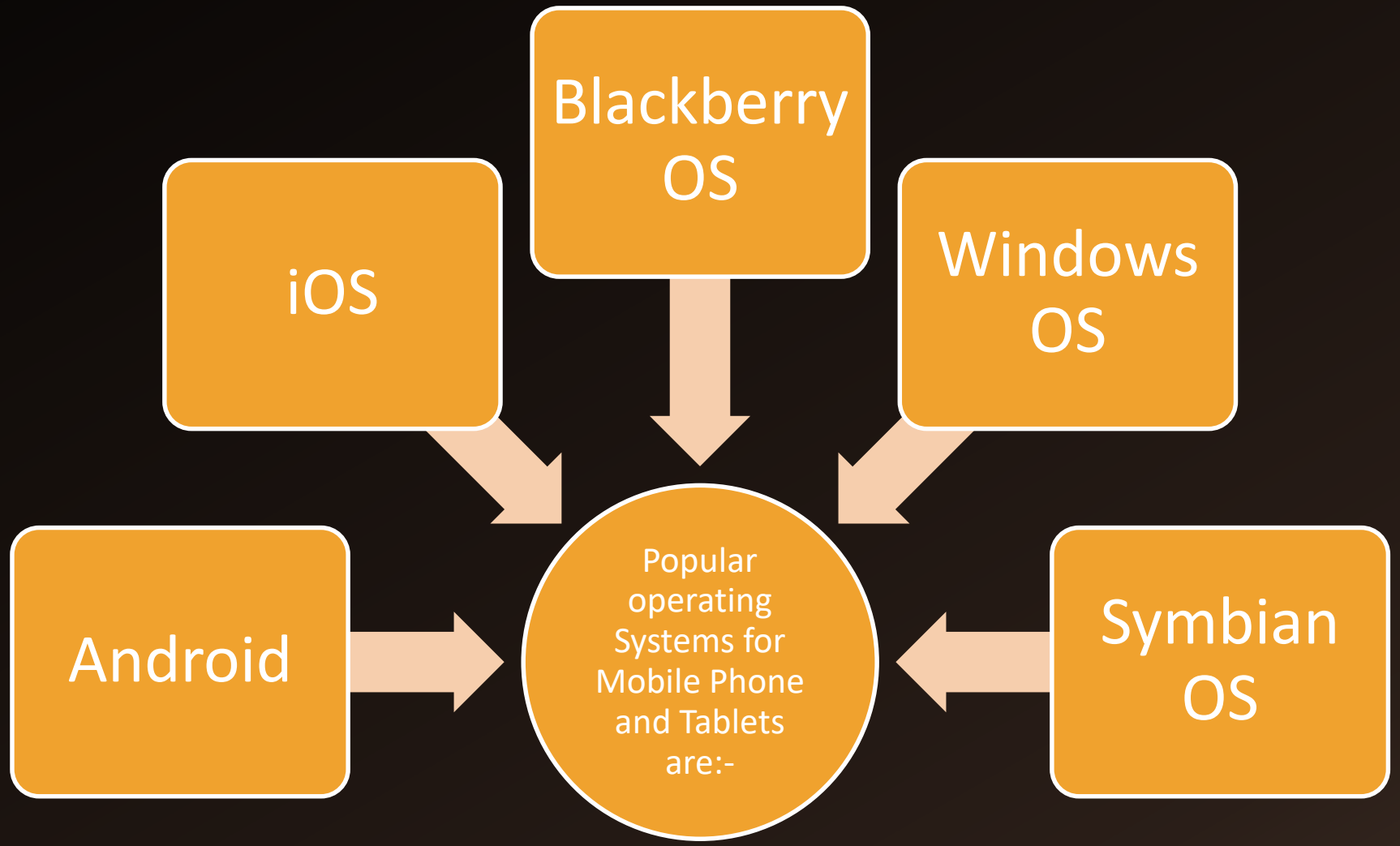
OBJECTIVES OF OPERATING SYSTEM

- OS allocate resources to processes(Manage resources)
- OS Provide an effective user interface
- OS manage the input and output
- OS Provides Graphical User Interface (GUI) in the form of menu, icons, and buttons
- OS manage the interrupts and handle the interrupts etc.

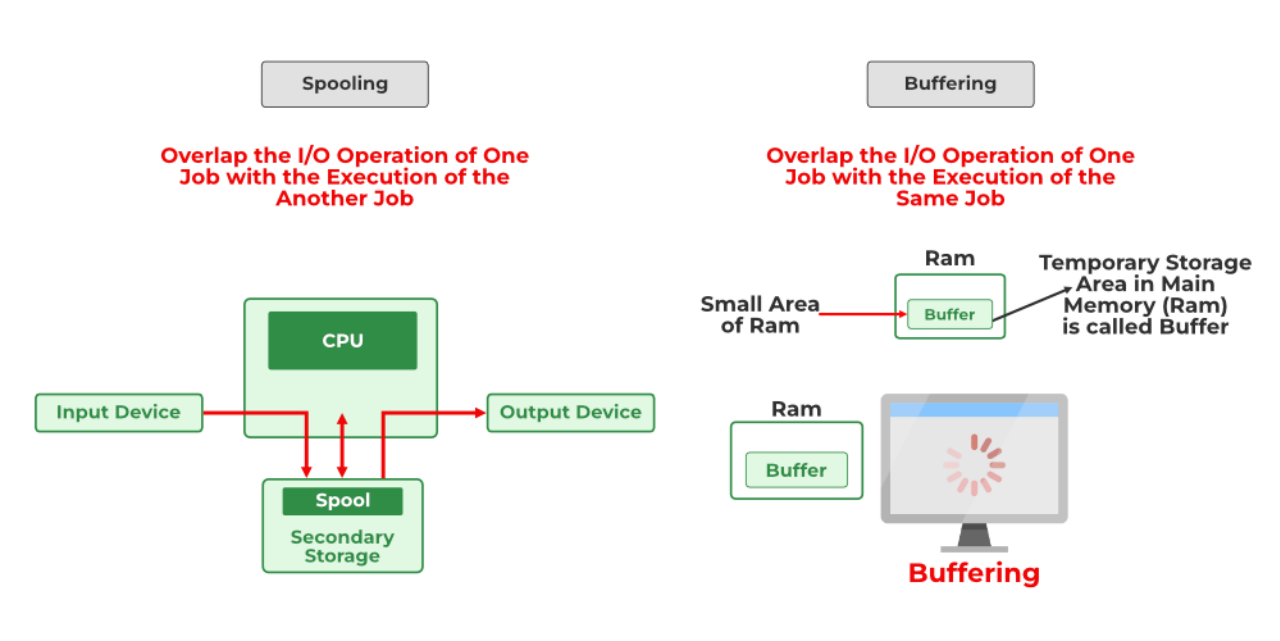
SYSTEMS FOR DESKTOP AND LAPTOP



SYSTEMS FOR DESKTOP AND LAPTOP



SPOOLING IN OPERATING SYSTEM



Spooling is an acronym for simultaneous peripheral operation online. Spooling is the process of temporary storage of data for use and execution by a device, program, or system. Data is sent to and stored in main memory or other volatile storage until it is requested for execution by a program or computer. Spooling makes use of the disc as a large buffer to send data to printers and other devices. It can also be used as an input, but it is more commonly used as an output. Its primary function is to prevent two users from printing on the same page at the same time, resulting in their output being completely mixed together. It prevents this because it uses the FIFO(First In First Out) strategy to retrieve the stored jobs in the spool, and that creates a synchronization preventing the output to be completely mixed together.

It also aids in the reduction of idle time, as well as overlapped I/O and CPU. Simple forms of file management are frequently provided by batch systems. The access to the file is sequential. Batch systems do not necessitate the management of time-critical devices.

How Spooling Works in Operating Systems?

Spooling requires the creation of a buffer known as SPOOL, which is used to hold off jobs and data until the device in which the SPOOL is created is ready to use and execute the job or operate on the data.

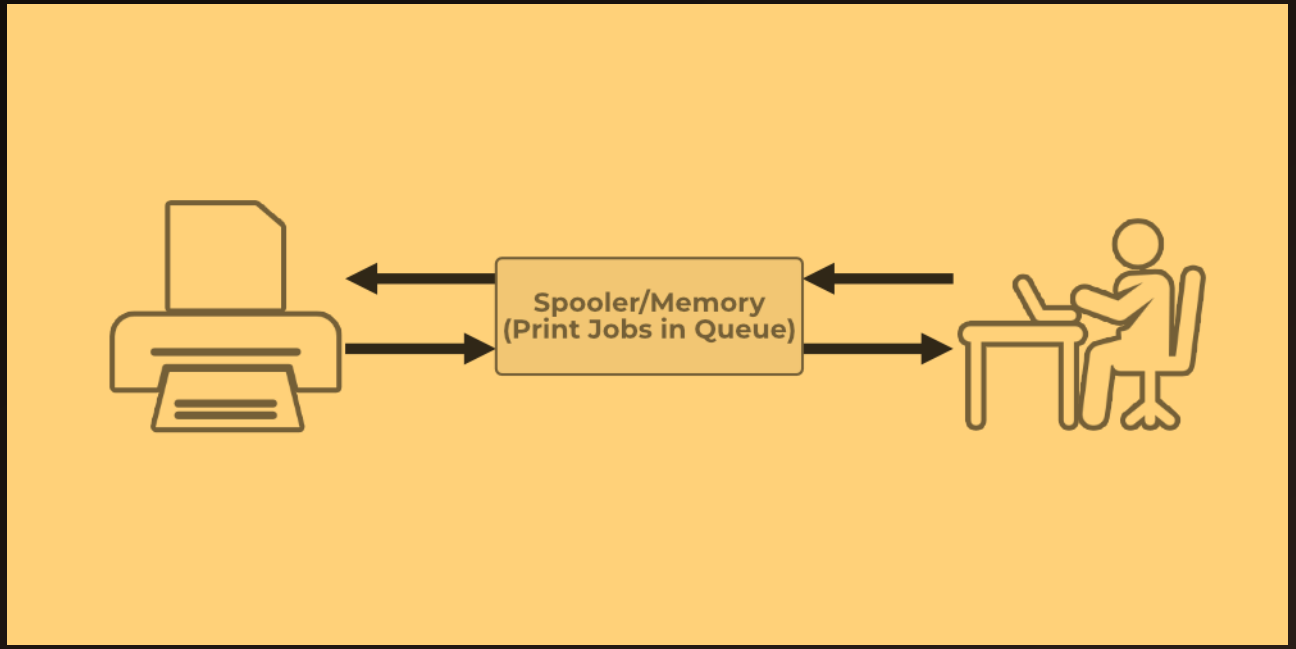
When a faster device sends input to a slower device to perform an operation, it acts as a SPOOL buffer by using any secondary memory attached. This input is retained in the SPOOL until the slower device is ready to use it. When the slower device is ready, the input in the SPOOL is loaded into main memory for the operations that are required.

A device can connect to multiple input devices, each of which may require some data processing. As a result, all of these input devices may store their data in secondary memory (SPOOL), which can then be executed sequentially by the device. This prevents the CPU from becoming idle at any time. As a result, Spooling is a combination of buffering and queuing.

After the CPU generates some output, this output is first saved in the main memory. This output is transferred to the secondary memory from the main memory, and from there, the output is sent to the respective output devices.

EXAMPLE

Printing is the most obvious application of Spooling. The documents to be printed are saved in the SPOOL and then added to the printing queue. During this time, many processes can run and use the CPU without waiting while the printer runs the printing process on each document one by one.



Advantages

The spooling operation makes use of a disc as a very large buffer.	It enables applications to run at the CPU's speed while I/O devices operate at their full speed.	Spooling, on the other hand, is capable of overlapping I/O operations for one job with processor operations for another.

Disadvantages

Depending on the volume of requests received and the number of input devices connected, spooling needs a lot of storage.	Since the SPOOL is created in the secondary storage, having lots of input devices active at once may cause the secondary storage to fill up quickly and increase disc traffic. As a result, the disc becomes slower and slower as the volume of traffic grows.

What is Process? / Process in Operating Systems?

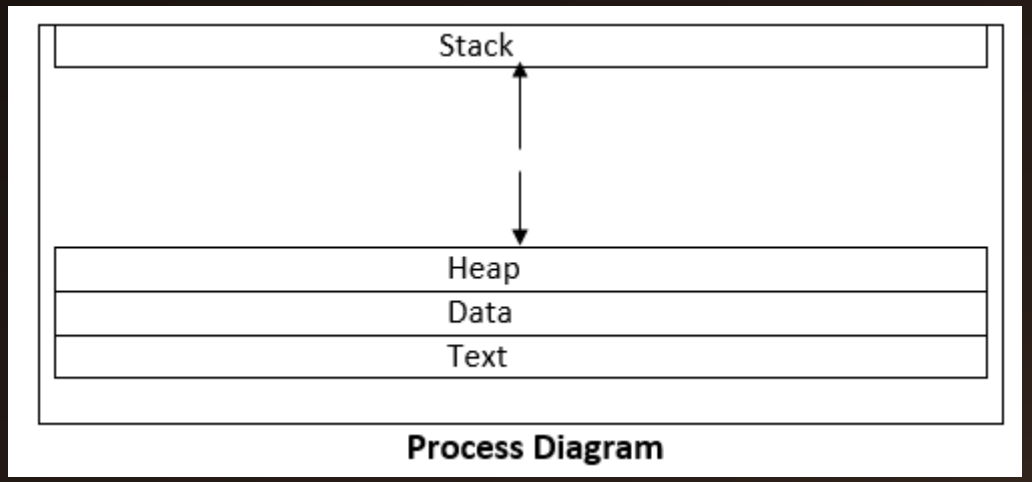
A process is defined as an entity which represents the basic unit of work to be implemented in the system.

Basically, a process is a simple program.

An active program which running now on the Operating System is known as the process. The Process is the base of all computing things. Although process is relatively similar to the computer code but, the method is not the same as computer code. A process is a "active" entity, in contrast to the program, which is sometimes thought of as some sort of "passive" entity. The properties that the process holds include the state of the hardware, the RAM, the CPU, and other attributes.

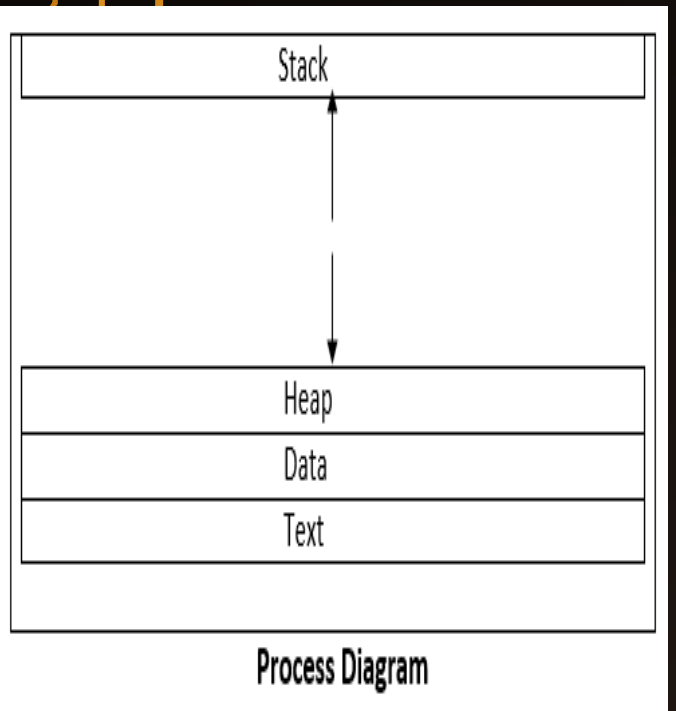
Process in an Operating System

- A process is actively running software or a computer code. Any procedure must be carried out in a precise order. An entity that helps in describing the fundamental work unit that must be implemented in any system is referred to as a process.
- In other words, we create computer programs as text files that, when executed, create processes that carry out all of the tasks listed in the program.
- When a program is loaded into memory, it may be divided into the four components stack, heap, text, and data to form a process. The simplified depiction of a process in the main memory is shown in the diagram below.



Process Diagram

Process Operating System



Process Diagram

- The process Stack contains the temporary data such as method/function parameters, return address and local variables.

- This is dynamically allocated memory to a process during its run time.

Stack

Heap

Data

Text

- This section contains the global and static variables.

- This includes the current activity represented by the value of Program Counter and the contents of the processor's registers.

What are the attributes of process in OS?

- The Attributes of the process are used by the Operating System to create the process control block (PCB) for each of them.
- This is also called context of the process. Attributes which are stored in the PCB.

Process Control Block (PCB)

1. Process ID

- When a process is created, a unique id is assigned to the process which is used for unique identification of the process in the system.

2. Program counter

- A program counter stores the address of the last instruction of the process on which the process was suspended. The CPU uses this address when the execution of this process is resumed.

3. Process State

- The Process, from its creation to the completion, goes through various states which are new, ready, running and waiting. We will discuss about them later in detail.

4. Priority

- Every process has its own priority. The process with the highest priority among the processes gets the CPU first. This is also stored on the **process control block**.

5. General Purpose Registers

- Every process has its own set of registers which are used to hold the data which is generated during the execution of the process.

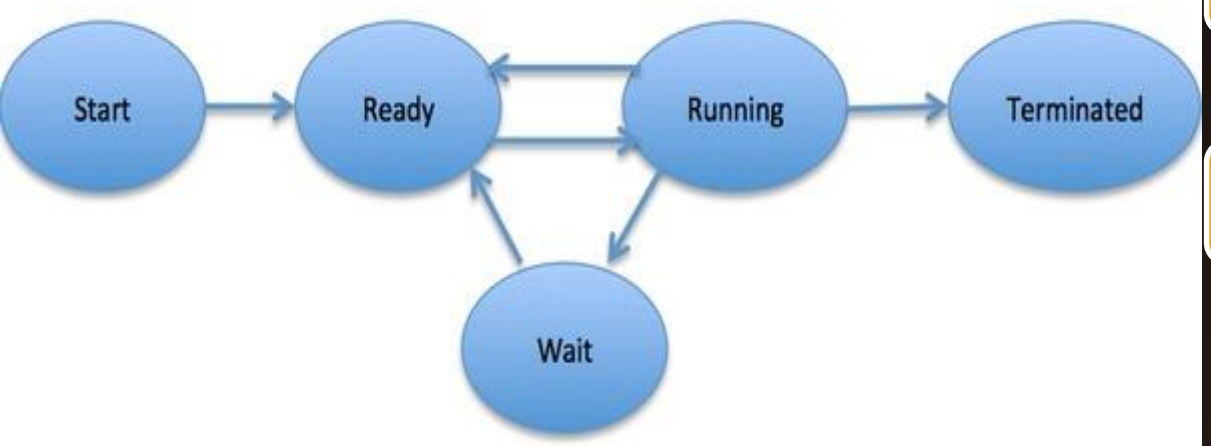
6. List of open files

- During the Execution, Every process uses some files which need to be present in the main memory. OS also maintains a list of open files in the PCB.

7. List of open devices

- OS also maintain the list of all open devices which are used during the execution of the process.

Process Life Cycle



1 Start

- This is the initial state when a process is first started/created.

2 Ready

- The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run. Process may come into this state after **Start** state or while running it by but interrupted by the scheduler to assign CPU to some other process.

3 Running

- Once the process has been assigned to a processor by the OS scheduler, the process state is set to running and the processor executes its instructions.

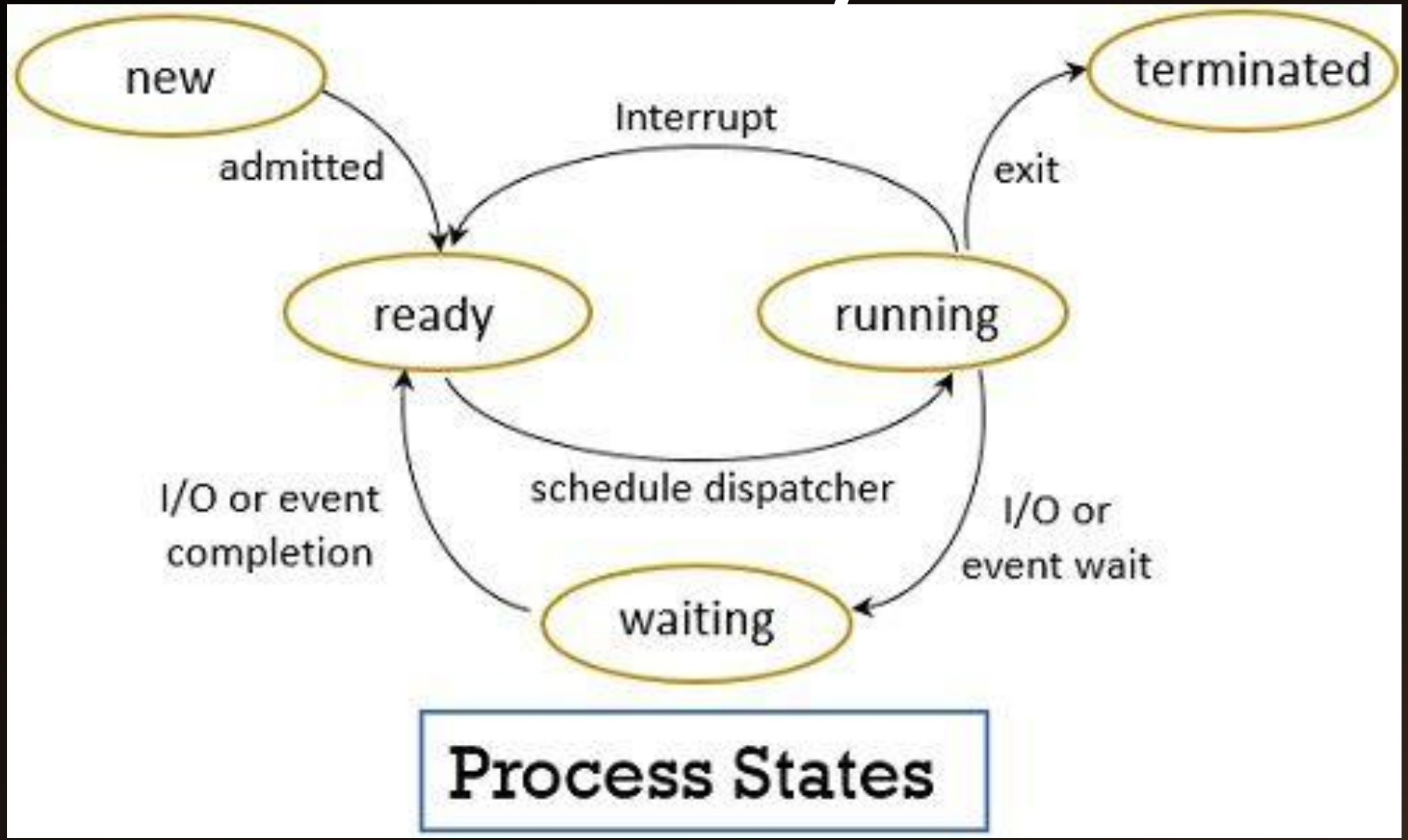
4 Waiting

- Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available.

5 Terminated or Exit

- Once the process finishes its execution, or it is terminated by the operating system, it is moved to the terminated state where it waits to be removed from main memory.

Process Life Cycle



Process Scheduling in OS (Operating System)

Long term scheduler

Long term scheduler is also known as job scheduler. It chooses the processes from the pool (secondary memory) and keeps them in the ready queue maintained in the primary memory. Long Term scheduler mainly controls the degree of Multiprogramming. The purpose of long term scheduler is to choose a perfect mix of IO bound and CPU bound processes among the jobs present in the pool. If the job scheduler chooses more IO bound processes then all of the jobs may reside in the blocked state all the time and the CPU will remain idle most of the time. This will reduce the degree of Multiprogramming. Therefore, the Job of long term scheduler is very critical and may affect the system for a very long time.

2. Short term scheduler

Short term scheduler is also known as CPU scheduler. It selects one of the Jobs from the ready queue and dispatch to the CPU for the execution. A scheduling algorithm is used to select which job is going to be dispatched for the execution. The Job of the short term scheduler can be very critical in the sense that if it selects job whose CPU burst time is very high then all the jobs after that, will have to wait in the ready queue for a very long time. This problem is called starvation which may arise if the short term scheduler makes some mistakes while selecting the job.

3. Medium term scheduler

Medium term scheduler takes care of the swapped out processes. If the running state processes needs some IO time for the completion then there is a need to change its state from running to waiting. Medium term scheduler is used for this purpose. It removes the process from the running state to make room for the other processes. Such processes are the swapped out processes and this procedure is called swapping. The medium term scheduler is responsible for suspending and resuming the processes.

Comparison of Scheduler

Long-Term Scheduler	Short-Term Scheduler	Medium-Term Scheduler
A job scheduler	A CPU scheduler	A process swapping scheduler
Slowest speed	Fastest Speed	Speed is between the other two
Controls the degree of multiprogramming	Provides less control over the degree of multiprogramming	Reduces the degree of multiprogramming
Absent or minimal in the time-sharing os	Minimal in time-sharing OS	Part of time-sharing OS
Selects a process from pool and loads it into memory for execution	Selects a process that is ready for execution	Re-introduces processes into memory for continued execution

CPU Scheduling Process ID in Operating Systems

1. Arrival Time

- The time which is required for the Process to enter the ready queue or the time when the Process is ready to be executed by the CPU.

2. Burst Time

- The Time Slot which the Process requires to complete the Process is known as the Burst Time. The Burst Time can be represented as BT in short form. The Burst Times of a process is always greater than zero.

3. Completion Time

- The Total Time required by the CPU to complete the process is known as Completion Time. The Completion Time can be represented as CT in short form. The Completion will always be greater than zero.

Waiting Time

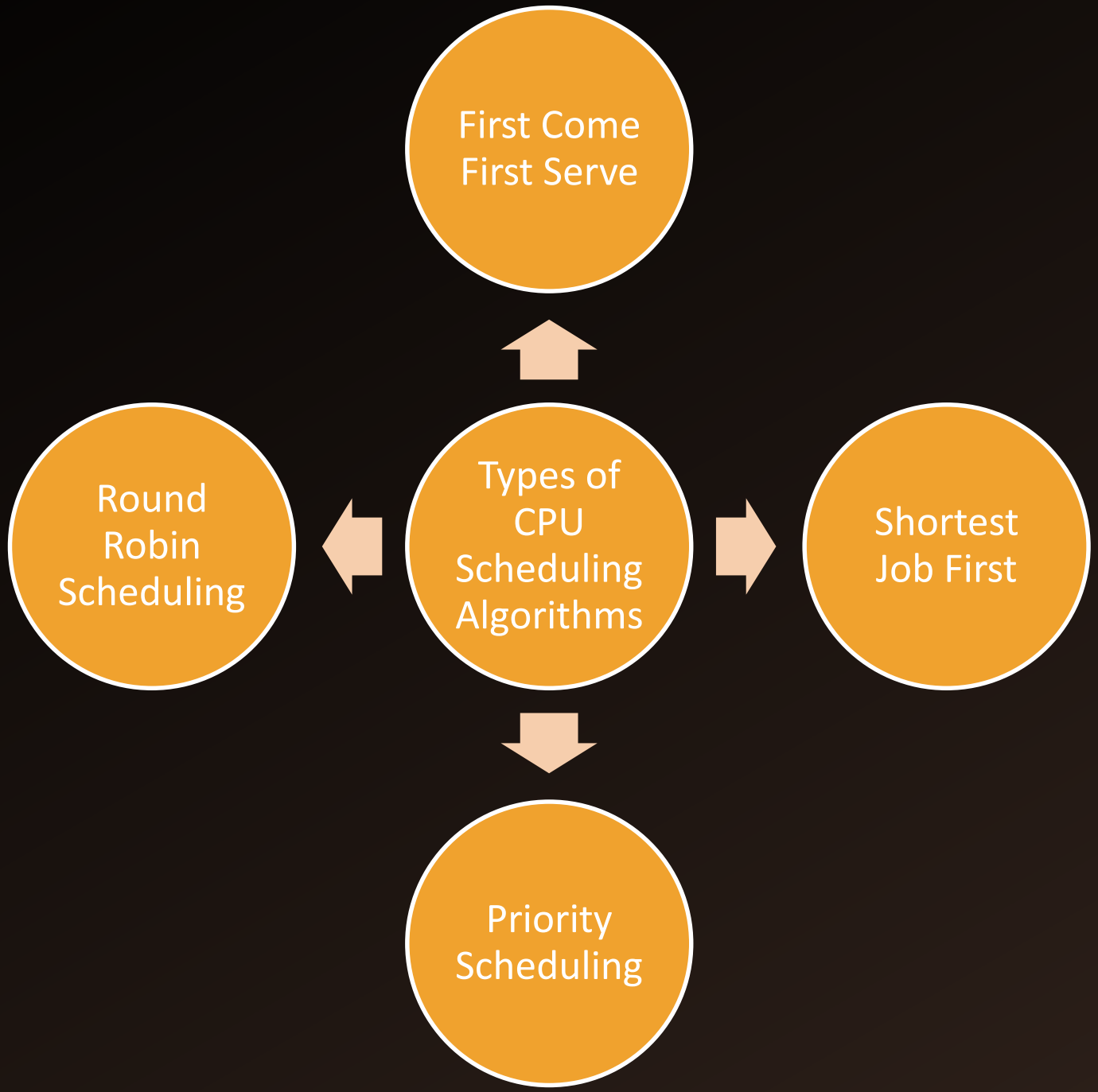
- The time the Process has been waiting to complete its process since the assignment of process for completion is known as Waiting Time. The Waiting Time can be represented as WT in short form. The Waiting Time can be calculated with the help of Turn Around Time and Burst Time.
- WT = TAT - BT

4. Turn Around Time

- The time taken by the CPU since the Process has been ready to execute or since the process is in Ready Queue is known as Turn Around Time. The Turn Around Time can be calculated with the help of Completion Time and Arrival Time. The Turn Around Time can be represented as TAT in short form.
- Wating time + Burst time= Turn around time

Ready Queue

- The Queue where all the processes are stored until the execution of the previous process. This ready queue is very important because there would be confusion in CPU when two same kinds of processes are being executed at the same time.


First Come First Serve
Round Robin Scheduling
Types of CPU Scheduling Algorithms
Shortest Job First
Priority Scheduling

First Come First Serve Scheduling Algorithm

- This is the first type of CPU Scheduling Algorithms. Here, in this CPU Scheduling Algorithm we are going to learn how CPU is going to allot resources to the certain process.
- Here, in the First Come First Serve CPU Scheduling Algorithm, the CPU allots the resources to the process in a certain order. The order is serial way. The CPU is allotted to the process in which it has occurred.
- We can also say that First Come First Serve CPU Scheduling Algorithm follows First In First Out in Ready Queue.
- First Come First Serve can be called as FCFS in short form.

Characteristics of FCFS (First Come First Serve):

- First Come First Serve can follow or can be executed in Pre emptive Approach or Non-Pre emptive Approach
- The Process which enters the Ready Queue is executed First. So, we say that FCFS follows First in First Out Approach.
- First Come First Come First Serve is only executed when the Arrival Time (AT) is greater than or equal to the Time which is at present.

CPU Scheduling Algorithms

- **Advantages**
  - Very easy to perform by the CPU
  - Follows FIFO Queue Approach
- **Disadvantages**
  - First Come First Serve is not very efficient.
  - First Come First Serve suffers because of Convoy Effect.

Non - Preemptive

First Come First Serve Scheduling Algorithm

Process ID	Process Name	Burst Time	Arrival Time	Completion Time (BT1+BTN=CT)	Turn Around Time (T.A.T) CT-AT=TAT	Waiting Time (W.T) TAT-BT=WT
P 1	A	3	0	3	3	0
P 2	B	10	0	13	13	3
P 3	C	5	0	18	18	13
P 4	D	2	0	20	20	18
P 5	E	1	0	21	21	20

Gantt Chart				
P1	P2	P3	P4	P5
0-3	10	5	2	1

Non - Preemptive

Non - Preemptive

First Come First Serve Scheduling Algorithm

Process ID	Process Name	Burst Time	Arrival Time	Completion Time (BT1+BTN=CT)	Turn Around Time (T.A.T) CT-AT=TAT	Waiting Time (W.T) TAT-BT=WT	Response Time First time CPU Allocated-AT
P 1	A	3	1	7	6	3	3
P 2	B	5	3	13	10	5	5
P 3	C	4	0	4	4	0	0
P 4	D	1	2	8	6	5	5

Gantt Chart			
P3	P1	P4	P2
0-4	7	8	13

Total AVERAGE **Turn Around Time** =

Total AVERAGE **Waiting Time** =

Total AVERAGE **Response Time** =

Non - Preemptive

Non - Preemptive

First Come First Serve Scheduling Algorithm

Process ID	Process Name	Burst Time	Arrival Time	Completion Time (BT1+BTN=CT)	Turn Around Time (T.A.T) CT-AT=TAT	Waiting Time (W.T) TAT-BT=WT	Response Time First time CPU Allocated-AT
P 1	A	3	3	10	7	4	4
P 2	B	5	2	7	5	0	0
P 3	C	4	5	15	10	6	6
P 4	D	1	4	11	7	6	6

	Gantt Chart			
	P3	P1	P4	P2
0-2	7	10	11	15

Total AVERAGE **Turn Around Time** =

Total AVERAGE **Waiting Time** =

Total AVERAGE **Response Time** =

Non - Preemptive

Non - Preemptive

Shortest Job First (SJF) Algorithm

Process	Burst Time	Arrival Time	Completion Time (BT1+BTN=CT)	Turn Around Time (T.A.T) CT-AT=TAT	Waiting Time (W.T) TAT-BT=WT	Response Time First time CPU Allocated-AT
P1	1	1	5	4	3	3
P2	2	3	8	5	3	3
P3	1	2	6	4	3	3
P4	4	0	4	4	0	0

	Gantt Chart			
	P3	P1	P4	P2
0	4	5	6	8

Total AVERAGE **Turn Around Time** =

Total AVERAGE **Waiting Time** =

Total AVERAGE **Response Time** =

Non - Preemptive

Preemptive

Shortest Job First (SJF) Algorithm

Process	Burst Time	Arrival Time			
P1	6 ms	2 ms			
P2	2 ms	5 ms			
P3	8 ms	1 ms			
P4	3 ms	0 ms			

	Gantt Chart			
	P3	P1	P4	P2
0-2				

Total AVERAGE **Turn Around Time** =

Total AVERAGE **Waiting Time** =

Total AVERAGE **Response Time** =

Preemptive

Time Instance	Process	Arrival Time	Waiting Table	Execution Time	Initial Burst Time	Remaining Burst Time
0-1ms	P4	0ms		1ms	3ms	2ms
1-2ms	P4	0ms		1ms	2ms	1ms
	P3	1ms	P3	0ms	8ms	8ms
2-3ms	~~P4~~	~~0ms~~		~~1ms~~	~~1ms~~	~~0ms~~
	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3, P1	0ms	6ms	6ms
3-4ms	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3	1ms	6ms	5ms
4-5ms	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3	1ms	5ms	4ms
	P5	4ms	P3, P5	0ms	4ms	4ms

Time Instance	Process	Arrival Time	Waiting Table	Execution Time	Initial Burst Time	Remaining Burst Time
0-1ms	P4	0ms		1ms	3ms	2ms
1-2ms	P4	0ms		1ms	2ms	1ms
	P3	1ms	P3	0ms	8ms	8ms
2-3ms	~~P4~~	~~0ms~~		~~1ms~~	~~1ms~~	~~0ms~~
	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3, P1	0ms	6ms	6ms
3-4ms	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3	1ms	6ms	5ms
4-5ms	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3	1ms	5ms	4ms
	P5	4ms	P3, P5	0ms	4ms	4ms
5-6ms	P3	1ms	P3	0ms	8ms	8ms
	P1	2ms	P3	1ms	4ms	3ms
	P5	4ms	P3, P5	0ms	4ms	4ms
	P2	5ms	P3, P5, P2	0ms	2ms	2ms
6-9ms	P3	1ms	P3	0ms	8ms	8ms
	~~P1~~	~~2ms~~	~~P3~~	~~3ms~~	~~3ms~~	~~0ms~~
	P5	4ms	P3, P5	0ms	4ms	4ms
	P2	5ms	P3, P5, P2	0ms	2ms	2ms
9-11ms	P3	1ms	P3	0ms	8ms	8ms
	P5	4ms	P3, P5	0ms	4ms	4ms
	~~P2~~	~~5ms~~	~~P3, P5~~	~~2ms~~	~~2ms~~	~~0ms~~
11-15ms	P3	1ms	P3	0ms	8ms	8ms
	~~P5~~	~~4ms~~	~~P3~~	~~4ms~~	~~4ms~~	~~0ms~~
15-23ms	~~P3~~	~~1ms~~		~~8ms~~	~~8ms~~	~~0ms~~

Non - Preemptive

Priority Scheduling Algorithm

Total AVERAGE **Turn Around Time** =
Total AVERAGE **Waiting Time** =
Total AVERAGE **Response Time** =

Priority	Process ID	Process Name	Burst Time	Arrival Time	Completion Time (BT1+BTN=CT)	Turn Around Time (T.A.T) CT-AT=TAT	Waiting Time (W.T) TAT-BT=WT	Response Time First time CPU Allocated-AT
1	P 1	A	5	0	5	5	0	0
2	P 2	B	4	1	12	11	7	7
3	P 3	C	2	2	8	6	4	4
4	P 4	D	1	4	6	2	1	1

Gantt Chart			
P1	P4	P3	P2
0-5	6	8	12

Ready Queue Chart		
P2	P3	P4

Non - Preemptive

Non - Preemptive

Priority Scheduling Algorithm

Total AVERAGE **Turn Around Time** =
Total AVERAGE **Waiting Time** =
Total AVERAGE **Response Time** =

Priority	Process ID	Process Name	Burst Time	Arrival Time	Completion Time (BT1+BTN=CT)	Turn Around Time (T.A.T) CT-AT=TAT	Waiting Time (W.T) TAT-BT=WT	Response Time First time CPU Allocated-AT
1	P 1	A	5	0				
3	P 2	B	4	1				
2	P 3	C	2	2				
4	P 4	D	1	4				

Gantt Chart			
P1	P4	P3	P2

Ready Queue Chart		
P2	P3	P4

Non - Preemptive

Thanks