Operating Systems Development (A Historical perspective)

Operating systems have developed through a number of phases (generations) and are basically connected to the generations of computer hardware and software. Operating systems are very close to the computer architecture.

Following are some of important functions of an operating System.

- Processor Management
- Memory Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

Memory Management

Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory. An Operating System does the following activities for memory management −

- Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- In multiprogramming, the OS decides which process will get memory when and how much.
- Allocates the memory when a process requests it to do so.
- De-allocates the memory when a process no longer needs it or has been terminated.

Processor Management

The OS decides which process gets the processor when and for how much time. This function is called **process scheduling**. An Operating System does the following activities for processor management

- Keeps tracks of processor and status of process. The program responsible for this task is known as **traffic controller**.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

Device Management

An Operating System manages device communication via their respective drivers. It does the following activities for device management −

- Keeps tracks of all devices. Program responsible for this task is known as the **I/O controller**.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

File Management

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.

An Operating System does the following activities for file management −

- Keeps track of information, location, uses, status etc. The collective facilities are often known as **file system**.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources.

Following are the major activities of an operating system with respect to program management −

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

I/O Operation

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

An Operating System manages the communication between user and device drivers.

- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

Communication

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

The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication −

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

Error handling

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

- The OS constantly checks for possible errors.
- The OS takes an appropriate action to ensure correct and consistent computing.

Protection

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

Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection −

- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.

Other Important Activities

Following are some of the important activities that an Operating System performs −

- **Security** − By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- **Control over system performance** − Recording delays between request for a service and response from the system.
- **Job accounting** − Keeping track of time and resources used by various jobs and users.
- **Error detecting aids** − Production of dumps, traces, error messages, and other debugging and error detecting aids.
- **Coordination between other softwares and users** − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

Types of Operating System

Operating systems are there from the very first computer generation and they keep evolving with time. In this chapter, we will discuss some of the important types of operating systems which are most commonly used.

Batch operating system

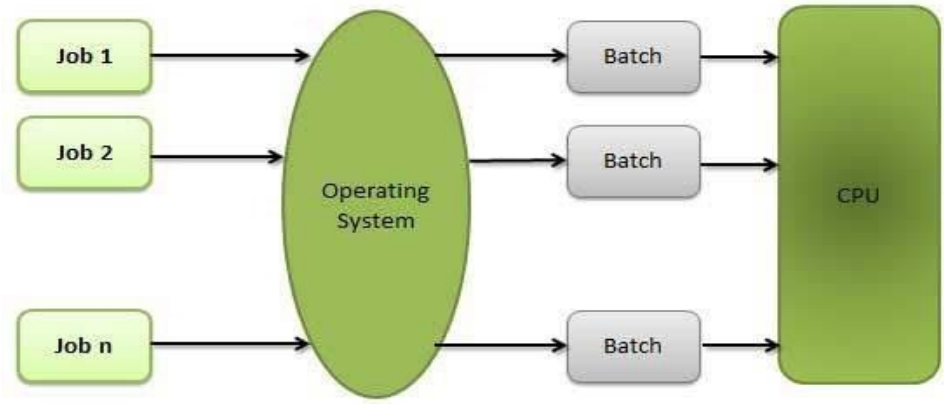
The users of a batch operating system do not interact with the computer directly. Each user prepares his job on an off-line device like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group. The programmers leave their programs with the operator and the operator then sorts the programs with similar requirements into batches.

The problems with Batch Systems are as follows −

- Lack of interaction between the user and the job.
- CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.
- Difficult to provide the desired priority.

Batch processing is a technique in which an Operating System collects the programs and data together in a batch before processing starts. An operating system does the following activities related to batch processing −

- The OS defines a job which has predefined sequence of commands, programs and data as a single unit.
- The OS keeps a number a jobs in memory and executes them without any manual information.
- Jobs are processed in the order of submission, i.e., first come first served fashion.
- When a job completes its execution, its memory is released and the output for the job gets copied into an output spool for later printing or processing.



Advantages

- Batch processing takes much of the work of the operator to the computer.
- Increased performance as a new job get started as soon as the previous job is finished, without any manual intervention.

Disadvantages

- Difficult to debug program.
- A job could enter an infinite loop.
- Due to lack of protection scheme, one batch job can affect pending jobs.

Time-sharing operating systems

Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing.

The main difference between Multiprogrammed Batch Systems and Time-Sharing Systems is that in case of Multiprogrammed batch systems, the objective is to maximize processor use, whereas in Time-Sharing Systems, the objective is to minimize response time.

Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, the processor executes each user program in a short burst or quantum of computation. That is, if users are present, then each user can get a time quantum. When the user submits the command, the response time is in few seconds at most.

The operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems have been modified to time-sharing systems.

Advantages of Timesharing operating systems are as follows −

- Provides the advantage of quick response.
- Avoids duplication of software.
- Reduces CPU idle time.

Disadvantages of Time-sharing operating systems are as follows −

- Problem of reliability.
- Question of security and integrity of user programs and data.
- Problem of data communication.

Distributed operating System

Distributed systems use multiple central processors to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly.

The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as **loosely coupled systems** or distributed systems. Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers, and so on.

The advantages of distributed systems are as follows −

- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speedup the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

Network operating System

A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks.

Examples of network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

The advantages of network operating systems are as follows −

- Centralized servers are highly stable.
- Security is server managed.
- Upgrades to new technologies and hardware can be easily integrated into the system.
- Remote access to servers is possible from different locations and types of systems.

The disadvantages of network operating systems are as follows −

- High cost of buying and running a server.
- Dependency on a central location for most operations.
- Regular maintenance and updates are required.

Real Time operating System

A real-time system is defined as a data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. The time taken by the system to respond to an input and display of required updated information is termed as the **response time**. So in this method, the response time is very less as compared to online processing.

Real-time systems are used when there are rigid time requirements on the operation of a processor or the flow of data and real-time systems can be used as a control device in a dedicated application. A real-time operating system must have well-defined, fixed time constraints, otherwise the system will fail. For example, Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

There are two types of real-time operating systems.

Hard real-time systems

- Hard real-time systems guarantee that critical tasks complete on time. In hard real-time systems, secondary storage is limited or missing and the data is stored in ROM. In these systems, virtual memory is almost never found.

Soft real-time systems

- Soft real-time systems are less restrictive. A critical real-time task gets priority over other tasks and retains the priority until it completes. Soft real-time systems have limited utility than hard real-time systems. For example, multimedia, virtual reality, Advanced Scientific Projects like undersea exploration and planetary rovers, etc.

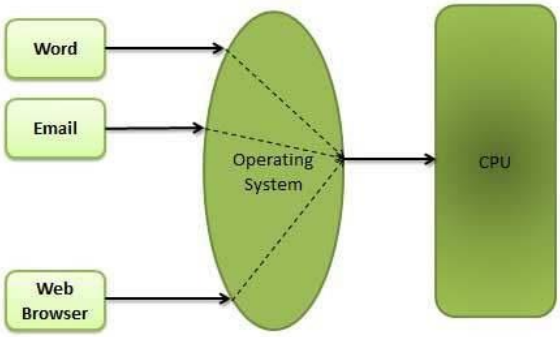
Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use).

Multitasking

Multitasking is when multiple jobs are executed by the CPU simultaneously by switching between them. Switches occur so frequently that the users may interact with each program while it is running. An OS does the following activities related to multitasking −

- The user gives instructions to the operating system or to a program directly, and receives an immediate response.
- The OS handles multitasking in the way that it can handle multiple operations/executes multiple programs at a time.
- Multitasking Operating Systems are also known as Time-sharing systems.
- These Operating Systems were developed to provide interactive use of a computer system at a reasonable cost.
- A time-shared operating system uses the concept of CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared CPU.
- Each user has at least one separate program in memory.



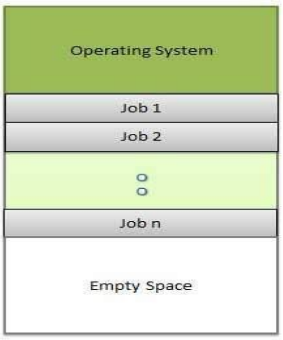
- A program that is loaded into memory and is executing is commonly referred to as a **process**.
- When a process executes, it typically executes for only a very short time before it either finishes or needs to perform I/O.
- Since interactive I/O typically runs at slower speeds, it may take a long time to complete. During this time, a CPU can be utilized by another process.

- The operating system allows the users to share the computer simultaneously. Since each action or command in a time-shared system tends to be short, only a little CPU time is needed for each user.
- As the system switches CPU rapidly from one user/program to the next, each user is given the impression that he/she has his/her own CPU, whereas actually one CPU is being shared among many users.

Multiprogramming

Sharing the processor, when two or more programs reside in memory at the same time, is referred as **multiprogramming**. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

The following figure shows the memory layout for a multiprogramming system.



An OS does the following activities related to multiprogramming.

- The operating system keeps several jobs in memory at a time.
- This set of jobs is a subset of the jobs kept in the job pool.
- The operating system picks and begins to execute one of the jobs in the memory.
- Multiprogramming operating systems monitor the state of all active programs and system resources using memory management programs to ensures that the CPU is never idle, unless there are no jobs to process.

Advantages

- High and efficient CPU utilization.
- User feels that many programs are allotted CPU almost simultaneously.

Disadvantages

- CPU scheduling is required.
- To accommodate many jobs in memory, memory management is required.

Interactivity

Interactivity refers to the ability of users to interact with a computer system. An Operating system does the following activities related to interactivity −

- Provides the user an interface to interact with the system.
- Manages input devices to take inputs from the user. For example, keyboard.
- Manages output devices to show outputs to the user. For example, Monitor.

The response time of the OS needs to be short, since the user submits and waits for the result.

Spooling

Spooling is an acronym for simultaneous peripheral operations on line. 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 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 spooling process as a computer can perform I/O in parallel fashion. It becomes possible to have the computer read data from a tape, write data to disk and to write out to a tape printer while it is doing its computing task.

Advantages

- The spooling operation uses a disk as a very large buffer.
- Spooling is capable of overlapping I/O operation for one job with processor operations for another job.

OPERATING SYSTEMS DEVELOPMENT

First generation (1945 –1955) – Vacuum tubes and plug boards

The first generation computers were basically mechanical devices, there was only computer hardware e.g. ENIAC. These computers were programmed manually by setting switches, plugging and unplugging cables. You had to interact with the hardware directly. The operations of these machines were very slow, cumbersome and tedious.

A method called the 'stored program' concept was developed to enhance their operations. This was made possible by Von Neumann in 1949 and the computers that resulted from this were known as Von Neumann machines. This led to the realization of digital computers and the beginning of the programming concept. Some computers that made use of this concept were: EDVAC, EDSAC and the IAS machines. These computers were made to have all features of modern computers i.e. CPU(ALU, Registers), memory and I/O facilities.

Programs were written in machine language and programming done by one individual at a time. These computers would have a sign-up sheet, which allocated every user a block time and exclusive access at any time. Preparing a machine language program for execution involved:

- A programmer would write a program and operate it directly from the console
- Programs and data would be manually loaded into memory, one instruction at a time
- The appropriate buttons would then be pushed to set the starting address and start the execution of the program
- If errors occurred, the programmer would halt the program and examine the contents of memory and registers and debug the program directly from the console

The situation changed in the 1950's:

- Assembly (symbolic) languages were developed
- Punched cards were used to input programs and data into the computer using a punched card reader.
- Assemblers were developed to compile assembly language to machine language

Problems:

- There was a significant setup time (time taken to load, assemble and execute programs)
- The CPU sat idle for lengths of time, under utilizing the resources and abilities of the computer, that is, during I/O transfers, setup times, control being transferred to another program etc

Second Generation Computers (1955 – 1965) – Transistors and Batch Systems

Characteristics of Computers and software at this time included:

- They used transistors instead of vacuum tubes.
- The computers became more reliable and grew smaller in size.
- Computers became more commercialized and were sold to businesses for data processing.
- Machine independent languages like COBOL, FORTRAN and ALGOL were being developed.
- Additional system software like compilers, utilities for tape/card conversion and batch monitors were made available
- The computers were being used for both scientific and business data processing

The problems with the computers at this time included:

- The CPU sat idle for some time due to time reserved and not being used and slow I/O data transfers. This resulted into low utilization of the computer facilities and because computers were being used commercially, it prompted scientists to develop mechanisms for high utilization of the computer. For example, sharing of CPU by many users
- There was an I/O and computer speed disparity. The computers had become faster than the I/O devices.

Solutions developed to address the problem include:

1. The use of *professional operators* to operate the computers. Programmers normally wrote programs and were not allowed to run them, rather they presented them to the professional operators
2. The jobs were usually batched and the operator would run the programs one by one.
3. Jobs with similar needs would be *batched* together to use the available setups common to them. When the computer is available the batch is run as a group with intermediate setups. This was the basis of job scheduling, but the operator performed the job transition manually.
4. A rudimentary kind of operating system called a *batch monitor* was developed to reduce the human setup time. It automated job sequencing. The monitor would be invoked and transfer control to the first program, the program would then execute and return control to the monitor. The monitor would then invoke the second program and transfer control to it.
5. *Off-line Spooling* was introduced to supplement batch processing. Off-line spooling uses high-speed I/O devices. It was decided that tapes rather than punched cards could be used for input of programs. A number of jobs were collected the copied to a tape on an off-line computer. The tape was rewound then mounted to the main computer by an operator and job executed one by one delivering output to a tape.
6. *Multiple buffering* was introduced. This is overlapping CPU and I/O operations. This method builds blocks of data into memory before output and also builds blocks of data before input to memory. When a program requested for data or records, the data is read from tape to memory and then from memory to the CPU, while the CPU was processing, another block of data could be read to the second empty buffer of memory simultaneously.

Third Generation Computers (1965 – 1980) – ICs and multiprogramming

IC technology replaced transistor technology. This led to a great reduction in the cost of computers, size, efficiency, speed and reliability. Semi-conductor (IC) memory became common. Microprogramming came to a wide spread use to simplify CPU design. Concurrent programming techniques like multiprogramming and time-sharing were introduced for sharing computer CPU. This time also witnessed a greater increase in the use of magnetic disks

The problems during this time included:

- There were I/O and CPU speed disparities. The time the CPU sat idle was increased because of the change in technology for developing processors, which was relatively faster compared to the change in I/O technology by using magnetic disks.
- Poor utilization of the CPU, which had become even faster resulting in high *turnaround times* and low *throughput*.

Solution to these problems:

- Introduction of *multiprogramming*. Memory was divided into several partitions and each job was given a partition in memory. A job would then be given the CPU for execution; at one point the job would perform an I/O operation. When this happened, the CPU would switch to another job until it also needed the services on an I/O device. The CPU would keep on searching for jobs that are ready to execute. This resulted in improvements in the turnaround times and increased throughput.
- *Online Spooling* became a common phenomenon as it was easier to read data from disk when requested for unlike the tape, which had to be rewound or forwarded to retrieve data.
- The desire for applications requiring quick response paved way for *time-sharing*. In time-sharing, the computer operates in an interactive mode with a number of logged in users. Time-sharing systems use CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer called a time slice
- Real time systems also became popular.

Fourth Generation Computers (1980 …) – VLSI and Personal Computers

VLSI (Very Large Scale Integration) led to the possibility of manufacturing a whole component in a single IC chip. This made the computers at this time to be fast, reliable, cheap and efficient. This led to:

- Personal Computers
- Parallel Processing systems
- Computer Networks

1. **Personal Computers -** Large software industries were developed to manufacture the required software like single user O/S for use in personal computers. Initially, they were only single user O/S but nowadays we have multitasking operating systems like Windows '95 where a user runs several tasks concurrently. These operating systems have also opted for user convenience and responsiveness through better user interface (Graphical User Interface).
2. **Parallel Processing Systems –** Parallel systems make use of more than one processor to perform tasks. Operating systems for parallel systems were developed to efficiently exploit the underlying available hardware.
3. **Computer Networks** – Independent computers were interconnected by a network to allow users share the available resources like printers, servers and hard disks. They could and exchange data through some communication medium. Network operating systems were developed to manage computers in a network.

CAT 1

1. Briefly explain the concept of swapping and relocation in memory management in operating systems.

   Swapping and relocation are concepts in memory management in operating systems.

   Swapping refers to the process of temporarily transferring parts of the memory contents to a disk, called the swap space or swap file, in order to free up main memory (RAM) for other processes. When a process that was swapped out to disk needs to be executed again, it is swapped back into memory. Swapping is used by the operating system to improve the overall performance of the system by allowing more processes to run simultaneously.

   Relocation refers to the process of assigning a base address to a program or process in memory, which is different from the address it was compiled with. This allows multiple programs or processes to be loaded into memory and executed at the same time without interfering with each other. The operating system uses relocation to avoid conflicts between programs or processes that use the same memory addresses.

Both swapping and relocation are important concepts in memory management in operating systems and are used to improve the performance and stability of the system by efficiently utilizing the available memory resources.

[5 MKS]

2. Explain why CentOS is safe and not affected by viruses? [5 MKS]

- It uses stable and thoroughly tested software - CentOS releases security updates regularly to fix any vulnerabilities that may have been discovered. This helps to keep the operating system secure and prevent viruses and other types of malware from compromising the system.
  -Has a large and active community- CentOS has a large and active community of developers and users who collaborate to identify and fix security issues
- Security-focused design: CentOS is built from the Red Hat Enterprise Linux (RHEL) source code, which is known for its focus on security and stability. This means that CentOS inherits the security features and best practices from RHEL, making it a secure operating system.
- Regular security updates: CentOS releases security updates regularly to fix any vulnerabilities that may have been discovered. This helps to keep the operating system secure and prevent viruses and other types of malware from compromising the system

3. Consider performance of each of the three algorithms, FCFS, SJF and RR for three computer-bound processes. What if have 3 processes P1 (takes 10 seconds), P2 (takes 4 seconds) and P3 (takes 6 seconds). If arrive in order P2, P3, P1, what is
   [10 Mks]
   a. Waiting Time?
   b. Turnaround Time?
   c. Throughput?

   What about if processes come in order P2, P3, P1 (Assume all processes arrive at the same time here)? What is
   d. Waiting Time?
   e. Turnaround Time?
   f. Throughput?

- Ans

  a. Waiting time in FCFS:
- For P1: waiting time is 0 seconds, since it is the first process to arrive.
- For P2: waiting time is 10 seconds, since it has to wait for P1 to finish before it starts.
- For P3: waiting time is 14 seconds, since it has to wait for both P1 and P2 to finish before it starts.
- b. Turnaround time in FCFS:
- For P1: turnaround time is 10 seconds, which is equal to the total time it took for the process to complete.
- For P2: turnaround time is 14 seconds, which is the sum of the waiting time (10 seconds) and the time it took to complete (4 seconds).
- For P3: turnaround time is 20 seconds, which is the sum of the waiting time (14 seconds) and the time it took to complete (6 seconds).
- c. Throughput in FCFS:

- Throughput is the number of processes completed per unit time. In this case, the total time taken for all three processes to complete is 20 seconds, so the throughput is 3/20 = 0.15 processes per second.
- d. Waiting time in SJF:
- For P2: waiting time is 0 seconds, since it is the shortest job and it starts first.
- For P3: waiting time is 4 seconds, since it has to wait for P2 to finish before it starts.
- For P1: waiting time is 10 seconds, since it has to wait for both P2 and P3 to finish before it starts.
- e. Turnaround time in SJF:
- For P2: turnaround time is 4 seconds, which is equal to the total time it took for the process to complete.
- For P3: turnaround time is 10 seconds, which is the sum of the waiting time (4 seconds) and the time it took to complete (6 seconds).
- For P1: turnaround time is 20 seconds, which is the sum of the waiting time (10 seconds) and the time it took to complete (10 seconds).
- f. Throughput in SJF:
- Throughput is the number of processes completed per unit time. In this case, the total time taken for all three processes to complete is 20 seconds, so the throughput is 3/20 = 0.15 processes per second.
- d. Waiting time in RR:
- For P2: waiting time is 0 seconds, since it is the first process to arrive.
- For P3: waiting time is 2 seconds, since it has to wait for 2 seconds for its turn in the queue.
- For P1: waiting time is 8 seconds, since it has to wait for 2 seconds for its turn in the queue, and then wait for P2 and P3 to finish.
- e. Turnaround time in RR:
- For P2: turnaround time is 4 seconds, which is equal to the total time it took for the process to complete.
- For P3: turnaround time is 8 seconds, which is the sum of the waiting time (2 seconds) and the time it took to complete (6 seconds).
- For P1: turnaround time is 18 seconds, which is the sum of the waiting time (8 seconds) and the time it took to complete (10 seconds).
- f. Throughput in RR:
- Throughput is the number of processes completed per unit time. In this case, the total time taken for all three processes to complete is 20 seconds, so the throughput is 3/20 = 0.
-