Assignement 1A:

1. Commands description
2. Syntax of all commands
3. Options related to commands(atleast 3)
4. echo, ls, read, cat, touch, test, loops, arithmetic comparison, conditional loops, grep, sed etc.

1.echo command in linux is used to display line of text/string that are passed as an argument .
This is a built in command that is mostly used in shell scripts and batch files to output status text to the screen or a file.
Syntax :

echo [option] [string]

-e here enables the interpretation of backslash escapes
echo -e "Geeks \bfor \bGeeks"

-n : this option is used to omit echoing trailing newline

2.The ls is the list command in Linux. It will show the full list or content of your directory
ls
ls -a : Hidden files
ls - l: files in longlist format
ls -r: print list in reverse order

3.The Linux read command is used to read the contents of a line into a variable.
This is a built-in command for Linux systems. Therefore, we do not need to install any additional tools. I
t is an easy tool to take user input when creating a bash script.
It is a powerful utility and as important as echo command and positional parameter.

read [options] [name...]

The '-p' option is used for the prompt text. It reads the data along with some hint text.
The '-n' option limits the length of the character in the entered text. It will not let
you enter text more than the specified number of characters.

The '-s' option is used for security purpose. It is used to read the sensitive data. By using this option, the entered text won't appear in the terminal.

4.Cat
It can be used to display the content of a file, copy content from one file to another, concatenate the contents of multiple files, display the line number, display $ at the end of the line, etc.

Syntax:

cat

6.A test command is a command that is used to test the validity of a command. It checks whether the command/expression is true or false. It is used to check the type of file and the permissions related to a file.

Syntax:

test [expression]

test -e filename: Checks whether the file exists or not. And return 0 if file exists and returns 1 if file doesnot exits.
test -d filename: Checks whether the file is a directory or not. And returns 0 if the file is a directory and returns 1 if the file is not a directory.
test -f filename: Checks whether the file is a regular file or not. And returns 0 if the file is a regular file and returns 1 if the file is not a regular file.

7.The for loop operates on lists of items. It repeats a set of commands for every item in a list.

Syntax
for var in word1 word2 ... wordN
do
  Statement(s) to be executed for every word.
done

8.Assume variable a holds 10 and variable b holds 20 then −

Operator Description Example
+ (Addition) Adds values on either side of the operator ` expr $a + $b` will give 30
- (Subtraction) Subtracts right hand operand from left hand operand `expr $a - $b` will give -10
* (Multiplication) Multiplies values on either side of the operator `expr $a \* $b` will give 200
/ (Division) Divides left hand operand by right hand operand `expr $b / $a` will give 2
% (Modulus) Divides left hand operand by right hand operand and returns remainder `expr $b % $a` will give 0
= (Assignment) Assigns right operand in left operand a = $b would assign value of b into a
== (Equality) Compares two numbers, if both are same then returns true. [$a == $b] would return false.
!= (Not Equality) Compares two numbers, if both are different then returns true. [$a != $b] would return true.

9.The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression
(grep stands for global search for regular expression and print out).
Syntax:

grep [options] pattern [files]

Options Description
-c : This prints only a count of the lines that match a pattern
-h : Display the matched lines, but do not display the filenames.
-i : Ignores, case for matching
-l : Displays list of a filenames only.
-n : Display the matched lines and their line numbers.
-v : This prints out all the lines that do not matches the pattern
-e exp : Specifies expression with this option. Can use multiple times.

Assignment 1B:

1. What is shell scripting?

A shell script is a list of commands in a computer program that is run by the Unix shell which is a command line interpreter. A shell script usually has comments that describe the steps. The different operations performed by shell scripts are program execution, file manipulation and text printing. A wrapper is also a kind of shell script that creates the program environment, runs the program etc.

2. What are different types of shell?

>. The Bourne Shell (sh)

Developed at AT&T Bell Labs by Steve Bourne, the Bourne shell is regarded as the first UNIX shell ever. It is denoted as sh. It gained popularity due to its compact nature and high speeds of operation.

This is what made it the default shell for Solaris OS. It is also used as the default shell for all Solaris system administration scripts. Start reading about shell scripting here.

>. The GNU Bourne-Again Shell (bash)

More popularly known as the Bash shell, the GNU Bourne-Again shell was designed to be compatible with the Bourne shell. It incorporates useful features from different types of shells in Linux such as Korn shell and C shell.

It allows us to automatically recall previously used commands and edit them with help of arrow keys

The complete path-name for the GNU Bourne-Again shell is /bin/bash. By default, it uses the prompt bash-VersionNumber# for the root user and bash-VersionNumber$ for the non-root users.

>. The C Shell (csh)

The C shell was created at the University of California by Bill Joy. It is denoted as csh. It was developed to include useful programming features like in-built support for arithmetic operations and a syntax similar to the C programming language.

Further, it incorporated command history which was missing in different types of shells in Linux like the Bourne shell. Another prominent feature of a C shell is “aliases”.

The complete path-name for the C shell is /bin/csh. By default, it uses the prompt hostname# for the root user and hostname% for the non-root users.

>. The Korn Shell (ksh)

The Korn shell was developed at AT&T Bell Labs by David Korn, to improve the Bourne shell. It is denoted as ksh. The Korn shell is essentially a superset of the Bourne shell.

Besides supporting everything that would be supported by the Bourne shell, it provides users with new functionalities. It allows in-built support for arithmetic operations while offereing interactive features which are similar to the C shell.

The Korn shell runs scripts made for the Bourne shell, while offering string, array and function manipulation similar to he C programming language. It also supports scripts which were written for the C shell. Further, it is faster than most different types of shells in Linux, including the C shell.

The complete path-name for the Korn shell is /bin/ksh. By default, it uses the prompt # for the root user and $ for the non-root users.

Assignment 2:

1. What is system call?

A system call is a procedure that provides the interface between a process and the operating system. It is the way by which a computer program requests a service from the kernel of the operating system.

2. What is fork system call?

fork() is used to create processes. It takes no arguments and returns a process ID. The purpose of fork() is to create a new process, which becomes the child process of the caller. After a new child process is created, both processes will execute the next instruction following the fork() system call.

3. What are the return values of fork?Done

4. What is pid?

PID is the operating system's unique identifier for active programs that are running

5. What is ppid?

PPID. A process that creates a new process is called a parent process; the new process is called a child process. The parent process ID (PPID) becomes associated with the new child process when it is created.

6. How to get the values of pid and ppid? (getpid, getppid)Done

7. What is wait system call?

The wait() system call suspends execution of the current process until one of its children terminates.

8. What is execve system call?

execve() executes the program referred to by pathname. This causes the program that is currently being run by the calling process to be replaced with a new program, with newly initialized stack, heap, and (initialized and uninitialized) data segments.

9. How to create orphan and zombie state using wait system call?

Zombie process is that process which has terminated but whose process control block has not been cleaned up from main memory because the parent process was not waiting for the child. If there are a large number of zombie processes, their PCBs IN WORST CASE can occupy the whole RAM.

Orphan process is a process that doesn’t have a parent process. This can happen when the parent process terminates due to some reasons before the completion of the child.

Assignment 3:

1. What is waiting time?
   WT refers to the total time that a process spends while waiting in a ready queue before reaching the CPU.
2. What is turnaround time?
   TAT refers to the total time interval present between the time of process submission and the time of its completion.
3. What is Burst time?
   Burst time is the total time taken by the process for its execution on the CPU.
4. What is arrival time?
   Arrival time is the time when a process enters into the ready state and is ready for its execution.
5. What is the formula for waiting time?
   TAT – BT = WT (BT - Burst time,TAT - turn around time)
6. What is the formula for turnaround time?
   BT + WT = TAT

Assignment 4:

1. What is producer consumer/reader writer problem?

Producer-Consumer problem and Readers-Writers problem at first can look very similar. But conceptually those are two separate synchronization problems.

PCP - Producer-Consumer Problem
RWP - Readers-Writers Problem

Shared resource
In PCP case a shared resource is some collection of messages - bounded/unbounded buffer. Each message is put to buffer by the producer. Where in RWP problem a shared resource is a more generalized abstraction - it can be anything that can be writable, buffer, array or just a single variable.

In PCP case it is about the single message in a buffer, where is RWP it is about the whole resource.

Usage
PCP is related to Queues systems, IPC's Pipes where RWP is related to Relational Database systems - for example all (S)hared,(Ex)clusive,(U)pdate locks.

2. What is up and down code???????
3. What is pseudo code for producer consumer/Reader writer problem?

```
Producer()
{
    while (1)
    {
       <<< produce item >>>
      P(empty); /* Get an empty buffer (decrease count) , block if unavail */
      P(mutex); /* acquire critical section: shared buffer */

      <<< critical section: Put item into shared buffer >>>

      V(mutex); /* release critical section */
      V(full); /* increase number of full buffers */
    }
}


Consumer()
{
    while (1)
    {
      P(full);
      P(mutex);

      <<< critical section: Remove item from shared buffer */

      V(mutex);
      V(empty);
```

```
Writer()
   {
      while (1)
      {
         P(writing);
         <<< perform write >>>
         V (writing);
      }
   }


   Reader() {
     while (1)   {
         P(mutex);
         rd_count++;
         if (1 == rd_count) /* If we are the first reader -- get write lock */
              P(writing); /* Once we have it, it keeps writers at bay */
         V(mutex); /*

         <<< perform read >>>

         P(mutex)
         rd_count--;
         if ( 0 == rd_count) /* If we are the last reader to leave -- */
              V(writing);   /* Allow writers */
         V(mutex);
      }
   }
```

4. What is critical section?
   Critical Section refers to the segment of code or the program which tries to access or modify the value of the variables in a shared resource.
5. What is mutual exclusion?
   A mutual exclusion (mutex) is a program object that prevents simultaneous access to a shared resource. This concept is used in concurrent programming with a critical section, a piece of code in which processes or threads access a shared resource

6. What is pseudo code for producer consumer / reader writer problem using sempahore?Done
7. Difference between producer and consumer/ reader writer?Done

8. What is process synchronization?

Processes Synchronization or Synchronization is the way by which processes that share the same memory space are managed in an operating system. It helps maintain the consistency of data by using variables or hardware so that only one process can make changes to the shared memory at a time.

Assignment 4:

1. **What is producer consumer/reader writer problem?**

   Producer-Consumer problem is a classical synchronization problem in the operating system. With the presence of more than one process and limited resources in the system the synchronization problem arises. If one resource is shared between more than one process at the same time then it can lead to data inconsistency. In the producer-consumer problem, the producer produces an item and the consumer consumes the item produced by the producer

2. **What is up and down code?**

In the top-down model, an overview of the system is formulated without going into detail for any part of it. Each part of it then refined into more details, defining it in yet more details until the entire specification is detailed enough to validate the model. if we glance at a haul as a full, it's going to appear not possible as a result of it's so complicated For example: Writing a University system program, writing a word processor. Complicated issues may be resolved victimization high down style, conjointly referred to as Stepwise refinement where,

1. We break the problem into parts,
2. Then break the parts into parts soon and now each of parts will be easy to do.

3. **What is pseudo code for producer consumer/Reader writer problem?**

   The producer-consumer problem is **an example of a multi-process synchronization problem**. The problem describes two processes, the producer and the consumer that shares a common fixed-size buffer use it as a queue. The producer's job is to generate data, put it into the buffer, and start again

4. **What is critical section?**

   What is the Critical Section in OS? Critical Section refers to **the segment of code or the program which tries to access or modify the value of the variables in a shared resource**.

5. **What is mutual exclusion?**

   A mutual exclusion (mutex) is **a program object that prevents simultaneous access to a shared resource**. This concept is used in concurrent programming with a critical section, a piece of code in which processes or threads access a shared resource

6.What is pseudo code for producer consumer / reader writer problem using sempahore?

Prerequisite – Semaphores in operating system, Inter Process Communication

Producer consumer problem is a classical synchronization problem. We can solve this problem by using semaphores.

A semaphore S is an integer variable that can be accessed only through two standard operations : wait() and signal().
The wait() operation reduces the value of semaphore by 1 and the signal() operation increases its value by 1.

```
wait(S){
while(S<=0);   // busy waiting
S--;
}


signal(S){

S++;

 }
```

Semaphores are of two types:

Binary Semaphore – This is similar to mutex lock but not the same thing. 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 problem with multiple processes.

Counting Semaphore – Its value can range over an unrestricted domain. It is used to control access to a resource that has multiple instances.

6. **Difference between producer and consumer/ reader writer?**

7. **What is process synchronization?**
   Processes Synchronization or Synchronization is the way by which processes that share the same memory space are managed in an operating system. It helps maintain the consistency of data by using variables or hardware so that only one process can make changes to the shared memory at a time

Assignment No 5:

1. What is deadlock?

A deadlock is **a situation in which two computer programs sharing the same resource are effectively preventing each other from accessing the resource, resulting in both programs ceasing to function**. The earliest computer operating systems ran only one program at a time.

2. What are the necessary conditions for deadlock?

necessary for the occurrence of a deadlock. They can be understood with the help of the above illustrated example of staircase:

Mutual Exclusion: When two people meet in the landings, they can't just walk through because there is space only for one person. This condition allows only one person (or process) to use the step between them (or the resource) is the first condition necessary for the occurrence of the deadlock.

Hold and Wait: When the two people refuse to retreat and hold their ground, it is called holding. This is the next necessary condition for deadlock.

No Preemption: For resolving the deadlock one can simply cancel one of the processes for other to continue. But the Operating System doesn't do so. It allocates the resources to the processors for as much time as is needed until the task is completed. Hence, there is no temporary reallocation of the resources. It is the third condition for deadlock.

Circular Wait: When the two people refuse to retreat and wait for each other to retreat so that they can complete their task, it is called circular wait. It is the last condition for deadlock to occur.

3. What is the use of bankers' algorithm?

Bankers algorithm in Operating System is used **to avoid deadlock and for resource allocation safely to each process in the system**. As the name suggests, it is mainly used in the banking system to check whether the loan can be sanctioned to a person or not.

It is a banker algorithm used **to avoid deadlock and allocate resources safely to each process in the computer system**. The 'S-State' examines all possible tests or activities before deciding whether the allocation should be allowed to each process

4. What are the Strategies to deal with deadlock? (Prevention, Avoidance, Detection and Recovery)

Deadlock Prevention

We can prevent Deadlock by eliminating any of the above four conditions.

Eliminate Mutual Exclusion
It is not possible to dis-satisfy the mutual exclusion because some resources, such as the tape drive and printer, are inherently non-shareable.
Eliminate Hold and wait

1. Allocate all required resources to the process before the start of its execution, this way hold and wait condition is eliminated but it will lead to low device utilization. for example, if a process requires printer at a later time and we have allocated printer before the start of its execution printer will remain blocked till it has completed its execution.

2. The process will make a new request for resources after releasing the current set of resources. This solution may lead to starvation.

Eliminate No Preemption
Preempt resources from the process when resources required by other high priority processes.
Eliminate Circular Wait
Each resource will be assigned with a numerical number. A process can request the resources increasing/decreasing. order of numbering.
For Example, if P1 process is allocated R5 resources, now next time if P1 ask for R4, R3 lesser than R5 such request will not be granted, only request for resources more than R5 will be granted.

Deadlock Avoidance

Deadlock avoidance can be done with Banker's Algorithm.

Deadlock Detection :

1. If resources have a single instance –
In this case for Deadlock detection, we can run an algorithm to check for the cycle in the Resource Allocation Graph. The presence of a cycle in the graph is a sufficient condition for deadlock.

2. In the above diagram, resource 1 and resource 2 have single instances. There is a cycle R1 → P1 → R2 → P2. So, Deadlock is Confirmed.

3. If there are multiple instances of resources –
Detection of the cycle is necessary but not sufficient condition for deadlock detection, in this case, the system may or may not be in deadlock varies according to different situations.

Deadlock Recovery :

A traditional operating system such as Windows doesn't deal with deadlock recovery as it is a time and space-consuming process. Real-time operating systems use Deadlock recovery.

1. **Killing the process –**
   Killing all the processes involved in the deadlock. Killing process one by one. After killing each process check for deadlock again keep repeating the process till the

system recovers from deadlock. Killing all the processes one by one helps a system to break circular wait condition.

2. **Resource Preemption –**
   Resources are preempted from the processes involved in the deadlock, preempted resources are allocated to other processes so that there is a possibility of recovering the system from deadlock. In this case, the system goes into starvation.

5. How bankers algorithm avoid deadlock?

Bankers algorithm consists of two main algorithms used to avoid deadlock and control the processes within the system: **Safety algorithm and Resource request algorithm**. Safety algorithm in OS is used to mainly check whether the system is in a safe state or not.

6. What is need matrix, allocation matrix and maximum matrix?

Available: It is an array of length 'm' that defines each type of resource available in the system. When Available[j] = K, means that 'K' instances of Resources type R[j] are available in the system.

Max: It is a [n x m] matrix that indicates each process P[i] can store the maximum number of resources R[j] (each type) in a system.

Allocation: It is a matrix of m x n orders that indicates the type of resources currently allocated to each process in the system. When Allocation [i, j] = K, it means that process P[i] is currently allocated K instances of Resources type R[j] in the system.

Need: It is an M x N matrix sequence representing the number of remaining resources for each process. When the Need[i] [j] = k, then process P[i] may require K more instances of resources type Rj to complete the assigned work. Nedd[i][j] = Max[i][j] - Allocation[i][j].

Finish: It is the vector of the order **m**. It includes a Boolean value (true/false) indicating whether the process has been allocated to the requested resources, and all resources have been released after finishing its task.

7. Steps to solve the problem of banker algorithm?

Banker's algorithm consists of Safety algorithm and Resource request algorithm

1)Safety Algorithm

The algorithm for finding out whether or not a system is in a safe state can be described as follows:

Let Work and Finish be vectors of length 'm' and 'n' respectively.
Initialize: Work = Available
Finish[i] = false; for i=1, 2, 3, 4….n
2) Find an i such that both
a) Finish[i] = false
b) $Need_i$ <= Work
if no such i exists goto step (4)
3) Work = Work + Allocation[i]
Finish[i] = true
goto step (2)
4) if Finish [i] = true for all i
then the system is in a safe state

1) **Resource-Request Algorithm**

Let $Request_i$ be the request array for process P_i. $Request_i$ [j] = k means process P_i wants k instances of resource type R_j. When a request for resources is made by process P_i, the following actions are taken:

1) If $Request_i$ <= $Need_i$
Goto step (2) ; otherwise, raise an error condition, since the process has exceeded its maximum claim.
2) If $Request_i$ <= Available
Goto step (3); otherwise, P_i must wait, since the resources are not available.
3) Have the system pretend to have allocated the requested resources to process Pi by modifying the state as follows:
Available = Available – Requesti
$Allocation_i$ = $Allocation_i$ + $Request_i$
$Need_i$ = $Need_i$– $Request_i$

Assignment No 6:

1. What is virtual memory?

Virtual memory is **a common technique used in a computer's operating system (OS)**. Virtual memory uses both hardware and software to enable a computer to compensate for physical memory shortages, temporarily transferring data from random access memory (RAM) to disk storage.

Example:-

A business owner uses their computer's virtual memory system when running multiple applications simultaneously.

2. What is physical and logical address?

Parameter	LOGICAL ADDRESS	PHYSICAL ADDRESS
Basic	generated by CPU	location in a memory unit
Address Space	Logical Address Space is set of all logical addresses generated by CPU in reference to a program.	Physical Address is set of all physical addresses mapped to the corresponding logical addresses.
Visibility	User can view the logical address of a program.	User can never view physical address of program.
Generation	generated by the CPU	Computed by MMU
Access	The user can use the logical address to access the physical address.	The user can indirectly access physical address but not directly.
Editable	Logical address can be change.	Physical address will not change.
Also called	virtual address.	real address.

3. What is page?

A page, also known as a memory page or virtual page, refers to **a contiguous block of virtual memory with a defined length that is described by one single entry in a page table**. In a virtual memory OS, it is the smallest data unit for memory management.

4. What is frames?

A frame refers to **a storage frame or central storage frame**. In terms of physical memory, it is a fixed sized block in physical memory space, or a block of central storage. In computer architecture, frames are analogous to logical address space pages

5. Mapping of addresses?

Address mapping is **a process of determining a logical address knowing the physical address of the device and determining the physical address by knowing the logical address of the device**. Address mapping is required when a packet is routed from source host to destination host in the same or different network.

How many types of address types are there in OS?

In OS we have **two** kind of addresses: Virtual/Logical Address, and Physical Address.

6. What is the concept of paging?

Paging is a storage mechanism used in OS to retrieve processes from secondary storage to the main memory as pages. The primary concept behind paging is **to break each process into individual pages**. Thus the primary memory would also be separated into frames.
Paging is **a function of memory management where a computer will store and retrieve data from a device's secondary storage to the primary storage**. Memory management is a crucial aspect of any computing device, and paging specifically is important to the implementation of virtual memory.

7. What are the page replacement policies?

Page replacement is needed in the operating systems that use virtual memory using Demand Paging. As we know that in Demand paging, only a set of pages of a process is loaded into the memory. This is done so that we can have more processes in the memory at the same time.

When a page that is residing in virtual memory is requested by a process for its execution, the Operating System needs to decide which page will be replaced by this requested page. This process is known as page replacement and is a vital component in virtual memory management.

Need Page Replacement Algorithms:-

To understand why we need page replacement algorithms, we first need to know about page faults. Let's see what is a page fault.

Page Fault: A Page Fault occurs when a program running in CPU tries to access a page that is in the address space of that program, but the requested page is currently not loaded into the main physical memory, the RAM of the system

8. What is page fault?

in computing, a page fault (sometimes called PF or hard fault) is **an exception that the memory management unit (MMU) raises when a process accesses a memory page without proper preparations**. Accessing the page requires a mapping to be added to the process's virtual address space.

9. What is page hit?

When the CPU attempts to obtain a needed page from main memory and the page exists in main memory (RAM), it is referred to as a "PAGE HIT".

10. Concept of LRU, Optimal and FIFO in detail.

- Page Replacement

First In First Out (FIFO)

This is the simplest page replacement algorithm. In this algorithm, the OS maintains a queue that keeps track of all the pages in memory, with the oldest page at the front and the most recent page at the back.

When there is a need for page replacement, the FIFO algorithm, swaps out the page at the front of the queue, that is the page which has been in the memory for the longest time.

Advantages

- Simple and easy to implement.
- Low overhead.

Disadvantages

- Poor performance.
- Doesn’t consider the frequency of use or last used time, simply replaces the oldest page.
- Suffers from Belady’s Anomaly(i.e. more page faults when we increase the number of page frames).

Least Recently Used (LRU)

Least Recently Used page replacement algorithm keeps track of page usage over a short period of time. It works on the idea that the pages that have been most heavily used in the past are most likely to be used heavily in the future too.

In LRU, whenever page replacement happens, the page which has not been used for the longest amount of time is replaced.

Advantages

- Efficient.
- Doesn't suffer from Belady’s Anomaly.

Disadvantages

- Complex Implementation.
- Expensive.
- Requires hardware support.

Optimal Page Replacement

Optimal Page Replacement algorithm is the best page replacement algorithm as it gives the least number of page faults. It is also known as OPT, clairvoyant replacement algorithm, or Belady’s optimal page replacement policy.

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future, i.e., the pages in the memory which are going to be referred farthest in the future are replaced.

This algorithm was introduced long back and is difficult to implement because it requires future knowledge of the program behaviour. However, it is possible to implement optimal page replacement on the second run by using the page reference information collected on the first run.

11. **Difference between 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.

Assignment no 7:

1. What is Interprocess communication?

Interprocess communication is **the mechanism provided by the operating system that allows processes to communicate with each other**. This communication could involve a process letting another process know that some event has occurred or the transferring of data from one process to another.23-Jun-2020

IPC is **the way by which multiple processes or threads communicate among each other**. IPC in OS obtains modularity, computational speedup and data sharing. Different ways of IPC are pipe, message passing, message queue, shared memory, direct communication, indirect communication and FIFO

2. What is cooperating processes?

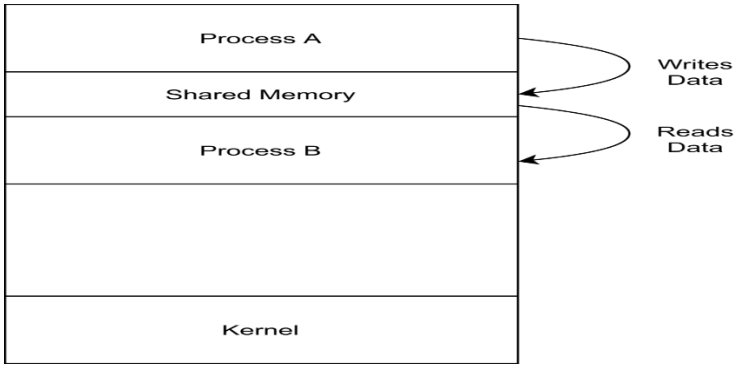
Cooperating processes can be used **to share information between various processes**. It could involve having access to the same files. A technique is necessary so that the processes may access the files concurrently. Modularity refers to the division of complex tasks into smaller subtasks.

3. What are the two ways of interprocess communication (Message passing and shared memory)?

There are two modes through which processes can communicate with each other – shared memory and message passing. As the name suggests, the shared memory region shares a shared memory between the processes. On the other hand, the message passing lets processes exchange information through messages. Let's explore these in detail in the subsequent sections.

Shared Memory

Interprocess communication through the shared memory model **requires communicating processes to establish a shared memory region**. In general, the process that wants to communicate creates the shared memory region in its own address space. Other processes that wish to communicate to this process need to attach their address space to this shared memory segment:



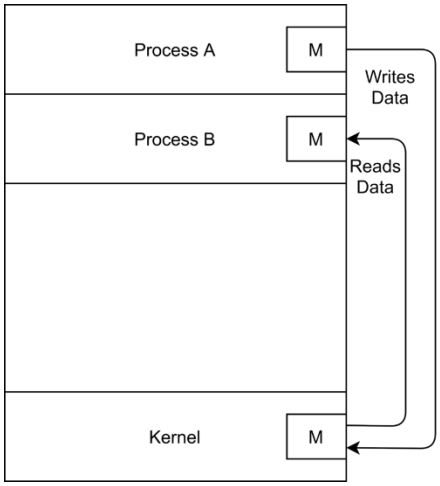
The above figure demonstrates the shared memory model of IPC. Process A and process B establishes a shared memory segment and exchange information through the shared memory region.

By default, the operating system prevents processes from accessing other process memory. The shared memory model requires processes to agree to remove this restriction. Besides, as shared memory is established based on the agreement between processes, the processes are also responsible to ensure synchronization so that both processes are not writing to the same location at the same time.

Message Passing

Although the shared memory model is useful for process communication, it is not always suitable and achievable. For instance, in a distributed computing environment, the processes exchanging data might reside in different computer systems. Thus, it is not straightforward to establish a shared memory region for communication.

The message passing mechanism provides an alternative means processes for communication. In this mode, **processes interact with each other through messages with assistance from the underlying operating system:**



In the above diagram two processes A, and B are communicating with each other through message passing. Process A sends a message M to the operating system (kernel). This message is then read by process B.

In order to successfully exchange messages, there needs to be a communication link between the processes. There are several techniques through which these communication links are established. Following are a bride overview of these mechanisms:

- Direct Communication: In the mode, each process explicitly specifies the recipient or the sender. For instance, if process A needs to send a message to process B, it can use the primitive
- Indirect Communication: In this mode, processes can exchange messages through a mailbox. A mailbox is a container that holds the messages. For instance, if X is a mailbox, then process A can send a message to mailbox using the primitive
- Synchronization: This is an extension to direct and indirect communication with an additional option of synchronization. Based on the need. a process can choose to block while sending or receiving messages. Besides, it can also asynchronously communicate without any blocking
- Buffering: The exchanged messages reside in a temporary queue. These queues can be of zero, bounded, and unbounded capacity

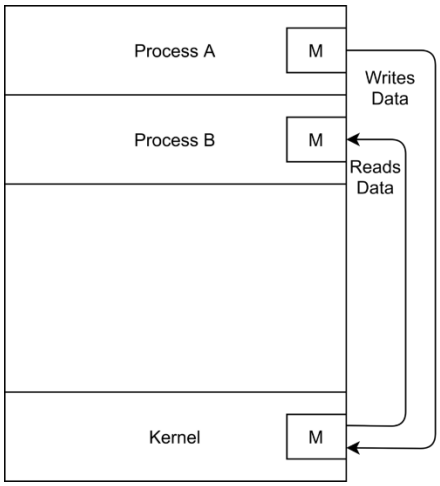
One classical **example of IPC is the producer-consumer problem**. A producer is an application/process that produces data. A consumer is an application/process that consumes the produced data. The producer and consumer processes can decide any of the mechanisms discussed before to communicate with each other.

4. What is Message passing? Explain in detail

Message Passing

Although the shared memory model is useful for process communication, it is not always suitable and achievable. For instance, in a distributed computing environment, the processes exchanging data might reside in different computer systems. Thus, it is not straightforward to establish a shared memory region for communication.

The message passing mechanism provides an alternative means processes for communication. In this mode, **processes interact with each other through messages with assistance from the underlying operating system:**



In the above diagram two processes A, and B are communicating with each other through message passing. Process A sends a message M to the operating system (kernel). This message is then read by process B.

In order to successfully exchange messages, there needs to be a communication link between the processes. There are several techniques through which these communication links are established. Following are a bride overview of these mechanisms:

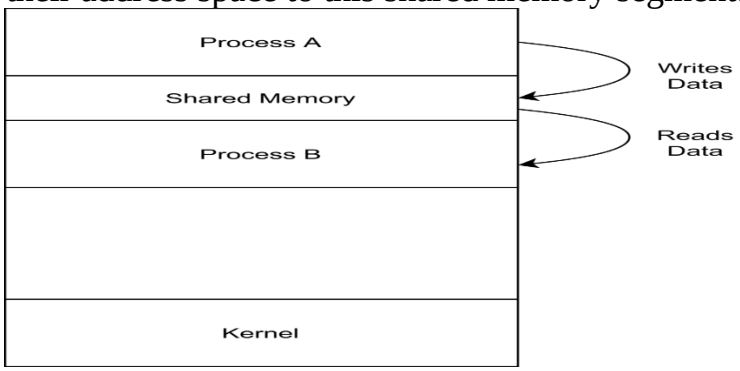
- Direct Communication: In the mode, each process explicitly specifies the recipient or the sender. For instance, if process A needs to send a message to process B, it can use the primitive
- Indirect Communication: In this mode, processes can exchange messages through a mailbox. A mailbox is a container that holds the messages. For instance, if X is a mailbox, then process A can send a message to mailbox using the primitive
- Synchronization: This is an extension to direct and indirect communication with an additional option of synchronization. Based on the need. a process can choose to block while sending or receiving messages. Besides, it can also asynchronously communicate without any blocking
- Buffering: The exchanged messages reside in a temporary queue. These queues can be of zero, bounded, and unbounded capacity

One classical **example of IPC is the producer-consumer problem**. A producer is an application/process that produces data. A consumer is an application/process that consumes the produced data. The producer and consumer processes can decide any of the mechanisms discussed before to communicate with each other.

5. What is Shared memory? Explain in detail.

Shared Memory

Interprocess communication through the shared memory model **requires communicating processes to establish a shared memory region**. In general, the process that wants to communicate creates the shared memory region in its own address space. Other processes that wish to communicate to this process need to attach their address space to this shared memory segment:

6. 

The above figure demonstrates the shared memory model of IPC. Process A and process B establishes a shared memory segment and exchange information through the shared memory region.

7.

By default, the operating system prevents processes from accessing other process memory. The shared memory model requires processes to agree to remove this restriction. Besides, as shared memory is established based on the agreement between processes, the processes are also responsible to ensure synchronization so that both processes are not writing to the same location at the same time.

8. What is pipes?

A pipe simply refers to **a temporary software connection between two programs or commands**. An area of the main memory is treated like a virtual file to temporarily hold data and pass it from one process to another in a single direction. In OSes like Unix, a pipe passes the output of one process to another process.

9. * Which functions are used to create pipes?

Creating ``pipelines'' with the C programming language can be a bit more involved than our simple shell example. To create a simple pipe with C, we

make use of the pipe() system call. It takes a single argument, which is an array of two integers, and if successful, the array will contain two new file descriptors to be used for the pipeline. After creating a pipe, the process typically spawns a new process (remember the child inherits open file descriptors).

10. Which functions are used to create shared memory?

Two functions **shmget() and shmat()** are used for IPC using shared memory. shmget() function is used to create the shared memory segment, while the shmat() function is used to attach the shared segment with the process's address space.

Assignment No 8:

1. What is seek time?

Seek time is **the time taken for a hard disk controller to locate a specific piece of stored data**. Other delays include transfer time (data rate) and rotational delay (latency). When anything is read or written to a disc drive, the read/write head of the disc needs to move to the right position.

2. What is the use of SSTF, SCAN and C-LOOK?

The full form of SSTF is Shortest Seek Time First. SSTF is a secondary storage scheduling algorithm that determines the motion of the disk's head and arm in servicing the read and write requests. SSTF acts as a disk scheduling algorithm, and it is an improvement upon the FCFS algorithm.

The scan is **a disk scheduling algorithm that serves requests generated by memory management unit**. It is also called an elevator algorithm. In this algorithm read and write head has to move in one direction and fulfill all the requests until we move to the end of the disk.

C-LOOK is the modified version of both LOOK and C-scan algorithms. In this algorithm, the head starts from the first request in one direction and moves towards the last request at another end, serving all request in between

3. What is disk scheduling?

Disk Scheduling

As we know, a process needs two type of time, CPU time and IO time. For I/O, it requests the Operating system to access the disk.

However, the operating system must be fare enough to satisfy each request and at the same time, operating system must maintain the efficiency and speed of process execution.

The technique that operating system uses to determine the request which is to be satisfied next is called disk scheduling.

Let's discuss some important terms related to disk scheduling.

Seek Time

Seek time is the time taken in locating the disk arm to a specified track where the read/write request will be satisfied.

Rotational Latency

It is the time taken by the desired sector to rotate itself to the position from where it can access the R/W heads.

Transfer Time

It is the time taken to transfer the data.

Disk Access Time

Disk access time is given as,Disk Access Time = Rotational Latency + Seek Time + Transfer Time

Disk Response Time

It is the average of time spent by each request waiting for the IO operation.

Purpose of Disk Scheduling

The main purpose of disk scheduling algorithm is to select a disk request from the queue of IO requests and decide the schedule when this request will be processed.

Goal of Disk Scheduling Algorithm

- Fairness
- High throughout
- Minimal traveling head time

Disk Scheduling Algorithms

The list of various disks scheduling algorithm is given below. Each algorithm is carrying some advantages and disadvantages. The limitation of each algorithm leads to the evolution of a new algorithm.

- FCFS scheduling algorithm
- SSTF (shortest seek time first) algorithm
- SCAN scheduling
- C-SCAN scheduling
- LOOK Scheduling

- **C-LOOK scheduling**

4. What is head?

head is **a program on Unix and Unix-like operating systems used to display the beginning of a text file or piped data**.
head command is **a command-line utility, which prints the first 10 lines of the specified files**. If more than one file name is provided then data from each file is preceded by its file name. We can change the number of lines the head command prints by using the -n command line option.

5. *What are three factors that determine that determine disk arm scheduling algorithm?(Seek time, rotational delay, actual data transfer time)

Seek Time: Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or write. So the disk scheduling algorithm that gives minimum average seek time is better.
Rotational Latency: Rotational Latency is the time taken by the desired sector of disk to rotate into a position so that it can access the read/write heads. So the disk scheduling algorithm that gives minimum rotational latency is better.
Transfer Time: Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and number of bytes to be transferred.
Disk Access Time: Disk Access Time is:

```
Disk Access Time = Seek Time +
                   Rotational Latency +
                   Transfer Time
```

6. **Working of SSTF, SCAN and C-LOOK.**

SSTF Scheduling Algorithm

Shortest seek time first (SSTF) algorithm selects the disk I/O request which requires the least disk arm movement from its current position regardless of the direction. It reduces the total seek time as compared to FCFS.

It allows the head to move to the closest track in the service queue.

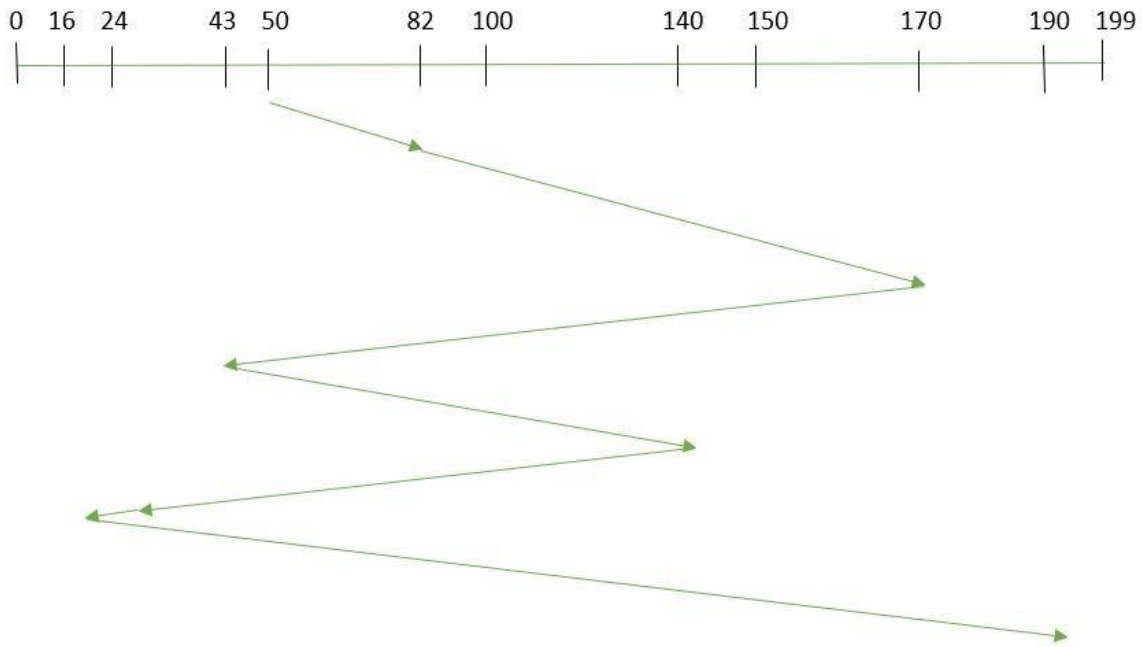
Disadvantages

- It may cause starvation for some requests.
- Switching direction on the frequent basis slows the working of algorithm.
- It is not the most optimal algorithm.

1. **FCFS:** FCFS is the simplest of all the Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.Let us understand this with the help of an example.

Example:

1. Suppose the order of request is- (82,170,43,140,24,16,190)
   And current position of Read/Write head is : 50



1. So, total seek time:
   =(82-50)+(170-82)+(170-43)+(140-43)+(140-24)+(24-16)+(190-16)
   =642

Advantages:

- Every request gets a fair chance
- No indefinite postponement

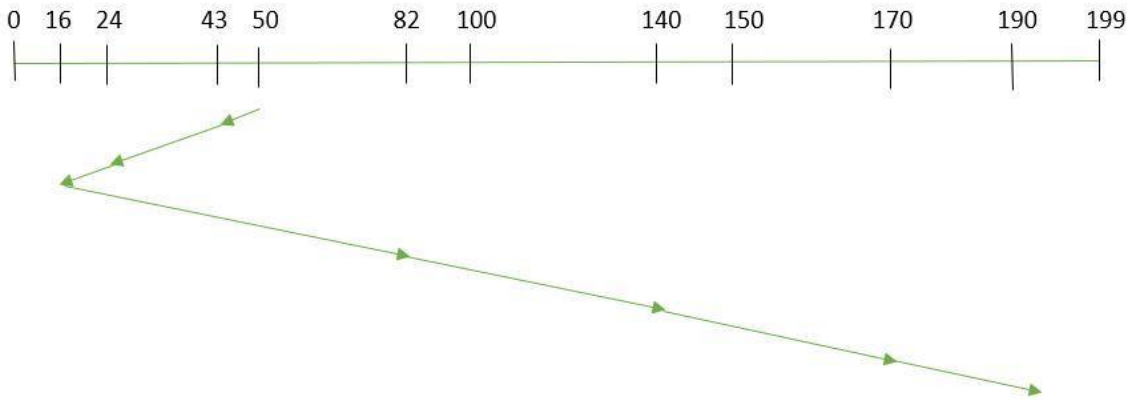
Disadvantages:

- Does not try to optimize seek time
- May not provide the best possible service

1. **SSTF:** In SSTF (Shortest Seek Time First), requests having shortest seek time are executed first. So, the seek time of every request is calculated in advance in the queue and then they are scheduled according to their calculated seek time. As a result, the request near the disk arm will get executed first. SSTF is certainly an improvement over FCFS as it decreases the average response time and increases the throughput of system.Let us understand this with the help of an example.

Example:

1. Suppose the order of request is- (82,170,43,140,24,16,190)
   And current position of Read/Write head is : 50



1.

So, total seek time:

1. =(50-43)+(43-24)+(24-16)+(82-16)+(140-82)+(170-140)+(190-170)
   =208

Advantages:

- Average Response Time decreases
- Throughput increases

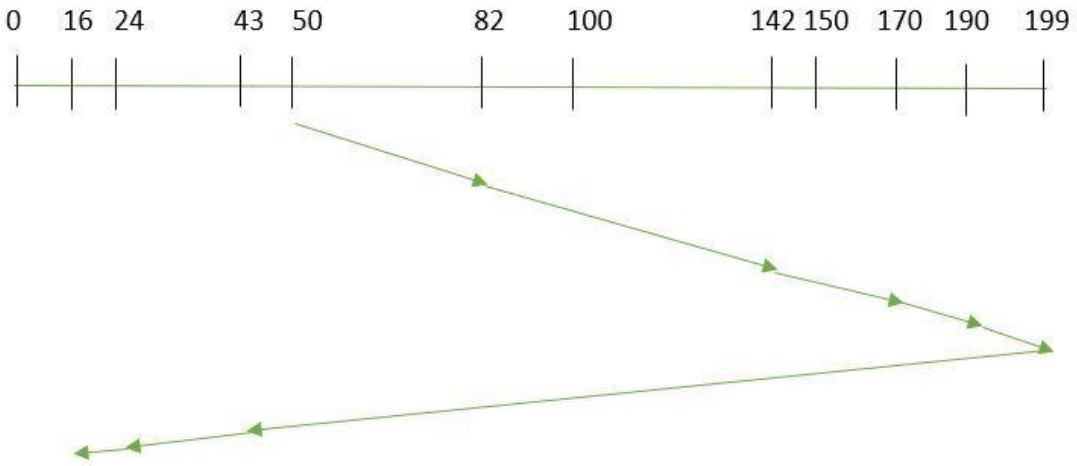
Disadvantages:

- Overhead to calculate seek time in advance
- Can cause Starvation for a request if it has higher seek time as compared to incoming requests
- High variance of response time as SSTF favours only some requests

1. **SCAN:** In SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path. So, this algorithm works as an elevator and hence also known as **elevator algorithm.** As a result, the requests at the midrange are serviced more and those arriving behind the disk arm will have to wait.

Example:

1. Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **“towards the larger value”.**



1.

Therefore, the seek time is calculated as:

1. =(199-50)+(199-16)
   =332

Advantages:

- High throughput
- Low variance of response time
- Average response time

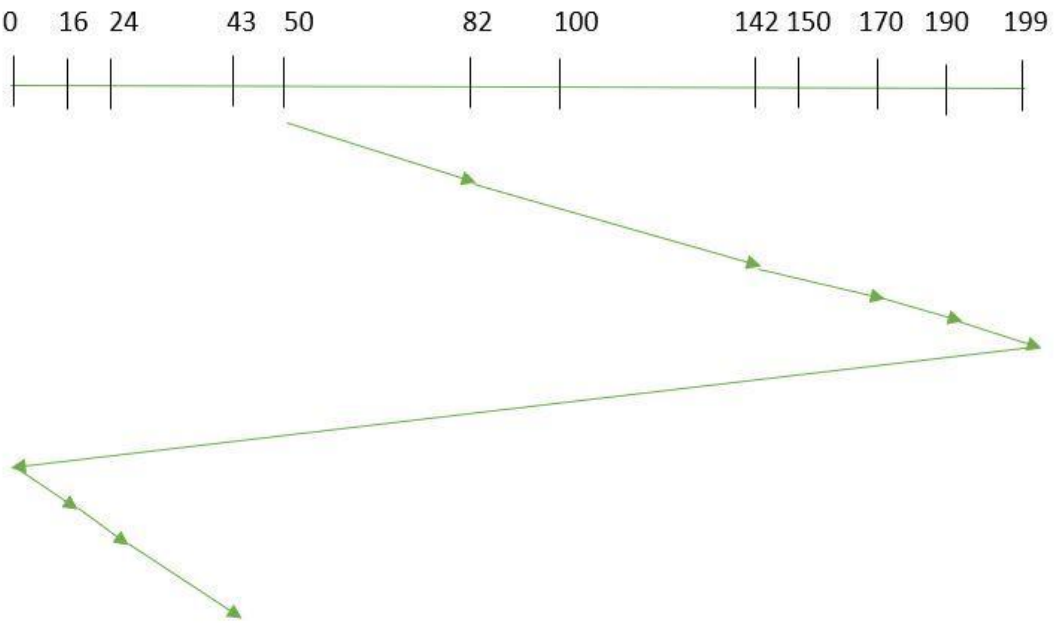
Disadvantages:

- Long waiting time for requests for locations just visited by disk arm

1. **CSCAN**: In SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction. So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending at the scanned area.

These situations are avoided in *CSCAN* algorithm in which the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the requests from there. So, the disk arm moves in a circular fashion and this algorithm is also similar to SCAN algorithm and hence it is known as C-SCAN (Circular SCAN).

Example:
Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **"towards the larger value".**



Seek time is calculated as:

=(199-50)+(199-0)+(43-0)
=391

Advantages:

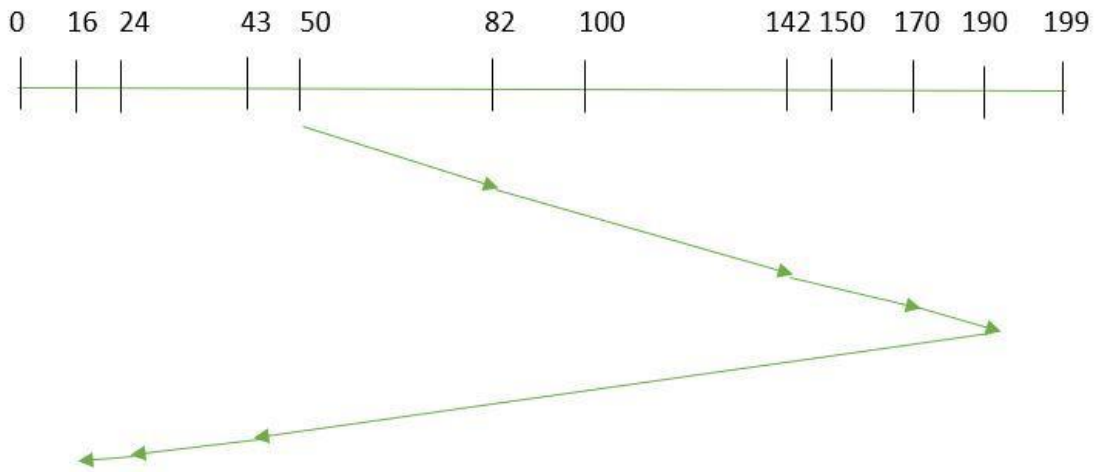
- Provides more uniform wait time compared to SCAN

1. **LOOK:** It is similar to the SCAN disk scheduling algorithm except for the difference that the disk arm in spite of going to the end of the disk

goes only to the last request to be serviced in front of the head and then reverses its direction from there only. Thus it prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

Example:

1. Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **"towards the larger value".**



1.

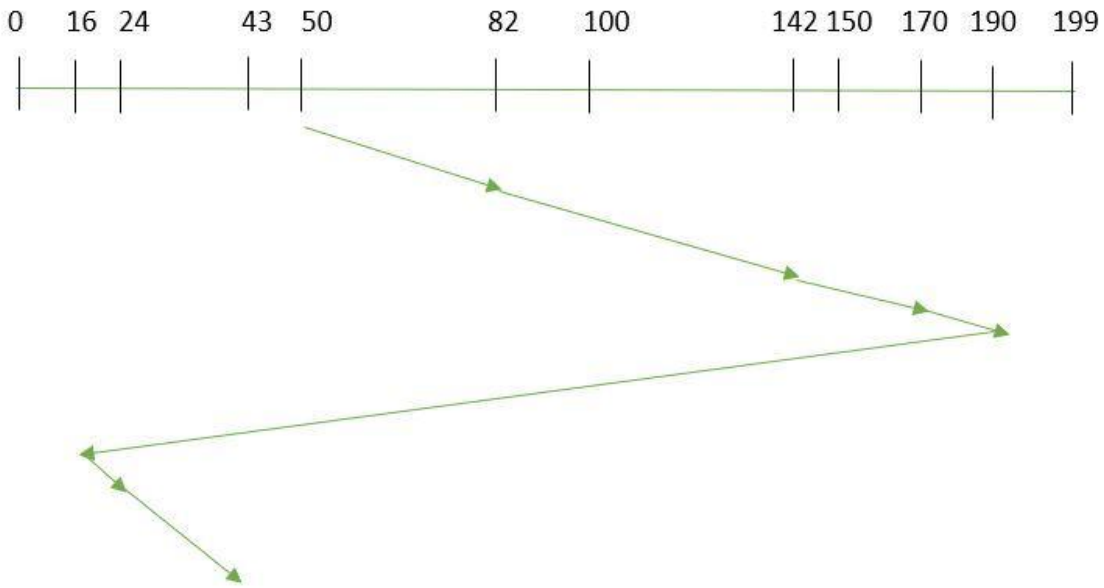
So, the seek time is calculated as:

1. =(190-50)+(190-16)
   =314

1. **CLOOK:** As LOOK is similar to SCAN algorithm, in similar way, CLOOK is similar to CSCAN disk scheduling algorithm. In CLOOK, the disk arm in spite of going to the end goes only to the last request to be serviced in front of the head and then from there goes to the other end's last request. Thus, it also prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

Example:

1. Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **“towards the larger value”**



1.

So, the seek time is calculated as:

1. =(190-50)+(190-16)+(43-16)
   =341
2. **RSS**– It stands for random scheduling and just like its name it is nature. It is used in situations where scheduling involves random attributes such as random processing time, random due dates, random weights, and stochastic machine breakdowns this algorithm sits perfect. Which is why it is usually used for and analysis and simulation.
3. **LIFO**– In LIFO (Last In, First Out) algorithm, newest jobs are serviced before the existing ones i.e. in order of requests that get serviced the job that is newest or last entered is serviced first and then the rest in the same order.
   Advantages
   - Maximizes locality and resource utilization
   - Can seem a little unfair to other requests and if new requests keep coming in, it cause starvation to the old and existing ones.

4. **N-STEP SCAN** – It is also known as N-STEP LOOK algorithm. In this a buffer is created for N requests. All requests belonging to a buffer will be serviced in one go. Also once the buffer is full no new requests are kept in this buffer and are sent to another one. Now, when these N requests are serviced, the time comes for another top N requests and this way all get requests get a guaranteed service
   Advantages
   - It eliminates starvation of requests completely
5. **FSCAN**– This algorithm uses two sub-queues. During the scan all requests in the first queue are serviced and the new incoming requests are added to the second queue. All new requests are kept on halt until the existing requests in the first queue are serviced.
   Advantages
   - FSCAN along with N-Step-SCAN prevents “arm stickiness” (phenomena in I/O scheduling where the scheduling algorithm continues to service requests at or near the current sector and thus prevents any seeking)