STUDENT ADVISORY

Dear Students,

Please be informed that the notes provided by the institute offer a concise presentation of the syllabus. While these notes are helpful for an overview and quick revision, We would strongly suggest that you refer to the prescribed textbooks / Reference book for a comprehensive understanding and thorough preparation of all exams and writing in the examination.

Best regards,

LJ Polytechnic.

પ્રિય વિદ્યાર્થીઓ,

તમને જાણ કરવામા આવે છે કે સંસ્થા દ્વારા પ્રદાન કરવામાં આવેલી નોંધો અભ્યાસક્રમની સંક્ષિપ્ત પ્રસ્તુતિ આપે છે. આ નોંધો વિહંગાવલોકન અને ઝડપી પુનરાવર્તન માટે મદદરૂપ હોઈ શકે છે તેમ છતા, અમે ભારપૂર્વક સૂચન કરીએ છીએ કે વિદ્યાર્થી તમામ પરીક્ષાઓ અને પરીક્ષામાં લેખનની વ્યાપક સમજણ અને સંપૂર્ણ તૈયારી માટે માત્ર સૂચવેલા પાઠ્યપુસ્તકો/સંદર્ભ પુસ્તકનો સંદર્ભ લો.

એલજે પોલિટેકનિક.

Chapter-1 Introduction to Operating System

1 Define Operating System along with its Goals and Objectives.

Definition

"Operating System is a program/software that manages the computer hardware and provides simple interface to the hardware for user programs."

Goals of Operating System

- ➢ To hide details of hardware by creating abstraction
- ➢ To allocate resources to processes (Manage resources)
- ➢ Provide a pleasant and effective user interface

Objectives of Operating System

- ➢ Convenience for users
- ➢ Efficient operation of the computer system.

2 Explain Need of Operating System.

- ➢ Operating System is needed to enable the user to design the application without Concerning the details of the computer's internal structure.
- ➢ In general the boundary between the hardware & software is transparent to the user.

Usage of Operating System:

- ➢ Easy interaction between the human & computer.
- ➢ Starting computer operation automatically when power in turned on.
- ➢ Loading & scheduling users program.
- ➢ Controlling input & output.
- ➢ Controlling program execution.
- ➢ Managing use of main memory.
- ➢ Providing security to users program.

3 Explain various Services of Operating System.

<u>Services from System Point of View</u>

Resource Allocation

When multiple users are sharing the same machine, or when multiple jobs are running simultaneously, there is a need of fair allocation of resources among them. OS does this.

Accounting

Accounting is the process of keeping information about which user uses which resource, and for what duration of time. Such information can be used to bill the users in multi-user environment, or to get usage statistics to make future planning.

Protection

OS ensures that all access to system resources is controlled. Also security from outsiders is important.

Services from User Point of View

Program execution

The main purpose of OS is to provide an efficient and convenient environment the execution of program. So, an OS must provide various functions for loading a program into main memory, execute it and after execution, terminate it.

I/O Operation

A running program needs I/O operations for reading the data or to provide output to the user. User cannot control I/O devices directly for security reasons, OS provides services for various kinds of I/O operations.

Communication

In multitasking environment more than one process is running simultaneously. Sometimes there is a need of exchanging information among my processes Such processes may be on the same machine, or on different machines. OS provides mechanisms for such inter process communications.

Error detection

Errors may be in user programs, CPU and memory hardware, devices. OS detects such errors and makes user aware from them. It also provide some error recovery mechanism

4   Explain various functions of Operating System.

The main functions perform by Operating System are as follow: -

1. Process Management: - The process management module of an Operating System takes care of the creation & deletion of processes, scheduling of various system resources to the different process requesting them, & providing mechanism for synchronization & communication among processes.

2. Memory Management: - The memory management module of an Operating System takes care of the allocation & reallocation of memory space to the various programs in need of this resource.

3. File Management: - Computer use a lot of data & programs, which are, stored on secondary storage devices. File management functions of an Operating System involves keeping track of all different files & maintaining the integrity of data stored in the files including file directory structure.

4. Security: - The security modules of an Operating System protect the resources & information of a computer system against destruction& unauthorized access.

5. Command Interpretation: -The Command Interpretation module of an Operating System takes care of interpreting of user commands, & directing the system resources to handle the requests. With this mode of interaction with the system, the user is usually not too concerned with the hardware details of the system.

6. Input/output or Device Management: - Coordination & control of various input & output devices is an important function of the Operating System. This involves receiving the request for I/O interrupts, & communicating back to the requesting process.

7. Job Control: - When the user wants to run an application program, he must communicate with the Operating System telling it what to do. He does this using Operating System job control language or JCL. JCL consists of a number of Operating Systems commands, called system commands that control the functioning of the Operating System.
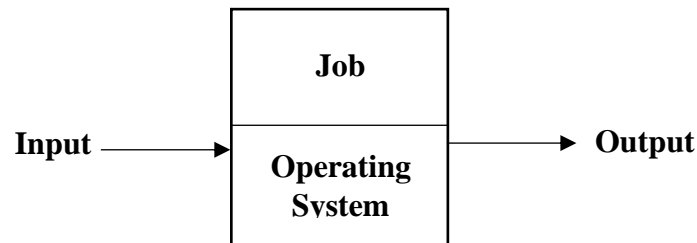
5   Explain Batch Operating System.

➢ First Operating System which was developed was Batch Operating System.

➢ The common input devices were card readers, tape drives.

➢ The common output devices were line printers, punch cards and tap drives.

Working

➢ The Operating System was very simple and always resident in main memory.

➢ Programmers used to prepare a job and submit it to the operator.

➢ Job was consisted of program, data and some control information.

➢ Operator used to sort them in batches with similar requirements, and as computer became available, run them batch wise.

➢ At some later time (may be after some hours or even after some days), output appeared. The output consisted of the result of the program and error information.

➢ Programmers needed to wait during this time, and then collect output from the operator.

➢ Here memory is divided in two parts the operating system and the job as shown in the figure. At a Time, one job is selected out of the batch of jobs and is loaded into memory for execution. Once its execution completes, another job is selected and loaded into memory for execution. This procedure will continue until all the jobs in batch get executed.

```
                    ┌──────────────┐
                    │     Job      │
                    ├──────────────┤
Input ─────────────▶│  Operating   │──────────▶ Output
                    │   System     │
                    └──────────────┘
```

Disadvantages

➢ Low throughput
➢ Programmer do not have interaction with job.
➢ Time Consuming
➢ Debugging was not possible in online mode.

6   Explain Multiprogramming Operating System.

➢ A multiprogramming Operating System provides the ability to run more than program concurrently

➢ Contrast to Batch Operating System, here more than one program for jobs loaded in main memory simultaneously. These programs can be executed concurrently Memory is shared between OS and such kind of programs as shown in figure.

➢ A simple multiprogramming Operating System works in a non-preemptive manner.

4

Job 1
Job 2
Job 3
Operating System

.
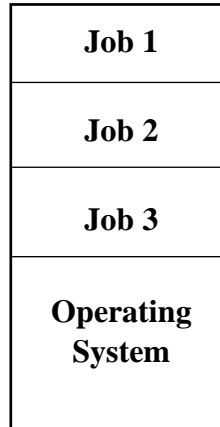
Here, a program is allowed to execute until it voluntarily gives up the CPU.

A program voluntarily gives up the CPU when it waits for some event, such as O operation, or when it terminates Once a CPU becomes free, it can be allocated to some other program.

The primary objective of Multiprogramming Operating System is to maximize CPU usage,

Advantages

➢ Multiprogramming significantly improves system throughput and resource utilization.

➢ Various resources, such as CPU, can be utilized as much as possible among various simultaneously executing programs.

7   Explain Time Sharing / Multitasking Operating System.

➢ Time Sharing Operating System is a logical extension of the Multi Program Operating System Here CPU is multiplied by time among several programs for jobs) kept is main memory and on disk.

➢ It provides direct interaction between user and the system.

Working

➢ It works in Preemptive manner.

➢ Here, a program is allowed to execute only for some maximum time duration

➢ After this time duration, a CPU is forcibly taken away from that program allocated to another program here, smaller programs need not to wait for other large program to finish Execution.

➢ So, it minimizes the response time for user And so, they are suitable for interactive programs in which user can direct interact with the program

➢ Such type of system is also referred as multitasking system User can execute more than one program and interact with them simultaneously.

➢ This means, can perform more than one task simultaneously, For example, user can execute.

➢ **Example:** User is working on some document in MS Word and at the same time he plays song in background. Both of these tasks can be performed simultaneously.

8   Explain Real Time Operating System.

➢ Real Time Operating Systems strictly require to process input data in a pre-specified can time duration.

➢ In such type of Operating Systems, time is the key parameter. Here, input immediately affects the output.

➢ Time is very critical Systems having such type of characteristics are systems to control nuclear power plants, oil refining, air traffic control systems, air defense systems, etc.

➢ In all such systems. Input coming from some sensors should be processed better immediate.

➢ All these operations must occur within some time limits.
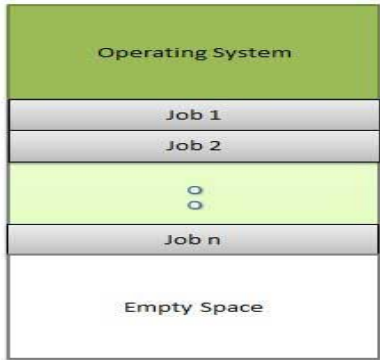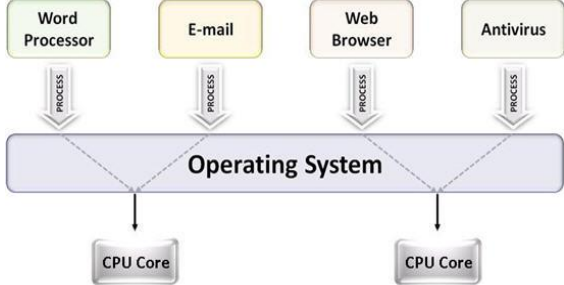
Types

Hard Real Time System

➢ All critical tasks must get completed strictly within the specified time limits.

➢ In other words, tasks are guaranteed to occur in time.

➢ **Example:** System which controls the operation of an oil refinery.

Soft Real Time System

➢ Time Limits is followed occasionally.

➢ In this type of system extension is given in time limits.

> ➢ **Example:** Digital audio or multimedia system.

9   Difference between Multiprogramming and Multiprocessing Operating System.

| Multiprogramming Operating System	Multiprocessing Operating System
Multiprogramming is the ability of an operating system to execute more than one program on a single processor machine.	Multiprocessing is the ability of an operating system to execute more than one process simultaneously on a multiprocessor machine.
The main idea of multiprogramming is to maximize the use of CPU time.	Multiprocessing refers to the hardware (i.e. the CPU units) rather than the software (i.e. running processes). If the underlying hardware provides more than one processor then that is multiprocessing.
Example: A computer running excel and Firefox browser simultaneously	**Example:** a computer uses more than one CPU at a time.

10.   List out Types of Operating System

1. Batch Operating System
2. Multiprogramming Operating System
3. Multitasking Operating System
4. Real time Operating System
5. Network Operating System

6. Distributed Operating System

7. Multiprocessing Operating System

11.   Explain Network Operating System

➤ These systems run on a server and provide the capability to manage data, users, groups, security, applications, and other networking functions.

➤ These types of operating systems allow shared access of files, printers, security, applications, and other networking functions over a small private network.

➤ One more important aspect of Network Operating Systems is that all the users are well aware of the underlying configuration, of all other users within the network, their individual connections, etc.
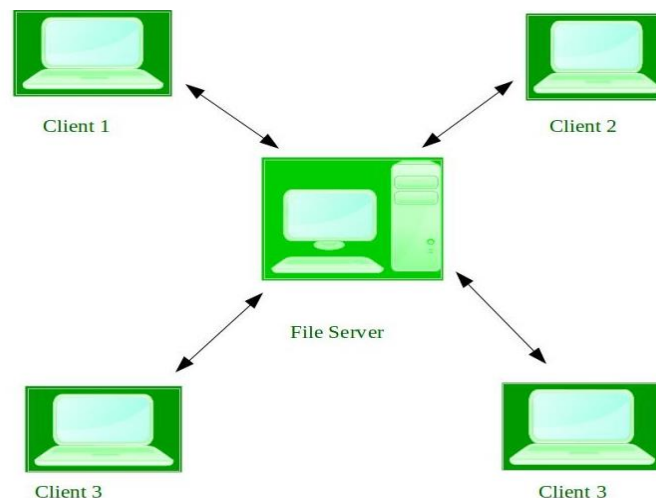
Advantages of Network Operating System:

➤ Highly stable centralized servers

➤ Security concerns are handled through servers

➤ New technologies and hardware up-gradation are easily integrated into the system

➤ Server access is possible remotely from different locations and types of systems

Disadvantages of Network Operating System:

➤ Servers are costly

➤ User has to depend on a central location for most operations

➢ Maintenance and updates are required regularly

Examples of Network Operating System are:

➢ Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD, etc.

Chapter-2: Process Management

1   **Explain process life cycle.**
OR
Explain various process state in brief.

➢ A process is a dynamic/active entity.
➢ It changes its during time duration of its execution.

Process states are:

New

When a process is first created, it occupies 'New' state. In this state process awaits to enter in Ready state.

Ready

New -> Ready to run. After the creation of a process, the process enters the ready state i.e., the process is loaded into the main memory. The process here is ready to run and is waiting to get the CPU time for its execution. Processes that are ready for execution by the CPU are maintained in a queue for ready processes.

Run

The process is chosen by CPU for execution and the instructions within the process are executed by any one of the available CPU cores.

Waiting

Whenever the process requests access to I/O or needs input from the user or needs access to a critical region (the lock for which is already acquired) it enters the blocked or wait state. The process continues to wait in the main memory and does not require CPU. Once the I/O operation is completed the process goes to the ready state.
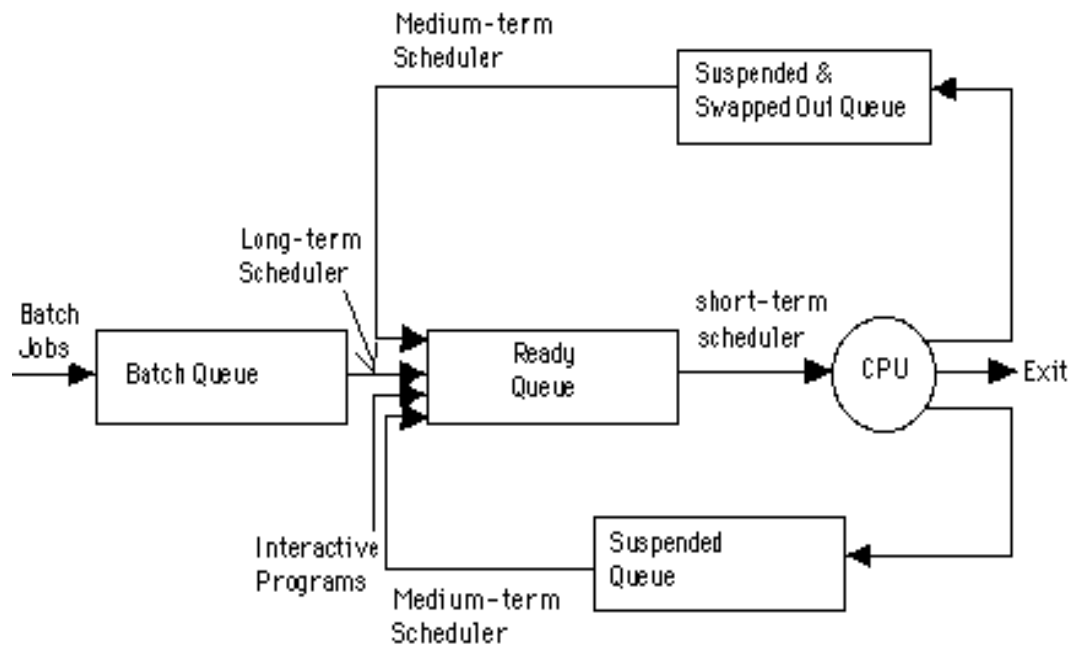
Terminated

Process is killed as well as PCB is deleted. A process can terminated by completing its execution.
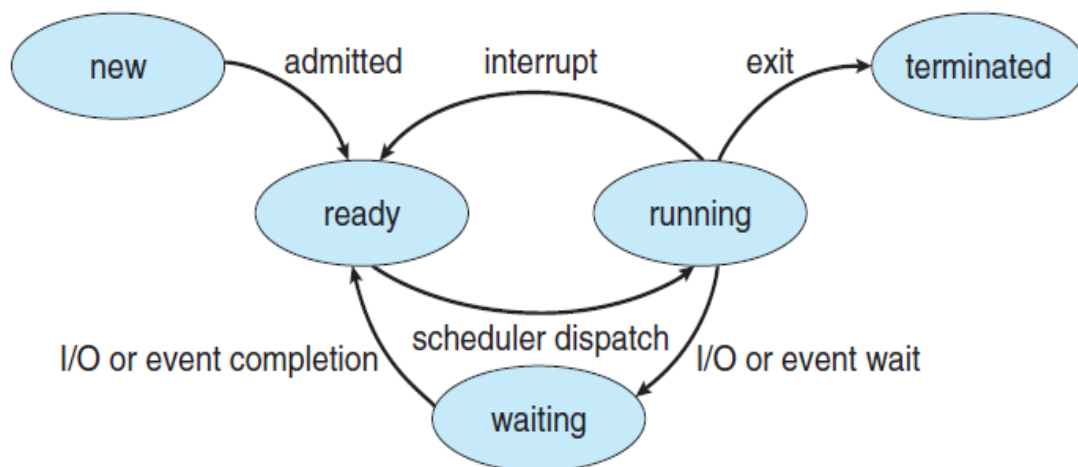
2   What are the various types of schedulers? Explain in detail.

There are three types of schedulers

1. Long term scheduler
2. Medium term scheduler
3. Short term scheduler

1. Long term scheduler

➢ It is a first level scheduler.
➢ It can be found in operating systems where there is support for 'batch', like as batch operating system.
➢ It works with the batch queue.
➢ It selects the next batch job/ process to be executed. Such process is loaded in main memory and waits in ready state for CPU to become free.
➢ It executes much less frequently compared to other schedulers.
➢ It controls the degree of multi programming, i.e. the total number of processes residing in the main memory.

2. Medium-term scheduler:

➢ It is a second level scheduler.
➢ It can be found in operating systems where there is support for swapping.
➢ It swaps-in and swap-out processes between main memory and disk.

➢ It also controls the degree of multi programming i.e. the total number of processes residing in the main memory.

3. Short term scheduler:
➢ It is a third level scheduler.
➢ It can be found always in all modern operating Systems.
➢ It works with the ready queue.
➢ It selects the next process to be executed by CPU whenever more than one process is simultaneously in the Ready state.
➢ It controls the Ready-to-Running state transitions in process life cycle.
➢ It executes much frequently compared to another scheduler.

3   Difference between Program and process.

Program	Process
➢ Program contains a set of instructions designed to complete a specific task.	➢ Process is an instance of an executing program.
➢ Program is a passive entity as it resides in the secondary memory.	➢ Process is a active entity as it is created during execution and loaded into the main memory.
➢ Program exists at a single place and continues to exist until it is deleted.	➢ Process exists for a limited span of time as it gets terminated after the completion of task.
➢ Program is a static entity.	➢ Process is a dynamic entity.
➢ Program does not have any resource requirement; it only requires memory space for storing the instructions.	➢ Process has a high resource requirement, it needs resources like CPU, memory address, I/O during its lifetime.
➢ Program does not have any control block.	➢ Process has its own control block called Process Control Block.

4   Explain Process control block.
➢ Operating System is responsible for various activities related to process management such as process creation, execution and termination.
➢ Operating System maintain a table, call process table, to store all the information about each process.
➢ Process table contains various entries one entry per process, call **process control block** or **task control block**.
➢ Each process contains some more information other than its address space. This information is stored in PCB as a collection of various fields.
➢ The various fields and information stored in these field are given below:

1. **Process Identifier**: It contains identifier, parent process identifier and user identifier.
2. **Process state**: It indicates the current state of a process such as new, ready, running, waiting and terminated.
3. **Program Counter**: It indicates the address of the next instruction to be executed for a process.
4. **Pointers**: It contains information about where the code, data and stack regions are stored in main memory.
5. **CPU registers**: Various CPU registers are used during execution of a process. They include accumulators, index registers, stack pointer etc. when context switch occurs, all information continue correctly afterward.
6. **CPU scheduling Information**: It includes process priority, pointer to various scheduling queues, information about events on which process is waiting and other scheduling parameters.
7. **Memory management Information**: It includes values of the base limit registers, information about page tables or segment tables.
8. **I/O Status information:** It includes the list of open files, I/O devices allocated to the process and so on.
9. **Accounting Information:** It includes the amount of CPU and real time used, time limits, account numbers and so on.

5   Define: i) Scheduling ii) Scheduler iii) Scheduling Algorithm.

i)   **Scheduling:** Scheduling is the process of making a choice of which process to run next whenever more than one processes are simultaneously in the Ready state.

ii)   **Scheduler:** Scheduler is an OS module which actually makes a choice of which process to run next whenever more than one processes are simultaneously in the ready state.

iii) **Scheduling Algorithm:** Scheduling Algorithm, which is used by the scheduler in making

6   What is context switch? Explain in brief.

➢ The Context switching is a technique or method used by the operating system to switch a process from one state to another to execute its function using CPUs in the system.

➢ When switching perform in the system, it stores the old running process's status in the form of registers and assigns the CPU to a new process to execute its tasks.

➢ While a new process is running in the system, the previous process must wait in a ready queue.

➢ The execution of the old process starts at that point where another process stopped it.

➢ It defines the characteristics of a multitasking operating system in which multiple processes shared the same CPU to perform multiple tasks without the need for additional processors in the system.

Example of Context Switching:

➢ Suppose that multiple processes are stored in a Process Control Block (PCB). One process is running state to execute its task with the use of CPUs.

➢ As the process is running, another process arrives in the ready queue, which has a high priority of completing its task using CPU.

➢ Here we used context switching that switches the current process with the new process requiring the CPU to finish its tasks. While switching the process, a context switch saves the status of the old process in registers.

➢ When the process reloads into the CPU, it starts the execution of the process when the new process stops the old process. If we do not save the state of the process, we have to start its execution at the initial level.

➢ In this way, context switching helps the operating system to switch between the processes, store or reload the process when it requires executing its tasks.

7   Define: i) CPU Utilization ii) Throughput iii) Turn-around Time iv) Waiting Time v) Response Time.

i) **CPU Utilization:** It is an average fraction of time during which CPU is busy. It ranges from 0 to 100%. In a real system it should be between 40 to 90%. CPU should remain as busy possible or CPU utilization should remain as high as possible.

ii) **Throughput:** Number of processes completed per time unit is called throughput.

iii) **Turn-around Time:** Time required to complete execution of a process is called turn-around time.

iv)**Waiting time:** It is total time duration spent by a process waiting in 'Ready' queue.

Waiting time = Turn Around Time- Actual Execution Time

v) **Response Time:** It is time between issuing a command/ request and getting output/ result.

8   Explain inter process communication with various issues and examples.

➢ 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.

Inter process communication
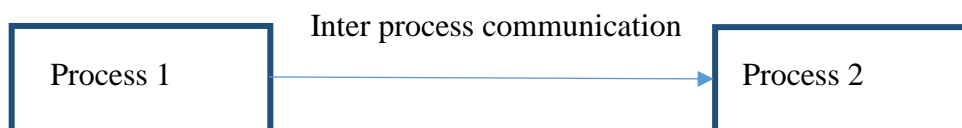
| Process 1 | → | Process 2 |

Examples

i) A shell pipeline in unix:

Here, output of one process is passes as input to the second process.

ii) Printing on printer in network:

Here, a process running on local machine communication to a process running on remote machine which contains a printer. One process requests for printing operation and other process does the job of printing to provide service.

iii) Chat or mail server:

Here, a process running on client machine communicates to a process running on server machine via internet.

Issues Related to IPC

i) How one process can pass information to another:

To communicate means to share some information or to pass information to others. One process can pass information to another by using shared memory, shared file, message passing, sockets etc.

ii) Two or more processes should not get into each other way:

Suppose one process is printing some document in printer. Meanwhile some other process comes which also wants to print something. But, it should wait till first process completes its printing.

iii) Proper sequencing should be maintained in execution of processes, when dependencies are present:

9 What is race condition? Explain mutual exclusion in brief.

- ➢ Race condition
- ➢ A race condition is a situation that may occur inside a critical section.
- ➢ This happens when the result of multiple thread execution in critical section differs according to the order in which the threads execute.
- ➢ Race conditions in critical sections can be avoided if the critical section is treated as an atomic instruction.
- ➢ Also, proper thread synchronization using locks or atomic variables can prevent race conditions.
- ➢ Example:

Process P0:	**Process P1:**
Read (A);	Read (A);
A: A-100;	A: A+200;
Write (A);	Write (A);

➢ Mutual exclusion
➢ A code segment of a process where the shared resource is accessed is referred critical section.
➢ Mutual exclusion is the way of making sure that if one process is executing critical section. Other processes will be excluded from executing critical section.
➢ Only one process should be allowed to access a shared resource at a time.

Mutual Exclusion in Critical Sections
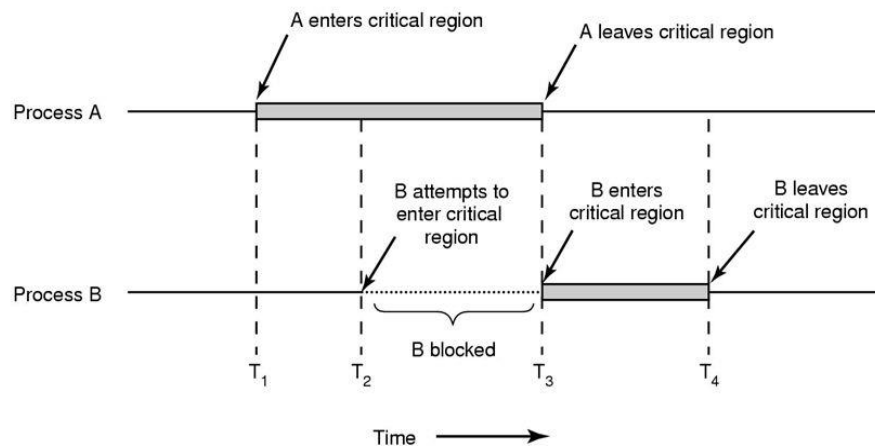
10   What is deadlock? Explain condition of deadlock.

➢ A set of processes is deadlocked, if each process in the set is waiting for an event that only another process in the set can cause.
➢ This event can be caused by some other process from the set. But none of them will cause any event, because all are waiting in waiting state.
➢ Conditions for deadlock
➢ There are four conditions of deadlock.
1. Mutual Exclusion
➢ Each resource can be assigned to exactly one process. If any process requests resource which is not free, then that process will wait until resource become free.
2. Hold and wait
➢ A process must be holding at least one resource, and waiting for additional resource. Also, such resources are currently being held by other processes.

7

3. No preemption
➢ Resources cannot be preempted. Once resources are granted, they cannot be forcibly taken away from a process. They must be explicitly released by the process holding them.
4. circular wait
➢ There must be a circular of two or more process. Each process is waiting for a resource which is held by the next member process of the chain.

11.   What is Semaphore? Explain in detail.

➢ A semaphore is a signalling mechanism and a thread that is waiting on a semaphore can be signalled by another thread.
➢ This is different than a mutex as the mutex can be signalled only by the thread that called the wait function.
➢ A semaphore uses two atomic operations, wait and signal for process synchronization.
➢ The wait operation decrements the value of its argument S, if it is positive. If S is negative or zero, then no operation is performed.

```
wait(S){
  while (S<=0);
      sleep();
  S--;
}
```

➢ The signal operation increments the value of its argument S.

```
signal(S){
  S++;
  Wakeup();
}
```

Chapter-3 Memory Management

1. **List out Memory allocation methods.**

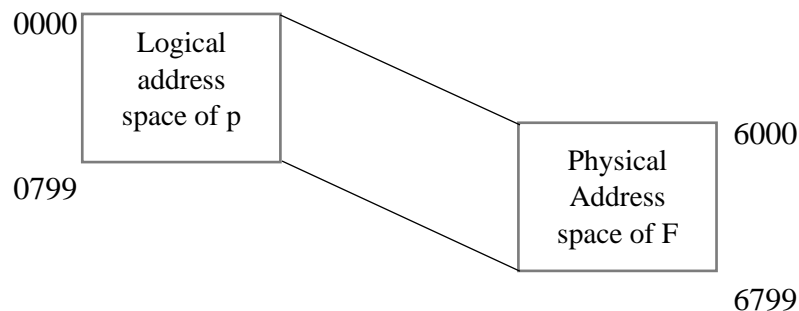   1. Contiguous Memory Allocation

    ➤ Single Process Monitor

    ➤ Multiprogramming with Fixed partitioning

    ➤ Multiprogramming with Dynamic partitioning

    2. Non-Contiguous Memory Allocation

    ➤ Paging

    ➤ Segmentation

2. **Explain Contiguous memory allocation method.**

    ➤ This is simple and old method allocation. It is not used in modern operating systems.

    ➤ Each process occupies a block of contiguous memory locations in main memory.

    ➤ Entire process is kept together in a contiguous section of memory.

    ➤ When a process is brought in memory, a memory is searched to find out a chunk of free memory having enough size to hold a process. Once such chunk is found required memory is allocated.

    ➤ If a contiguous memory space of the required size is not available in the main memory, the process is made to wait until contiguous space of the required size is available.

    ➤ Logical address space is not divided into any partitions. Also physical address space will be contiguous, without any gaps.
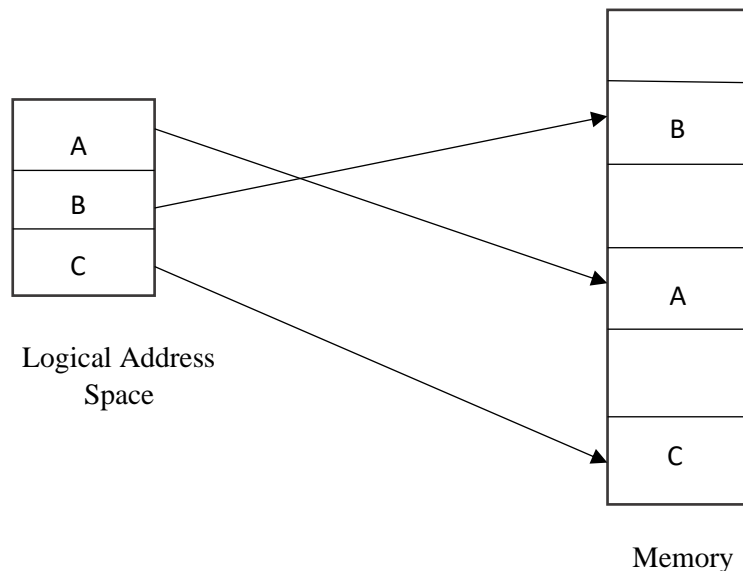
Advantages

➢ It is easy to implement and understand.

Disadvantages

➢ Having poor memory utilization.

3. **Explain Non- Contiguous memory allocation method.**

➢ This method is used by most modern operating systems.

➢ Logical address space of a process is divided into partitions. And for each partition contiguous chunk of free memory is allocated.

➢ Physical address space will not be contiguous now.

➢ Logical address space of a process is divided into three partitions (A, B and C) and each partition is allocated separate chunk of memory in physical memory.

Logical Address
Space

Memory

Advantages

➢ Having better memory utilization.

Disadvantages

➢ It is complex to implement.

4. **Explain Multiprogramming with Fixed (static) Partitions.**

➢ This method allows multiple processes to execute simultaneously.

➢ Size is fixed by manufacturers.

➢ Memory is shared among operating systems and various simultaneously running processes.

> ➢ Multiprogramming increases CPU utilization.
>
> ➢ Here, memory is divided into fixed-size partitions. Size can be equal or unequal for different partitions. Generally, unequally sized partitions are used for better utilization of memory.
>
> ➢ Each partition can accommodate exactly one process. Means only single process can be placed in one partition.
>
> ➢ Whenever any program needs to be loaded in memory, a free partition big enough to hold the program is found. This partition will be allocated to that program.
>
> ➢ If there is no free partition available of required size, then that process need to wait. Such process will be put in a queue.
>
> ➢ There are two ways possible
>
>   1. Using Multiple Input Queue
>   2. Using Single Input Queue

Partition 4
Partition 3
Partition 2
Partition 1
Operating System
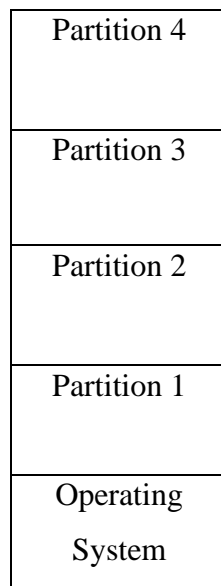
5. Explain Multiprogramming with Dynamic Partition.

> ➢ This method also has multiple processes to execute simultaneously.
>
> ➢ Memory is shared among operating systems and various simultaneously running processes.
>
> ➢ Memory is not divided into any fixed size partitions. Also the number of partitions is not fixed. Process is allocated exactly as much memory as it requires.

> ➢ Whenever any process enter in a system, a chunk of free memory big enough to fit the process is found and allocated.

> ➢ If enough free memory is not available to fit the process, process needs to wait until required memory becomes available.

OS 8M	OS	OS
56M	Process1 20M	Process1 20M
		Process 2 14M

Advantages:

1. Better utilization of memory.
2. There is no internal fragmentation
3. Degree of multiprogramming is not fixed here.

Disadvantages:

1. External fragmentation is possible

6.   What is Fragmentation? Explain in brief.

> ➢ Fragmentation refers to the unused memory that cannot be allocated to any process.

> ➢ There is free memory available, it cannot be used.

> ➢ Fragmentation is an unwanted problem in the operating system in which the processes are loaded and unloaded from memory, and free memory space is fragmented.

> ➢ Processes can't be assigned to memory blocks due to their small size, and the memory blocks stay unused.

> ➢ It is also necessary to understand that as programs are loaded and deleted from memory, they generate free space or a hole in the memory.

4

- ➢ These small blocks cannot be allotted to new arriving processes, resulting in inefficient memory use.
- ➢ There are two different kinds of fragmentations
1. External fragmentation
2. Internal fragmentation

1. External fragmentation

- ➢ It refers to the wastage of free memory between partitions, caused by scattered non-contiguous free space.
- ➢ This is a severe problem in contiguous memory allocation method with dynamic partitioning.
- ➢ Memory allocation and de-allocation operations eventually result in small holes in the memory.
- ➢ But, as memory is allocated contiguously here, these holes can't be used. This type of memory wastage is called external fragmentation.

2. Internal fragmentation

- ➢ It refers to the wastage of free memory within a partition, caused by the difference between the size of a partition and size of a process loaded.

7.   Explain Memory Relocation and protection.
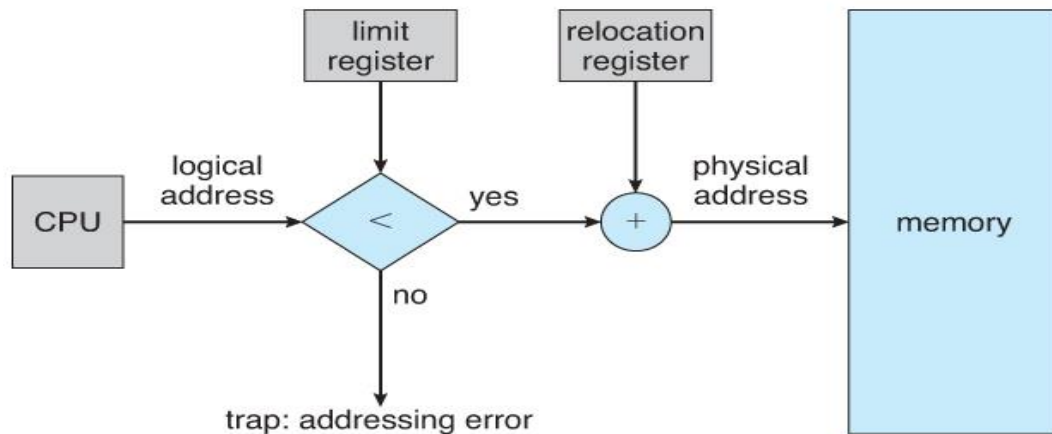
Memory Relocation

- ➢ A process can be loaded in any partition of main memory. Or a process can be loaded at any location in main memory.
- ➢ Address in logical address space and physical address space will not be same here.
- ➢ Logical address specify the locations of instructions and data within a process address space while physical address specify the actual locations in main memory.
- ➢ Physical addresses are required to actually fetch information.
- ➢ There is a reference to any logical address, it should be converted to physical address.
- ➢ This problem is called Memory Relocation.

Memory Protection

- ➢ Multiprogramming allows more than on process to run simultaneously.
- ➢ Memory is shared among various processes as well as operation system.

 ➢ All concurrent processes will be in memory at the same time. These processes should not have access to unauthorized information of other processes. Each process should read and write data belonging to that process only.

 ➢ This problem is called Memory Protection.

 ➢ Here pair of registers is used. These registers are called limit register and base register.

 ➢ Limit register is used to store the size of the process. Base register is used to store the staring location of process in main memory. Whenever any process is loaded in main memory its staring location and size are stored in these two registers respectively.

 ➢ CPU generated logical address start from 0 and goes up to the size of the process.

8. Explain paging.

 ➢ Logical address space of a process is divided into partitions. For each partition contiguous chunk of free memory is allocated in physical memory.

 ➢ Physical Address (represented in bits): An address actually available on memory unit

 ➢ Physical Address Space (represented in words or bytes): The set of all physical addresses corresponding to the logical addresses

 Address generated by CPU is divided into

 ➢ Page number(p): Number of bits required to represent the pages in Logical Address Space or Page number

 ➢ Page offset(d): Number of bits required to represent particular word in a page or page size of Logical Address Space or word number of a page or page offset.

Physical Address is divided into

➢ Frame number(f): Number of bits required to represent the frame of Physical Address Space or Frame number.

➢ Frame offset (d): Number of bits required to represent particular word in a frame or frame size of Physical Address Space or word number of a frame or frame offset.

Paging

Logical Address   Physical Address

CPU → P | d   f | d → Physical Memory

10000..00
Physical Memory
11111..11

P {

f
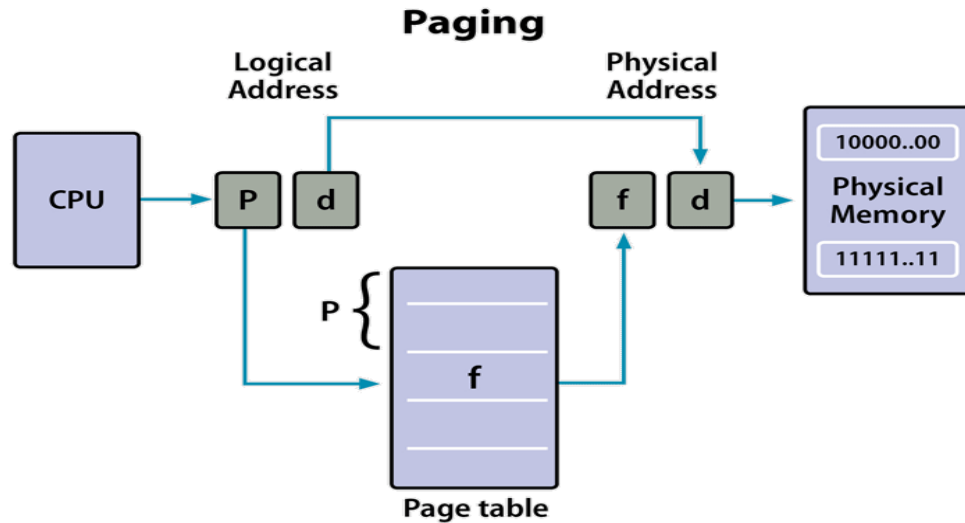
Page table

➢ A table called page table, is used to implement paging. When a process is to be executed, its pages are moved to free frames in physical memory.

➢ The information about frame number, where a page is stored, is kept in this page table.

➢ During the process execution a CPU generates a logical address to access instruction or data from a particular location.

➢ The page number is used to search the page table. This frame number indicates the actual frame on physical memory in which the page is stored.

9.   Explain Segmentation.

➢ In Operating Systems, Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a segment which can be allocated to a process.

➢ The details about each segment are stored in a table called a segment table. Segment table is stored in one (or many) of the segments.
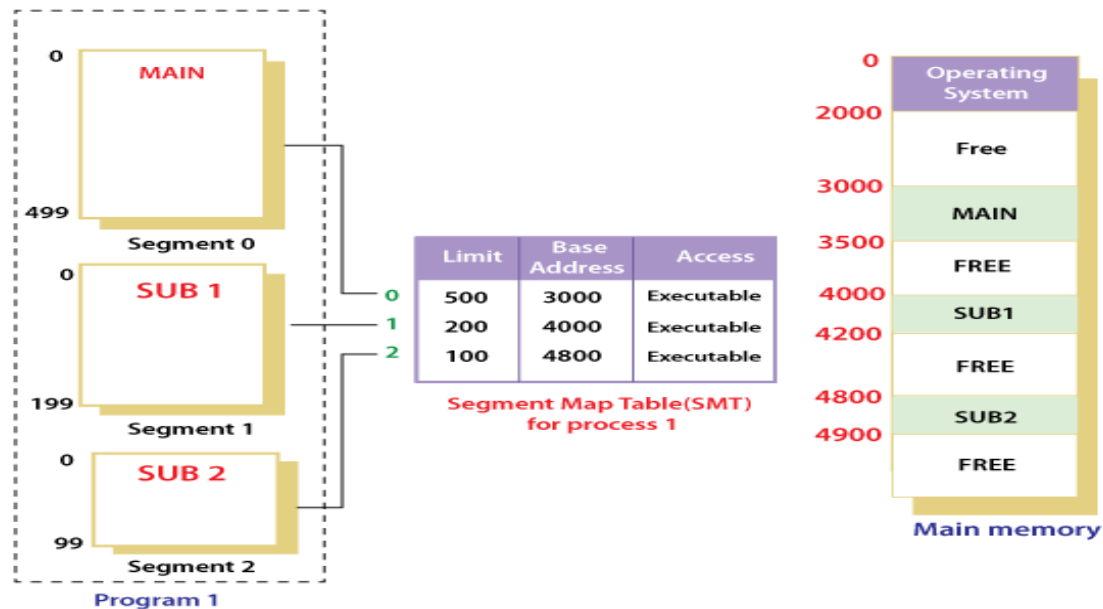
Segment table contains mainly two information about segment:

1. Base: It is the base address of the segment

2. Limit: It is the length of the segment.

CPU generates a logical address which contains two parts:

1. Segment Number

2. Offset

Limit	Base Address	Access
0	500	3000	Executable
1	200	4000	Executable
2	100	4800	Executable

Segment Map Table(SMT) for process 1

> The operating system can easily translate a logical address into physical address on execution of a program.

> The **Segment number** is mapped to the segment table. The limit of the respective segment is compared with the offset. If the offset is less than the limit then the address is valid otherwise it throws an error as the address is invalid.

> In the case of valid addresses, the base address of the segment is added to the offset to get the physical address of the actual word in the main memory.

Advantages of Segmentation

1. No internal fragmentation

2. Average Segment Size is larger than the actual page size.

3. Less overhead

4. It is easier to relocate segments than entire address space.

5. The segment table is of lesser size as compared to the page table in paging.

Disadvantages

1. It can have external fragmentation.

2. It is difficult to allocate contiguous memory to variable sized partition.

3. Costly memory management algorithms.

10.   Write a short note on Translation Look-aside Buffer.

➢ A translation lookaside buffer (TLB) is a memory cache that is used to reduce the time taken to access a user memory location.

➢ It is a part of the chip's memory-management unit (MMU). The TLB stores the recent translations of virtual memory to physical memory and can be called an address-translation cache.

➢ A Translation look aside buffer can be defined as a memory cache which can be used to reduce the time taken to access the page table again and again.

➢ It is a memory cache which is closer to the CPU and the time taken by CPU to access TLB is lesser then that taken to access main memory.

➢ In other words, we can say that TLB is faster and smaller than the main memory but cheaper and bigger than the register.

➢ TLB follows the concept of locality of reference which means that it contains only the entries of those many pages that are frequently accessed by the CPU.

➢ In translation look aside buffers, there are tags and keys with the help of which, the mapping is done.

➢ TLB hit is a condition where the desired entry is found in translation look aside buffer. If this happens then the CPU simply access the actual location in the main memory.

➢ However, if the entry is not found in TLB (TLB miss) then CPU has to access page table in the main memory and then access the actual frame in the main memory.

➢ Therefore, in the case of TLB hit, the effective access time will be lesser as compare to the case of TLB miss.
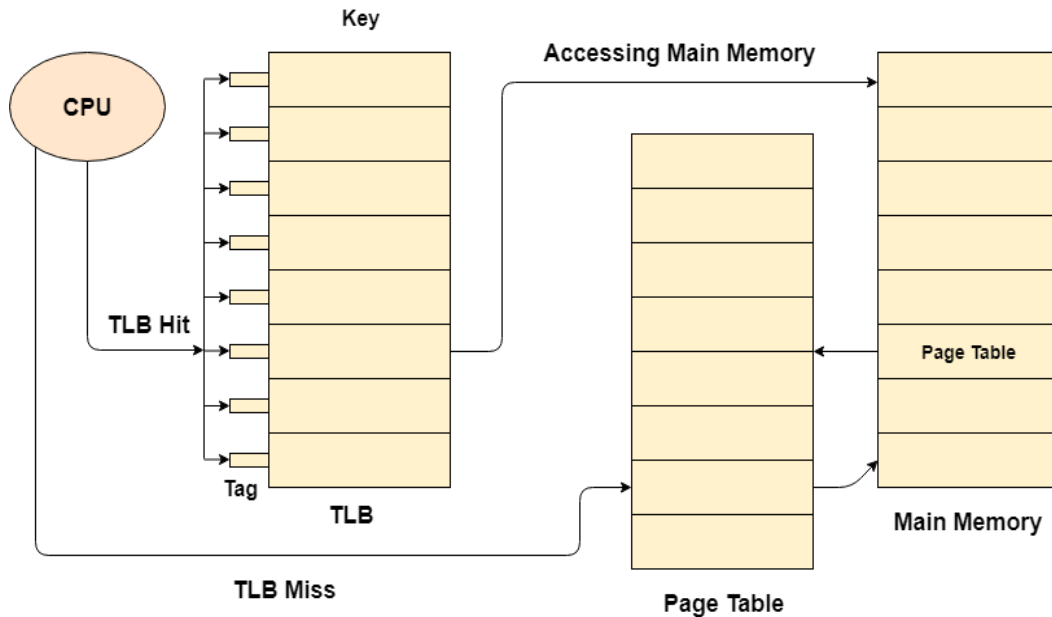
11.   What is swapping?

➢ Swapping is a memory management scheme in which any process can be temporarily swapped from main memory to secondary memory so that the main memory can be made available for other processes.

➢ It is used to improve main memory utilization. In secondary memory, the place where the swapped-out process is stored is called swap space.

➢ The purpose of the swapping in operating system is to access the data present in the hard disk and bring it to RAM so that the application programs can use it.

➢ The thing to remember is that swapping is used only when data is not present in RAM.

➢ Although the process of swapping affects the performance of the system, it helps to run larger and more than one process.

➢ This is the reason why swapping is also referred to as memory compaction.

➢ The concept of swapping has divided into two more concepts: Swap-in and Swap-out.

➢ Swap-out is a method of removing a process from RAM and adding it to the hard disk.

➢ Swap-in is a method of removing a program from a hard disk and putting it back into the main memory or RAM.

Swapping of Processes
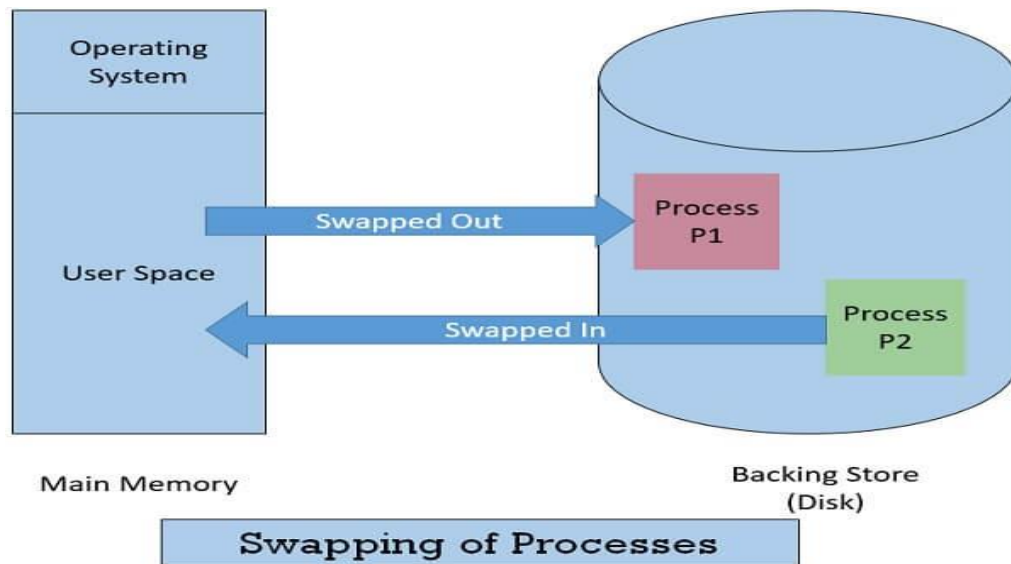
Advantages of Swapping

1. It helps the CPU to manage multiple processes within a single main memory.

2. It helps to create and use virtual memory.

3. Swapping allows the CPU to perform multiple tasks simultaneously. Therefore, processes do not have to wait very long before they are executed.

4. It improves the main memory utilization.

Disadvantages of Swapping

1. If the computer system loses power, the user may lose all information related to the program in case of substantial swapping activity.

2. If the swapping algorithm is not good, the composite method can increase the number of Page Fault and decrease the overall processing performance.

12. What is Virtual Memory? Explain Demand paging.

➢ A Virtual memory is a technic that allows a process to execute even though it is partially loaded in main memory.

➢ Virtual Memory is a storage scheme that provides user an illusion of having a very big main memory. This is done by treating a part of secondary memory as the main memory.

➢ In this scheme, User can load the bigger size processes than the available main memory by having the illusion that the memory is available to load the process.

➢ Instead of loading one big process in the main memory, the Operating System loads the different parts of more than one process in the main memory.

➢ By doing this, the degree of multiprogramming will be increased and therefore, the CPU utilization will also be increased.

➢ Demand Paging

➢ Demands paging system is similar to paging with swapping.

➢ Processes are kept on secondary storage, i.e on disk.

➢ To execute s process, it is swapped into the memory. But, rather swapping the entire process into memory, only pages, which are required for execution, are swapped.

➢ Page table includes the valid-invalid bit for each page entry. If it is set to valid, it indicates that the page is currently present in the memory. If it is set to invalid, it indicates that the page is still not loaded in the memory.

➢ When a process starts its execution this is set invalid for all the entries in the page table.

➢ When a page is loaded in memory, its corresponding frame number is entered and page validity bit is set to valid.

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

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

➢ With each page table entry, a valid-invalid bit is associated (where 1 indicates in the memory and 0 indicates not in the memory)

➢ Initially, the valid-invalid bit is set to 0 for all table entries.
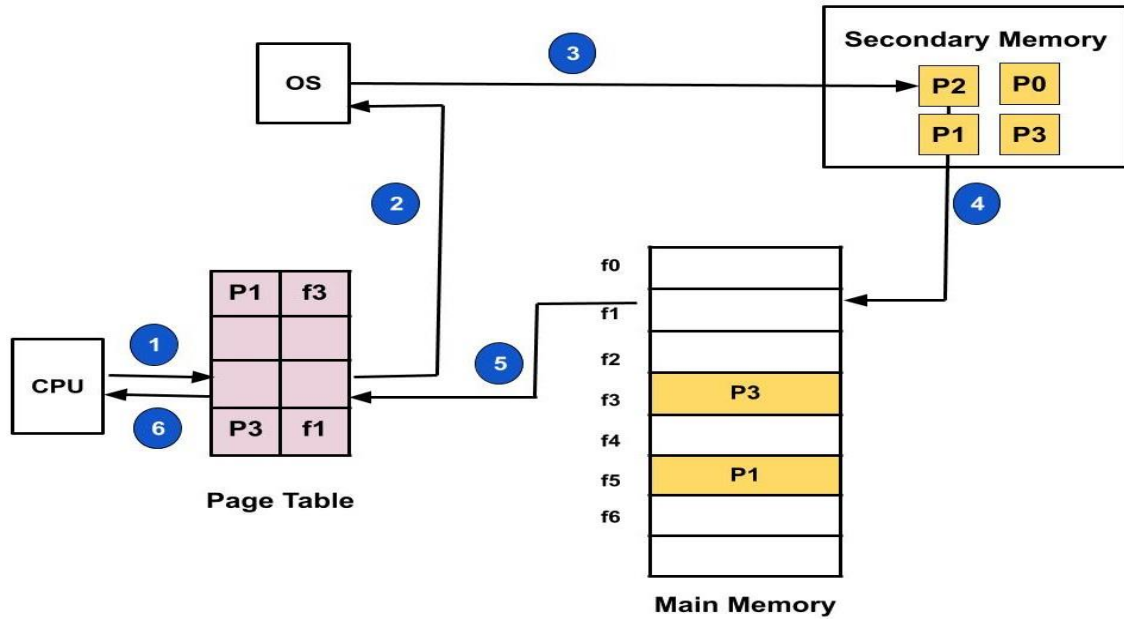
- ➢ If the bit is set to "valid", then the associated page is both legal and is in memory.
- ➢ If the bit is set to "invalid" then it indicates that the page is either not valid or the page is valid but is currently not on the disk.
- ➢ For the pages that are brought into the memory, the page table is set as usual.
- ➢ But for the pages that are not currently in the memory, the page table is either simply marked as invalid or it contains the address of the page on the disk.

Chapter-4 File Management

1. **Explain physical structure of hard disk.**

Moving-head Disk Mechanism
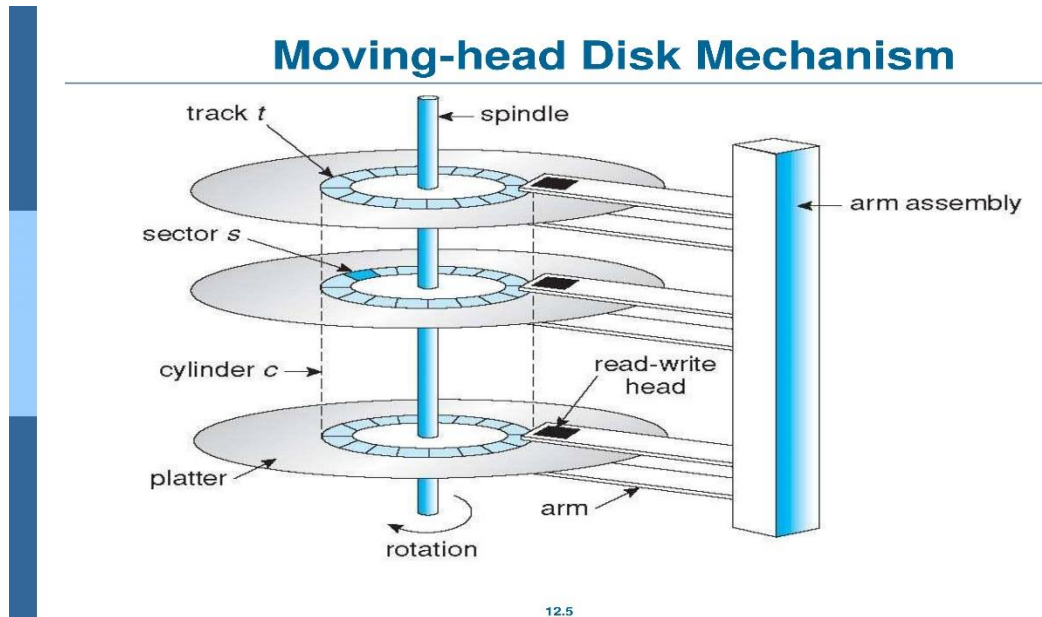
- A hard disk is a sealed unit.
- It contains one or more circular platters.
- Each platter contain magnetic coating on both of its surface.
- The platter are stacked one on top of another. They rotate together round a central spindle.
- Rotation speed is normally 3600, 5200 or 7200 rpm.
- Disk contain 1 to 8 platters. Each platter contains two surface.
- Data is read from and written to these platters using a number of read/write heads.
- Each of these head is attached to an arm. An arm can be moved in order to access different parts of the platter.
- A platter can be thought as a collection of circular, concentric, flat rings. These rings are called tracks.
- Each track is divided into fix size blocks called sectors. Each sector is of 512 bytes and there are hundreds of sectors on a single track.

➢ A group of tracks with the same radius are called cylinders. Number of cylinders is same as that of tracks.

2. What are file attributes? Describe various file attributes.

➢ Every file has its name and data. Such properties are called file attributes.

1. Name

➢ Every file carries a name by which the file is recognized in the file system. One directory cannot have two files with the same name.

2. Identifier

➢ Along with the name, Each File has its own extension which identifies the type of the file. For example, a text file has the extension **.txt,** A video file can have the extension **.mp4.**

3. Type

➢ In a File System, the Files are classified in different types such as video files, audio files, text files, executable files, etc.

4. Location

➢ In the File System, there are several locations on which, the files can be stored. Each file carries its location as its attribute.

5. Size

➢ The Size of the File is one of its most important attribute. By size of the file, we mean the number of bytes acquired by the file in the memory.

6. Protection

➢ The admin of the computer may want the different protections for the different files. Therefore each file carries its own set of permissions to the different group of Users.

7. Time & Date

➢ Every file carries a time stamp which contains the time and date on which the file is last modified.

8. Usage Count

➢ This value indicates the number of processes that are currently using this file.

3. Explain file operations.

1. Create operation:

➢ This operation is used to create a file in the file system.

➢ It is the most widely used operation performed on the file system. To create a new file of a particular type the associated application program calls the file system.

➢ This file system allocates space to the file. As the file system knows the format of directory structure, so entry of this new file is made into the appropriate directory.

2. Open operation:

➢ This operation is the common operation performed on the file. Once the file is created, it must be opened before performing the file processing operations.

➢ When the user wants to open a file, it provides a file name to open the particular file in the file system.

➢ It tells the operating system to invoke the open system call and passes the file name to the file system.

3. Write operation:

➢ This operation is used to write the information into a file.

➢ A system call write is issued that specifies the name of the file and the length of the data has to be written to the file.

➢ Whenever the file length is increased by specified value and the file pointer is repositioned after the last byte written.

4. Read operation:

➢ This operation reads the contents from a file. A Read pointer is maintained by the OS, pointing to the position up to which the data has been read.

5. Seek operation:

➢ The seek system call re-positions the file pointers from the current position to a specific place in the file i.e. forward or backward depending upon the user's requirement.

➢ This operation is generally performed with those file management systems that support direct access files.

6. Delete operation:

➢ Deleting the file will not only delete all the data stored inside the file it is also used so that disk space occupied by it is freed. In order to delete the specified file the directory is searched. When the directory entry is located, all the associated file space and the directory entry is released.

7. Close operation:

➢ When the processing of the file is complete, it should be closed so that all the changes made permanent and all the resources occupied should be released.

➢ On closing it deallocates all the internal descriptors that were created when the file was opened.

8. Append operation:

This operation adds data to the end of the file.

9. Rename operation:

➢ This operation is used to rename the existing file.

4. Explain file types.

A file is a collection of related information that is recorded on secondary storage. Or file is a collection of logically related entities. From user's perspective a file is the smallest allotment of logical secondary storage.

➢ **Character special files −** data is handled character by character as in case of terminals or printers.

➢ **Block special files −** data is handled in blocks as in the case of disks and tapes.

➢ **Regular files**- Contains user information.

➢ **Directories**- They are contains for files and other sub-directories.

5. Explain various directory structure with their advantages and disadvantages.

1. Single-Level Directory

➢ Single-Level Directory is the easiest directory structure. There is only one directory in a single-level directory, and that directory is called a root directory.

➢ In a single-level directory, all the files are present in one directory that makes it easy to understand. In this, under the root directory, the user cannot create the subdirectories.
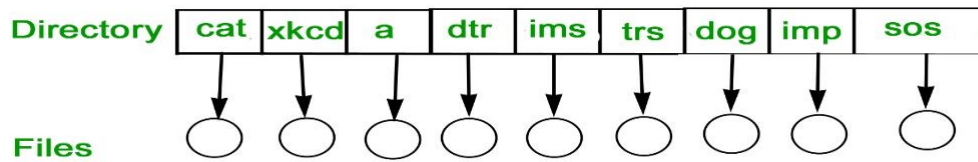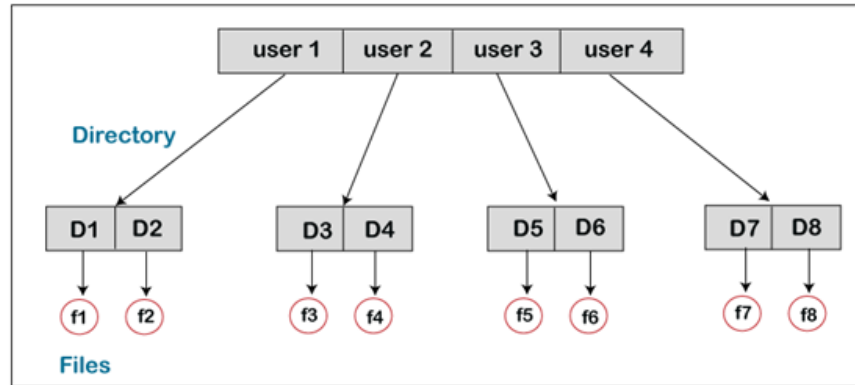
Advantages:

➢ The implementation of a single-level directory is so easy.

➢ In a single-level directory, if all the files have a small size, then due to this, the searching of the files will be easy.

➢ In a single-Level directory, the operations such as searching, creation, deletion, and updating can be performed.

Disadvantages:

➢ If the size of the directory is large in Single-Level Directory, then the searching will be tough.

➢ In a single-level directory, we cannot group the similar type of files.

➢ Another disadvantage of a single-level directory is that there is a possibility of collision because the two files cannot have the same name.

➢ The task of choosing the unique file name is a little bit complex.

2. Two-Level Directory

➢ Two-Level Directory is another type of directory structure. In this, it is possible to create an individual directory for each of the users.

➢ There is one master node in the two-level directory that include an individual directory for every user.

➢ At the second level of the directory, there is a different directory present for each of the users. Without permission, no user can enter into the other user's directory.
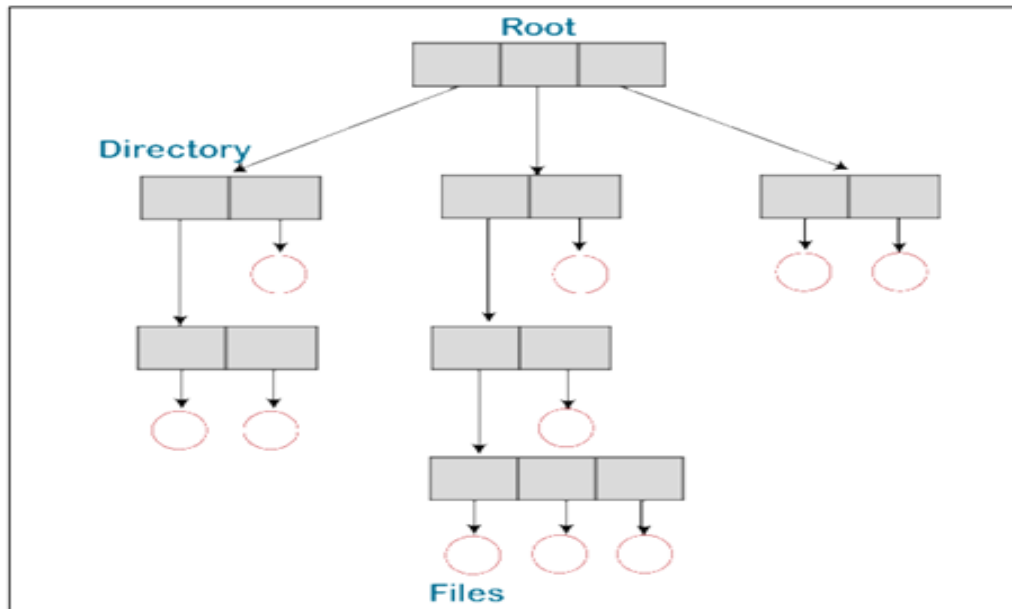
Advantages:

➢ In the two-level directory, various users have the same file name and also directory name.

➢ Because of using the user-grouping and pathname, searching of files are quite easy.

Disadvantages:

➢ The disadvantages of the two-level directory are:

➢ In a two-level directory, one user cannot share the file with another user.

➢ Another disadvantage with the two-level directory is it is not scalable.

3. Tree-Structured Directory

➢ A Tree-structured directory is another type of directory structure in which the directory entry may be a sub-directory or a file.

➢ The tree-structured directory reduces the limitations of the two-level directory. We can group the same type of files into one directory.

➢ In a tree-structured directory, there is an own directory of each user, and any user is not allowed to enter into the directory of another user.
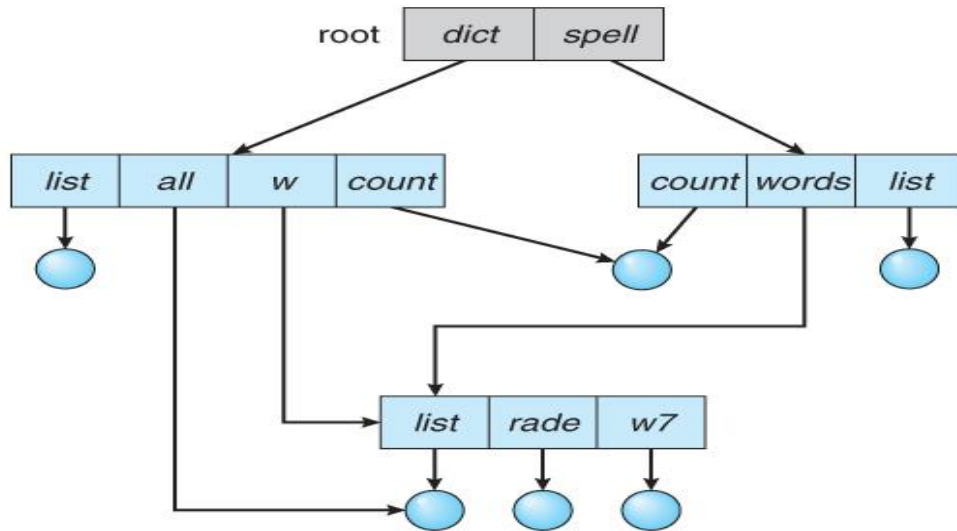
Advantages

➢ The tree-structured directory is very scalable.

➢ In the tree-structures directory, the chances of collision are less.

➢ In the tree-structure directory, the searching is quite easy because, in this, we can use both types of paths, which are the absolute path and relative path.

Disadvantages

➢ In the tree-structure directory, the files cannot be shared.

➢ Tree-structure directory is not efficient because, in this, if we want to access a file, then it may go under multiple directories.

➢ Another disadvantage of the tree-structure directory is that each file does not fit into the hierarchal model. We have to save the files into various directories.

4. Acyclic-Graph Directory

➢ In the acyclic-graph directory, more than one directory can point to a similar file or subdirectory. We can share those files among the two directory entries.

➢ With the help of aliases, and links, we can create this type of directory graph. We may also have a different path for the same file. Links may be of two kinds, which are hard link (physical) and symbolic (logical).
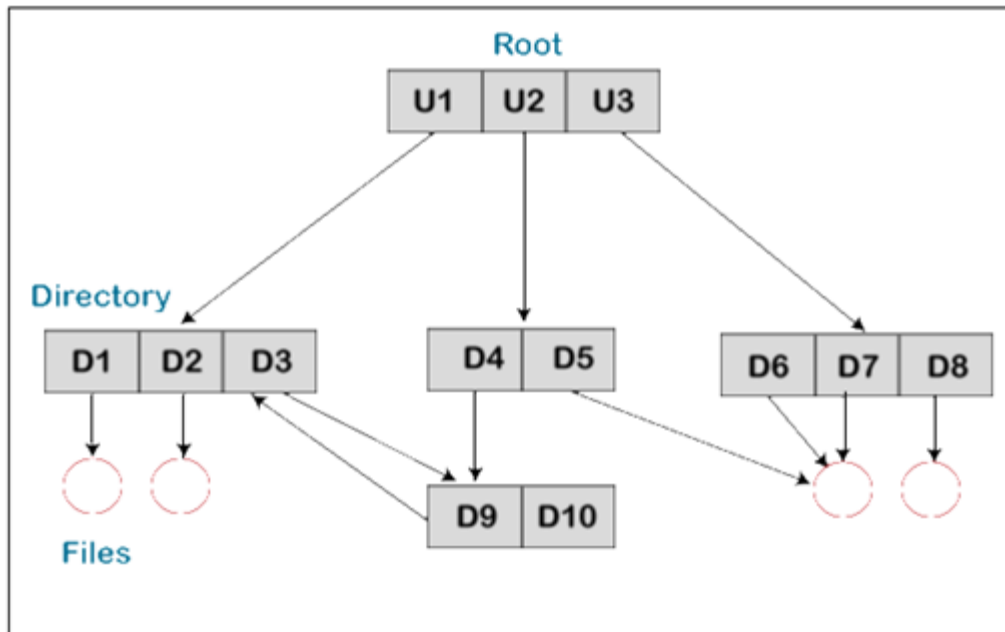
Advantages:

➢ In the acyclic-graph directory, the sharing of files is possible.

➢ In the acyclic-graph directory, because of different-different paths, searching is easy.

Disadvantages:

➢ If the files are shared through linking, there may be a problem in the case of deleting.

➢ If we are using softlink, then in this case, if the file is deleted then there is only a dangling pointer which is left.

➢ If we are using hardlink, in this case, when we delete a file, then we also have to remove all the reference connected with it.

5. General-Graph Directory

➢ The General-Graph directory is another vital type of directory structure. In this type of directory, within a directory we can create cycle of the directory where we can derive the various directory with the help of more than one parent directory.

➢ The main issue in the general-graph directory is to calculate the total space or size, taken by the directories and the files.

Advantages:

➢ The General-Graph directory is more flexible than the other directory structure.

➢ Cycles are allowed in the general-graph directory.

Disadvantages:

➢ In general-graph directory, garbage collection is required.

➢ General-graph directory is more costly, among other directory structures.

6. **Explain file allocation method and explain each of them.**

The allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.

- Contiguous Allocation

- Linked Allocation

- Indexed Allocation

1. Contiguous Allocation

➢ In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: b, b+1, b+2,……b+n-1.

➢ This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file. The directory entry for a file with contiguous allocation contains

- Address of starting block
- Length of the allocated portion.

➢ The file 'mail' in the following figure starts from the block 19 with length = 6 blocks. Therefore, it occupies 19, 20, 21, 22, 23, 24 blocks.
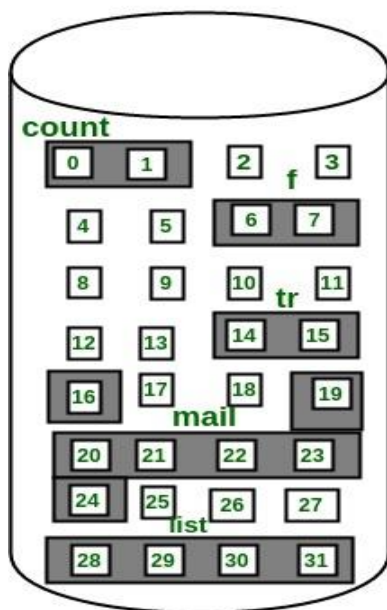
Directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Advantages:

➢ Both the Sequential and Direct Accesses are supported by this. For direct access, the address of the kth block of the file which starts at block b can easily be obtained as (b+k).

➢ This is extremely fast since the number of seeks are minimal because of contiguous allocation of file blocks.
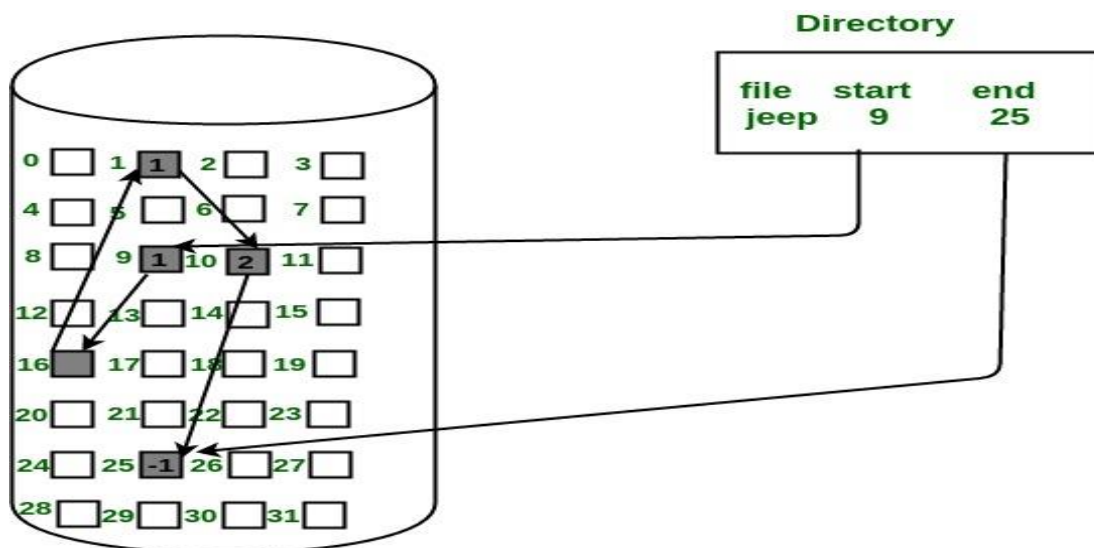
Disadvantages:

➢ This method suffers from both internal and external fragmentation. This makes it inefficient in terms of memory utilization.

10

➢ Increasing file size is difficult because it depends on the availability of contiguous memory at a particular instance.

2. Linked List Allocation

➢ In this scheme, each file is a linked list of disk blocks which need not be contiguous. The disk blocks can be scattered anywhere on the disk.

➢ The directory entry contains a pointer to the starting and the ending file block.

➢ Each block contains a pointer to the next block occupied by the file.

➢ The file 'jeep' in following image shows how the blocks are randomly distributed. The last block (25) contains -1 indicating a null pointer and does not point to any other block.

Advantages:

➢ This is very flexible in terms of file size. File size can be increased easily since the system does not have to look for a contiguous chunk of memory.

➢ This method does not suffer from external fragmentation. This makes it relatively better in terms of memory utilization.
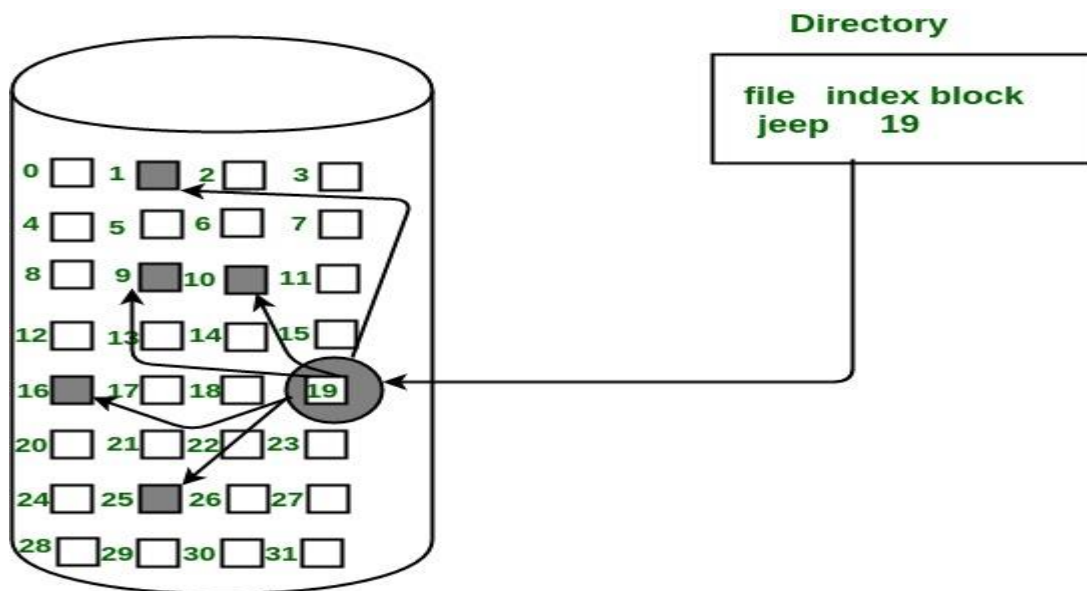
Disadvantages:

➢ Because the file blocks are distributed randomly on the disk, a large number of seeks are needed to access every block individually. This makes linked allocation slower.

- It does not support random or direct access. We cannot directly access the blocks of a file. A block k of a file can be accessed by traversing k blocks sequentially (sequential access) from the starting block of the file via block pointers.
- Pointers required in the linked allocation incur some extra overhead.

3. Indexed Allocation

- In this scheme, a special block known as the Index block contains the pointers to all the blocks occupied by a file.
- Each file has its own index block.
- The entry in the index block contains the disk address of the with file block.

Advantages:

- This supports direct access to the blocks occupied by the file and therefore provides fast access to the file blocks.
- It overcomes the problem of external fragmentation.

Disadvantages:

- The pointer overhead for indexed allocation is greater than linked allocation.

7. **Explain File Security & Protection Mechanism.**

   In computer systems, a lot of user's information is stored, the objective of the operating system is to keep safe the data of the user from the improper access to the system. Protection can be provided in number of ways.

Types of Access:

The files which have direct access of the any user have the need of protection. The files which are not accessible to other users doesn't require any kind of protection. The mechanism of the protection provides the facility of the controlled access by just limiting the types of access to the file. Several different types of operations can be controlled:

- Read – Reading from a file.
- Write – Writing or rewriting the file.
- Execute – Loading the file and after loading the execution process starts.
- Append – Writing the new information to the already existing file, editing must be end at the end of the existing file.
- Delete – Deleting the file which is of no use and using its space for another data.

Access Control:

There are different methods used by different users to access any file. The general way of protection is to associate identity-dependent access with all the files and directories a list called access-control list (ACL) which specify the names of the users and the types of access associate with each of the user. The main problem with the access list is their length. If we want to allow everyone to read a file, we must list all the users with the read access. This technique has two undesirable consequences:

Constructing such a list may be tedious and unrewarding task, especially if we do not know in advance the list of the users in the system.

Previously, the entry of the any directory is of the fixed size but now it changes to the variable size which results in the complicates space management. These problems can be resolved by use of a condensed version of the access list. To condense the length of the access-control list, many systems recognize three classifications of users in connection with each file:

Owner – Owner is the user who has created the file.

Group – A group is a set of members who has similar needs and they are sharing the same file.

Others – In the system, all other users are under the category called others.

Password: The access to any system is also controlled by the password. If the use of password is random and it is changed often, this may be result in limit the effective access to a file.

The use of passwords has a few disadvantages:

- The size of password is very large so it is difficult to remember the large passwords.
- If one password is used for all the files, then once it is discovered, all files are accessible; protection is on all-or-none basis.

Chapter-5 Linux Basics

1. **Explain features/characteristics of Linux Operating System.**

    1.Free and Open-Source Software

    Open-source means Linux is available with its source code. And free means users have freedom to make change in source code according to their requirements. These modification versions can also be redistributed.

    Also, most of the Linux flavors are either totally free or costs very less compared to other Operating systems.

    2.Flexibility in usage:

    Linux can be used for high performance server application, desktop application, and embedded systems.

    Due to this flexibility, Linux can be found on wide variety of devices such as mobiles phones, tablet computers, networks routers, personal computers, video game consoles and even in super computers

    3.A multi-user System:

    Linux is multi-user Operating System. This means, it allows multiple users to work simultaneously on the same system.

    Different users can login from different machines into the same machine by using programs line 'TELENT'.

    4.A multi-tasking system:

    Linux is a multi-tasking operating System too.it allows multiple programs to run simultaneously.

    Among simultaneously running process, one process will be foreground process. User can interact with this process directly. while other processes process. User can interact directly. While other processes will be background processes. They execute in background without requiring user interaction.

    5.High Performance and Reliability

    Linux provides high performance with minimum requirements of hardware compared to another operating system.

No other operating system is more stable reliable than Linux, system crashes, hangs, vires attacks are most absent from the Linux world.

6.The building-block Approach

Linux uses the building-block approach to perform complex tasks.

It provides a few hundred commands each of which can perform one simple job.to perform complex tasks, such simple commands can be combined using pipes and filters. thus, the small-is-beautiful philosophy is implemented here.

7.Flexible interface

Linux supports both types of interfaces-GUI (graphical user interface) as well as CLI (Command Line Interface).

GUI makes task of user easy, and so makes Operating system user friendly. CLI provides more options and control to the user. For example, complex tasks can be performed by combing multiple commands or creating shell scrips.

User can choose any interface according to their convenience and expertise.

8.File System Support:

Linux supports a wide range of file system such as ext, ext2, ext3, ext4, XFS, JFS etc.

Along with these file systems, it also supports file system supported by other operating system such as NTFS, so that, Linux users can also access files managed by those operating Systems.

9.Programing Facility:

The Linux all the programming features such as variables, control structures, loops and so on. These features can be used to develop shell programs, called shell scripts. Such programs can be used to control and automate many of the system's functions.

10.On-line help:

Linux provides an on-line help facility for all the commands. For this purpose, it provides a command named 'man'.by using this command, user can have an instant help on any command, system call, or file format. Along with this, as Linux is a community driven operating system, many developers and distributors works on it and there is a vast support available on internet.

2.   Explain in brief: Linux directory structure
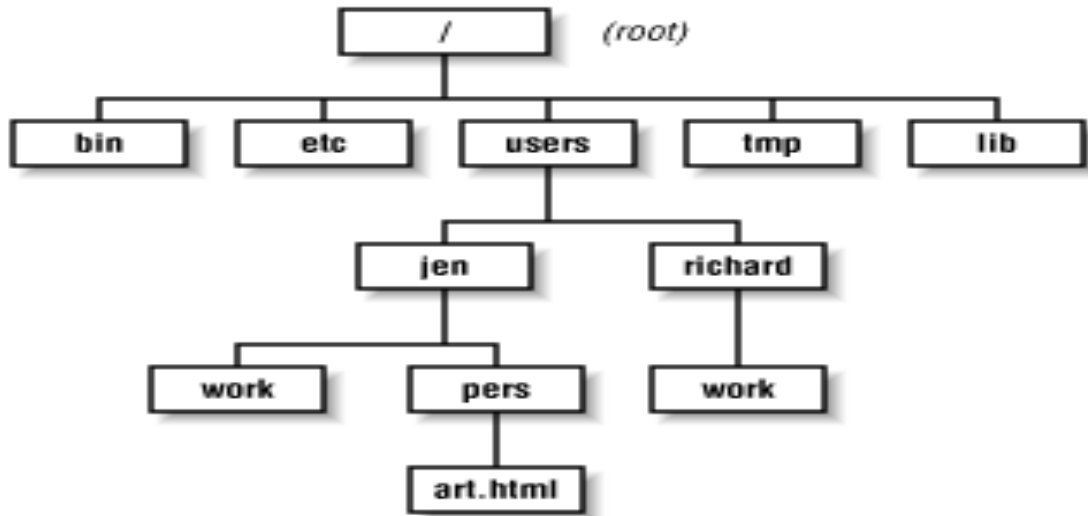
- ➢ A directory is container for other files and sub-directories.
- ➢ Directory are used to group and organize files & sub-directories. They provide a hierarchical file structure. They help in managing files in well-organized manner.
- ➢ In fact, a directory is file whose data is a sequence of entries. Each entry contains a file name and unique identification number called i-node number. This i-node number. This i-node number, in turn, provides a location of i-node where information about a file is stored.
- ➢ Whenever there is a need to access any file, the kernel maps the file name to its related i-node using directory entries based on the file path.
- ➢ Any directory contains two special entries: dot ('.') and double ('..'). these two entries refer to the current directory itself and its parent directory respectively.
- ➢ Linux uses a hierarchical directory structure similar to UNIX. Here all files are organized in an inverted tree like structure.
- ➢ The following figure shows this type of directory structure.it is also called a file system tree.
- ➢ The top of the hierarchy is called "root" directory, or simply a "root".it is denoted by "/". all the absolution path names start from a root ("/").

➢ Every non-leaf node in a tree structure is direction. Every leaf node is either a directory regular file or a device special file.'

➢ The name of a file is given by path name. every leaf node is either a directory, regular file or device special file.

➢ The name of a file is given by a path name. This path name describes the location of file in hierarchy.

➢ A path name can be either absolute or relative. Absolute path name starts from the root.

➢ For example, in above figure, an absolute path name for file art.html can be given as- /users/jen/pers/art.html

3. **Explain: kernel, shell and system call.**

Kernel:

➢ The kernel is the core of the Linux operating system. Kernel is a program, which is loaded in memory when system is turned on. It stays there and provides various services until the system is turned off.

➢ Linux uses monolithic, modular kernel. Device drivers can be loaded and unloaded into kernel in form of kernel modules.

➢ Kernel interacts with the hardware directly, When user program needs to use any hardware, it has to use services provided by the kernel. Special functions, called system calls, are used to request kernel. Kernel performs the job on behalf of the user process.

➢ In addition to providing services to user programs, kernel also provides other services like process management, memory management, file system management and so on. In short, kernel manages entire computer system.

Shell:

➢ The shell is an interface between the user program and the kernel.

➢ When user logs-in to the system, process for shell starts execution. It terminates when user logs-out from the system. enter Users can directly interact with the shell.

➢ It works as a command interpreter. It accepts commands from user and translates them into the form, which the kernel can understand easily.

➢ It is also a programming language. It provides various programming

4

System Call:

➢ System calls are special functions.

➢ They are used to request kernel to provide various services, such as reading from a file stored on hard disk.

➢ They can be invoked via library procedures, or via commands provided by shell, or even directly from C programs in Linux.

➢ System calls are similar to user-defined functions. Difference is that they execute in the kernel mode, having fully access to all the hardware; while user defined functions execute in user mode, having no direct access to the hardware.

4.   Explain Linux architecture or Linux layered structure.

➢ Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

➢ Kernel − Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.

➢ System Library − System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not require kernel module's code access rights.

➢ System Utility − System Utility programs are responsible to do specialized, individual level tasks.

➢ Hardware layer − Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).
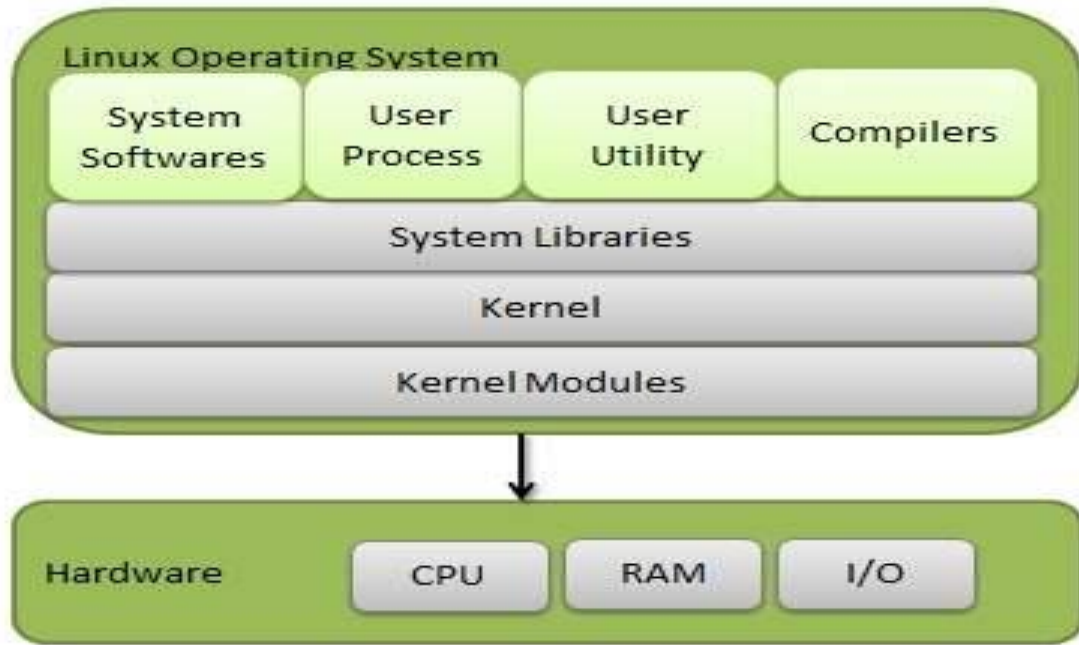
> ➢ Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each process and provides system services to processes, provides protected access to hardware to process.

> ➢ Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low-level tasks.

5. **List out directory command of Linux OS.**

> ➢ **pwd:** Print Working Directory
>
>   syntax: pwd
>
>   Usage: print working directory name.

> ➢ **cd:** Change Directory
>
>   syntax: cd[directory]
>
>   Usage: Change working directory
>
>   Example:
>
>   - cd   …change working directory to home directory

- cd..   … change working directory to parent directory

- cd   /usr/lib … … change working directory to absolute path/usr/lib

➢ **mkdir**: Make directory

 Syntax: mkdir directory

 Usage: Create new directory

 Examples: mkdir Test …create a new directory named Test

➢ **rmdir:** Remove Directory

Syntax: rmdir Directory

Usage: Removes an empty directory. If the directory is not empty, it will not be
        removed.

Examples: rmdir Test … removes directory named Test if it is empty

6.   Explain following command with example.

cal: calendar

Syntax: cal

Usage: Display calendar

date:

Syntax: date

Usage: Display current date and time

echo:

Syntax: echo[-n] message

Usage: Display message, as well as evaluates variables

Examples:

Echo "hello world"   …display message

passwd:

Syntax: passwd

Usage: changes password of user

who:

Syntax: who   …display all users

Usage: Display information about all user who have logged-in to the system currently

tty:

Syntax: tty

Usage: display filename associated with terminal

man:

Syntax: man[options] title

Usage:

print entries from the on-line reference manuals, find manual entries by keyword.

Example:

man ls …will display man-page(help) for ls command

head:

Syntax: head[-n] file

Usage: display top of the file.

Example: head test.txt

tail:

Syntax: tail[-n] file

Usage: display end of the file.

Example: tail test.txt

cut:

Syntax: cut -c list

　　　cut -f list

Usage: cutes a file vertically.

Example: cut -c 5-15,20-25 test.txt   ….extracts two columns having characters 5-15 and 20-25

paste:

Syntax: paste [-d char] file1 file2

Usage: pastes file

sort

Syntax: sort [options] file

Usage: Sorts data in file.

Example: sort test.txt

tr: translating characters

Syntax: tr [-cds][string1][string2] input

Usage: Translate character in file.it is used to replace characters, delete characters, changing case of text as well as compressing multiple consecutive characters.

Example: tr ':' '|'<text.txt   …replace all ':' from text.txt file with '|'

grep:

Syntax: grep[options] patten filename

Usage: Display lines from files that match the given patten.

Example: grep "hello" test.txt   ….simple pattern matching

ps:

Syntax: ps [options]

Usage: Display status of current running process.

time:

Syntax: time command/program

Usage: Executes command or program and display time usage on the terminal

Example: time date

kill:

Syntax: kill [-signal] pid

Usage: Send a signal to a process

Example: kill 125   …kill process having ID 125

expr:

Syntax: expr expression

Usage: Evaluates the given mathematical expression

Example: expr 5+3

tee:

Syntax: tee file input

Usage: splits input into two components. saves one component in a file and display the other to the standard output.

Example: who |tee user.txt   …output of the who command will be display on the terminal as well as stores in a file user.txt

Set:

Syntax: set arguments

Usage: Assigns positional parameters $1,$2 and so on, to its arguments.this is used to cut fields from a single lined input.

Example: set 'date'   …various fields of output of date command will be assigned to positional parameters, these fields can be accessed individually by using positional parameters.

7.   **List out file command of Linux OS.**

cat:

Syntax:

cat file ..displays file contents.

cat > file name   ..creates new file

cat >>file   … appends data to an existing file

Usage: display file contents, creates new file, or appends data to an existing file.

ls: List files

Syntax: ls [directory]

Usage: List out contents of a directory

Options:

a List out all files including hidden ones

d list only directories

F mark directories with /, executable files with *, symbolic links with @,and sockets with=

l long listing showing protection, number of links, owner, size and time of last modification

s size in kilobytes

cp: copy command

Usage: The cp command is used to copy a file or directory.

Syntax: cp <existing file name> <new file name>

Options:

-      i: Interactive copying

-      r: Recursive copying

mv: Move/Remove file

Syntax: mv <file name> <directory path>

Usage: The mv command is used to move a file or a directory form one location to another location

Example: mv file1 file2

rm: Remove file

Syntax: rm[-f][-i][-r]file

Usage: Removes files

options:

i: Interactive removal

r: Recursive removal

f: Forceful removal

wc:

Syntax: wc [file]

Usage: The wc command is used to count the lines, words, and characters in a file.

Example: wc marks.txt

chmod:

Syntax: chmod [ugoa…][-+=]perms…[,…] FILE...

Usage:Change file permissions

Examples: chmod 0-rwx test.out

cmp: Compare

Syntax: cmp file1 file2

Usage: Compares two files and gives location of first mismatch

diff:

Syntax: diff file1 file2

Usage: Compares two files, gives location of mismatch, as well as suggests changes to make two files identical.

comm: compare sorted files

Syntax: comm [-123] file1 file2

Usage: compares sorted files file1 and files 2 line by line.

8. **Add two number, or, find sum of given two numbers.**

echo "Enter two integers separated by white spaces between."

```
read a b                        #read input values.
sum=`expr $a + $b`          # perform addition
echo "Sum = $sum"              #display sum
```

9. **Read three integer values from user, and find the maximum value among them.**

```
echo "Enter three integers separated by white spaces between."
read a b c
if [ $a -gt $b -a $a -gt $c ]                #if(a>b && a>c)
then
max=$a                          #max =b
elif [ $b -gt $c ]                  # else if (b>c)
then
max=$a                              #max = b;
else                                #else
max=$c                          # max=c;
fi
echo "Maximum Number is: $max"              # Maximum is: max
```

10. **Find the sum of all the individual digits in a given 5-digit number.**
    (For example, the number is 23786, required sum is $2 + 3 + 7 + 8 + 6 = 26$.)

```
echo "Enter a 5 digit number"
read num
sum=0                       # sum = 0
while [ $num -ne 0 ]            #while (num != 0)
do                          # {
dig=`expr $num % 10`        #       dig = num % 10
sum=`expr $sum + $dig`      #       sum = sum + dig
num=`expr $num / 10`        #        num= num / 10
done                        #  }
echo "sum = $sum"              # display sum
```

11. **Reverse the digits of a given 5-digit number.**

(For example, if a number is 23786, output should display 68732.)

```
echo "Enter a 5 digit number"
read num
number=num                              #number num
ans=0                                   #ans=0
while [ $num -ne 0 ]                    #while (num != 0)
do                                       # {
    dig=`expr $num % 10`                    #dig=num % 10
    ans=`expr $ans \* 10 + $dig`            #ans = (ans*10) + dig
    num=`expr $num / 10`                    #num= num/10
done                                     #}
echo "Reverse of $number is $ans"       #display number and rev
```

12. **Find the factorial value of given input number.**

```
echo "Enter a number"
read num
fact=1        # fact=1
i=1           #i=1
while [ $i -le $num ]          #while(i <= num)
do                       # {
    fact=`expr $fact \* $i`         #fact=fact*i
    i=`expr $i + 1`                 # i=i+1
done                      #}
    echo "Factorial of $num is $fact"     # display result
```

13. **Generate and display Fibonacci series.**

```
echo "How many Fibonacci numbers do U want?"
read limit
a=0                      # initialize a with 0
b=1                      # initialize b with 1
i=3                      # i = 3 (counter)
```

```
echo -n $a                  #display a without newline character
echo -n " "                 #leave space between output
echo -n $b                   #display b without newline char
while [ $i -le $limit ]         #while (i <= limit)
do                          # {
c=`expr $a + $b`               #c=a+b
echo -n " "                 #leave space between output
echo -n $c                  #display c
a=$b                        #a = b
b=$c                        #b = c
i=`expr $i + 1`             # i=i+1
done                        # }
```

14. Concatenate two strings and find the length of the resultant string.

```
echo "Enter first string:"
read sl
echo "Enter second string:"
read s2
s3=$s1$s2       #concatenates two strings
len=`echo $s3 | wc -c`
len=`expr $len - 1`
echo "Concatenated string is $s3 of length $len "
```

15. Display all even numbers within given range.

```
echo "Enter starting number."
read n1
echo "Enter end number."
read n2
while [ $nl -ne $n2 ]       #while (n != n2)
do
   p=`expr $nl % 2`     #p=nl % 2
   if [ $p -eq 0 ]          # if (p == 0)
   then
```

14

```
        echo "$nl is Even"
    fi
  n1=`expr $n1 + 1`          #n1=n1 + 1
done
```

16. Find out number of characters, words and lines from a given file.

```
Echo "Enter the filename"
Read file
c=`cat $file | we -c`          #determine number of characters
w=`cat $file | we -w`          #determine number of words
l=`cat $file | we -l`          #determine number of lines
echo "Number of characters in $file is $c"
echo "Number of words in $file is $w"
echo "Number of lines in $file is $l"
```

17. Implement simple calculator, which can perform basic mathematical operations such as addition, subtraction, multiplication & division.

```
echo "enter n1"
read n1
echo "enter n2"
read n2
echo "enter operator [+:addition, -:subtraction, *:multiplication, /:division]"
read op
case $op in
   +) expr $n1 + $n2 ;;
   -) expr $n1 - $n2 ;;
   \*) expr $n1 \* $n2 ;;
   /) expr $n1 / $n2 ;;
esac
```