GOVT. POLYTECHNIC,DHENKANAL

**DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING**

OPERATING SYSTEM

NAME OF THE FACULTY – GIRIJA PRASAD SWAIN
SEMESTER - 4TH

INTRODUCTION

UNIT-1

1. INTRODUCTION

   ObjectivesandExplainfunctionsofoperatingsystem.

   EvolutionofOperatingsystem

   Structureofoperatingsystem.

DEFINITIONOFOS:-

IntheComputerSystem(comprisesofHardwareandsoftware),Hardwarecanonlyunderstandmachine code (in the form of 0 and 1) which doesn't make any sense to a naive user.

Weneedasystemwhichcanactasan intermediaryandmanagealltheprocessesandresources present in the system.

An **Operating System** can be defined as an **interface between user and hardware**. It is responsible for the execution of all the processes, Resource Allocation, CPU management, File Management and many other tasks.

The purpose of an operating system is to provide an environment in which a user can execute programs in convenient and efficient manner.

STRUCTUREOFACOMPUTER SYSTEM

AComputerSystemconsistsof:
o   Users(peoplewhoareusingthecomputer)
o   ApplicationPrograms(Compilers,Databases,Games,Videoplayer,Browsers,etc.)
o   SystemPrograms(Shells,Editors,Compilers,etc.)
o   OperatingSystem(Aspecialprogramwhichactsasaninterfacebetweenuserandhardware)
o   Hardware (CPU,Disks,Memory,etc)
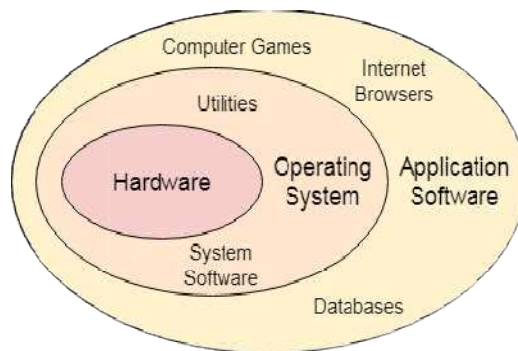
OBJECTIVESANDEXPLAINFUNCTIONSOFOPERATINGSYSTEM.

OBJECTIVES

ThemainobjectivesofanOperatingSystem are,

- **Convenience**–Anoperatingsystemmakesa computermoreconvenienttouse.
- **Efficiency**–Anoperatingsystempermitstheresourcestobeusedefficiently.
- **Ability to evolve** – An operating system should be constructed in such a way that, it should allow theeffectivedevelopment,testingandintroductionofnewfeatureswithoutanyinterferencewith service.

Th hardware and software in a computer system can be viewed in a layered fashion as shown in the

figure.
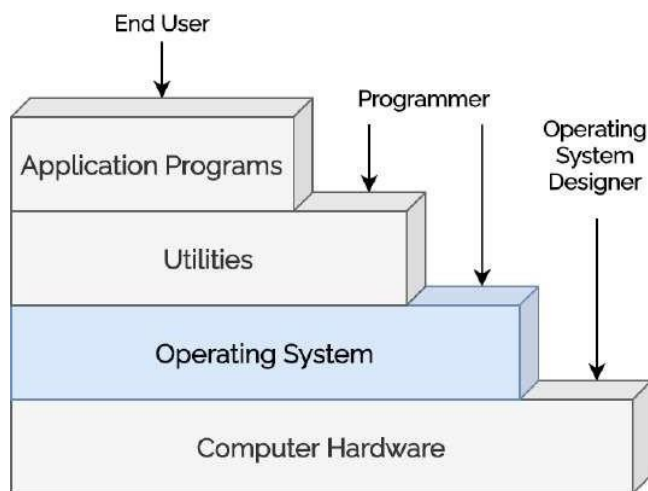
The**end-user**viewsthecomputersystemasasetofapplications.Applicationusersarenot

concerned with the details of computer hardware. The operating system masks the details of the

hardware from the **programmer** and acts as a mediator, making it easier to access the services.

FUNCTIONSOFOPERATINGSYSTEM

1. MemoryManagement

Itisthemanagementofthemainorprimarymemory.Whateverprogramisexecuted,ithastobepresent   in themain memory.Main memoryisa quickstorageareathatmay beaccesseddirectlyby theCPU. Whentheprogramiscompleted, thememoryregion isreleasedand can be usedbyotherprograms. Therefore,therecanbemorethanoneprogrampresentatatime.Hence,itisrequiredtomanagethe memory.

Theoperating system:

- Allocatesanddeallocatesthememory.

- Keepsarecordofwhichpartofprimarymemoryisusedbywhomandhowmuch.

- Distributesthememorywhilemultiprocessing.

- Inmultiprogramming,theoperatingsystemselectswhichprocessesacquirememorywhenand how much memory they get.

2. ProcessorManagement/Scheduling

Every software that runs on a computer, whether in the background or in the frontend, is a process. Processor management is an execution unit in which a program operates. The operating system determines the status of the processor and processes, selects a job and its processor, allocates the processor to the process, and de-allocates the processor after the process is completed.

WhenmorethanoneprocessrunsonthesystemtheOSdecideshowandwhenaprocesswillusethe CPU.Hence,thenameisalso**CPUScheduling**.TheOS:

- Allocatesanddeallocatesprocessortotheprocesses.

- KeepsrecordofCPUstatus.

CertainalgorithmsusedforCPUschedulingareasfollows:

- First ComeFirstServe(FCFS)

- ShortestJobFirst(SJF)

- Round-RobinScheduling

- Priority-basedschedulingetc.

Purposeof CPU scheduling

ThepurposeofCPUschedulingisasfollows:

- ProperutilizationofCPU.SincetheproperutilizationoftheCPUisnecessary.Therefore,theOS makes sure that the CPU should be as busy aspossible.

- Sinceeverydeviceshouldgetachancetousetheprocessor.Hence,theOSmakessurethatthe devices get fair processor time.

- Increasingtheefficiencyofthesystem.

3. DeviceManagement

Anoperatingsystemregulatesdeviceconnectionusingdrivers.Theprocessesmayrequiredevicesfor their use.ThismanagementisdonebytheOS.TheOS:

- Allocatesanddeallocatesdevicestodifferentprocesses.

- Keepsrecordsofthedevices.

- Decideswhichprocesscanusewhichdeviceforhowmuchtime.

4. FileManagement

Theoperatingsystemmanagesresourceallocationandde-allocation.Itspecifieswhichprocessreceives the file and for how long. It also keeps track of information, location, uses, status, and so on. These groupings of resources are referred to as file systems. The files on a system are storedin different directories. The OS:

- Keepsrecordsofthestatusandlocationsoffiles.

- Allocatesanddeallocatesresources.

- Decideswhogetstheresources.

5. StorageManagement

Storagemanagementisaprocedurethatallowsuserstomaximizetheutilizationofstoragedeviceswhile alsoprotectingdataintegrityonwhatevermediaonwhichitlives.Networkvirtualization,replication, mirroring, security, compression, deduplication, traffic analysis, processautomation, storage provisioning,andmemorymanagementaresomeofthefeaturesthatmaybeincluded.Theoperating system isincharge ofstoringand accessingfiles.The creationof files, thecreation of directories,the readingandwritingofdatafromfilesanddirectories,aswellasthecopyingofthecontentsoffilesand directoriesfromonelocationtoanotherareallincludedinstoragemanagement.

TheOSusesstoragemanagementfor:

- Improvingtheperformanceofthedatastorageresources.

- Itoptimizestheuseofvariousstoragedevices.

- Assistsbusinessesinstoringmoredataonexistinghardware,speedingupthedataretrieval process, preventing data loss, meeting data retention regulations, and lowering IT costs

The Operating System provides certain services to the users which can be listed **in the following manner:**

- **ProgramExecution**:

The Operating System is responsible for execution of all types of programs whether it be user programsorsystemprograms.TheOperatingSystemutilisesvariousresourcesavailableforthe efficient running of all types of functionalities.

- **HandlingInput/OutputOperations**:

The Operating System is responsible for handling all sort of inputs, i.e, from keyboard, mouse, desktop, etc. The Operating System does all interfacing in the most appropriate manner regrading all kind of Inputs and Outputs. For example, there is difference in nature of all types of peripheral devices such as mouse or keyboard, then Operating System is responsible for handling data betweenthem.

- **ManipulationofFileSystem**:

The Operating System is responsible for making of decisions regarding the storage of all types of data or files, i.e, floppy disk/hard disk/pen drive, etc. The Operating System decides as how should the data should be manipulated and stored.

- **ErrorDetectionandHandling**:

The Operating System is responsible for detection of any types of error or bugs that can occur while anytask. Thewellsecured OSsometimesalso actsascountermeasureforpreventingany sort of breach to the Computer System from any external source and probably handling them.

- **ResourceAllocation:**

The Operating System ensures the proper use of all the resources available by deciding which resource to be used by whom for how much time. All the decisions are taken by the Operating System.

- **Accounting:**TheOperatingSystemtracksanaccountofallthefunctionalitiestakingplacein thecomputersystematatime.Allthedetailssuchasthetypesoferrorsoccurredarerecorded by the Operating System.

- **Information and Resource Protection:** The Operating System is responsible for using allthe information and resources available on the machine in the most protected way. The Operating System must foil an attempt from any external resource to hamper any sort of data or information.
- All these services are ensured by the Operating System for the convenience of the users to maketheprogrammingtaskeasier.AlldifferentkindsofOperatingSystemmoreorlessprovide the same services

## EVOLUTIONOFOPERATINGSYSTEM

BATCHOPERATINGSYSTEM

Inthe1970s,Batchprocessingwasverypopular.Inthistechnique,similartypesofjobswerebatched together and executed in time. People were used to having a single computer which was called a mainframe.

InBatchoperatingsystem,accessisgiventomorethanoneperson;theysubmittheirrespectivejobs to the system for the execution.

The system put all of the jobs in a queue on the basis of first come first serve and then executes the jobs one by one. The users collect their respective output when all the jobs get executed.
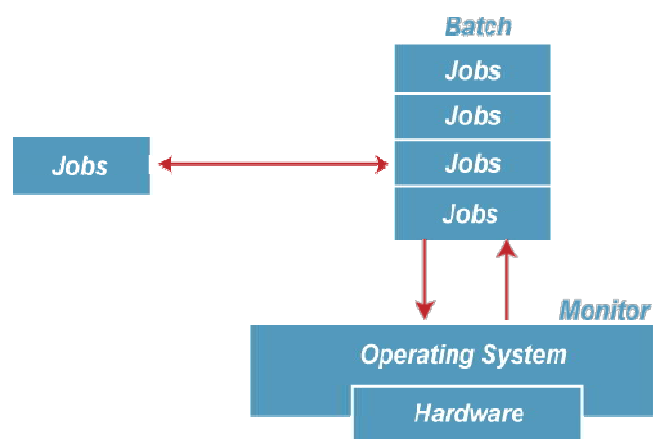
Thepurposeofthisoperatingsystemwasmainlytotransfercontrolfromonejobtoanotherassoon asthejobwascompleted.Itcontainedasmallsetofprogramscalledtheresidentmonitorthatalways resided in one part of the main memory. The remaining part is used for servicing jobs.

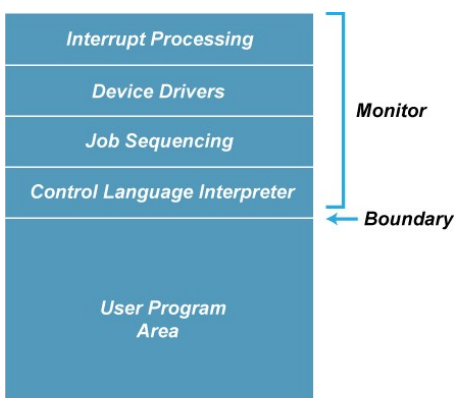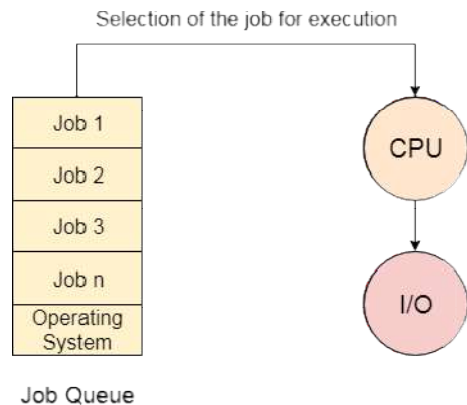Figure: Memory Layout of the resident monitor

AdvantagesofBatchOS

- o TheuseofaresidentmonitorimprovescomputerefficiencyasiteliminatesCPUtime between two jobs.

1. Starvation

Batchprocessingsuffersfromstarvation.

ForExample:

Job Queue

There are five jobs J1, J2, J3, J4, and J5, present in the batch. If the execution time of J1 is very high, then the other four jobswill never be executed, or they will have to wait for a very longtime. Hence the other processes get starved.

2. NotInteractive

Batch Processing is not suitable for jobs that are dependent on the user's input. If a job requires the inputoftwonumbersfromtheconsole,thenitwillnevergetitinthebatchprocessingscenariosince the user is not present at the time of execution.

MULTIPROGRAMMINGOPERATINGSYSTEM

MultiprogrammingisanextensiontobatchprocessingwheretheCPUisalwayskeptbusy.Each process needs two types of system time: CPU time and IO time.

Inamultiprogrammingenvironment,whenaprocessdoesitsI/O,TheCPUcanstarttheexecution of other processes. Therefore, multiprogramming improves the efficiency of the system.
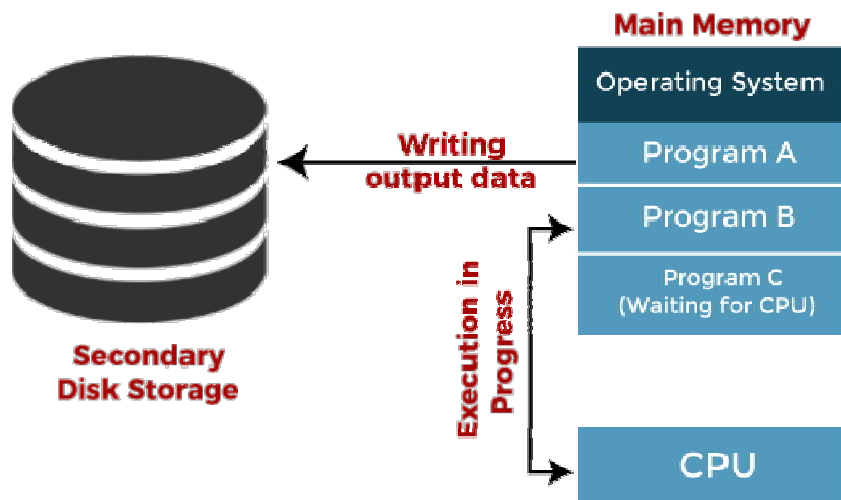
Jobs in multiprogramming system

AdvantagesofMultiprogrammingOS

- o ThroughoutthesystemitincreasedastheCPUalwayshadoneprogramtoexecute.
- o Responsetimecanalsobereduced.

DisadvantagesofMultiprogrammingOS

- o Multiprogrammingsystemsprovideanenvironmentinwhichvarioussystemsresourcesare used efficiently, but they do not provide any user interaction with the computersystem.

MULTIPROCESSINGOPERATINGSYSTEM

InMultiprocessing,Parallelcomputingisachieved.Therearemorethanoneprocessorspresentinthe systemwhichcanexecutemorethanoneprocessatthesametime.Thiswillincreasethethroughput of the system.
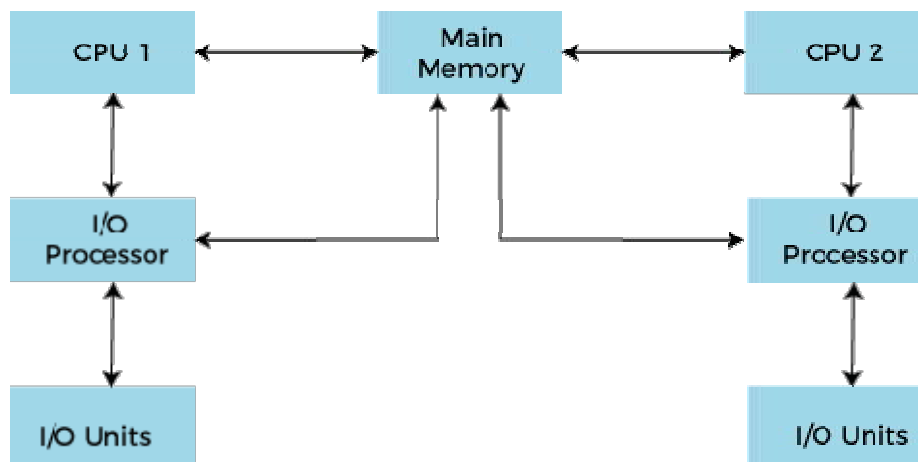
Working of Multiprocessor System

In Multiprocessing, Parallel computing is achieved. More than one processor present in the system canexecutemorethanoneprocesssimultaneously,whichwillincreasethethroughputofthesystem.
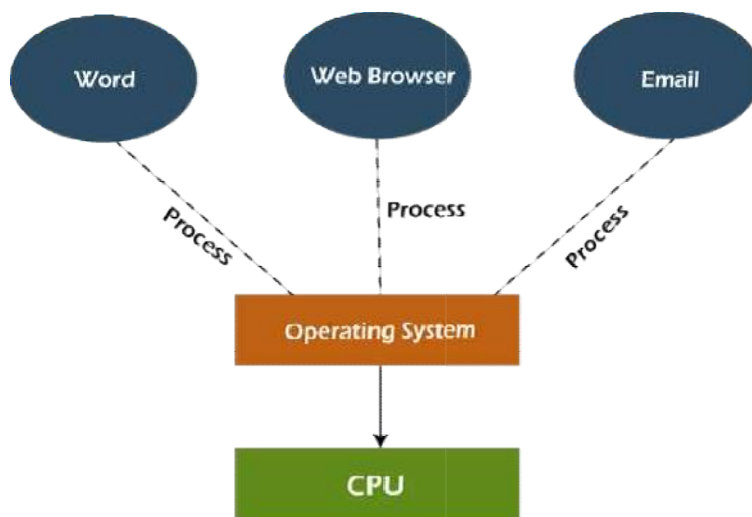
AdvantagesofMultiprocessingoperatingsystem:

- o **Increased reliability:** Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails, the task can be given to another processor for completion.
- o **Increasedthroughout:**Asseveralprocessorsincrease,moreworkcanbedoneinless.

DisadvantagesofMultiprocessingoperatingSystem

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

MultitaskingOperatingSystem

Themultitaskingoperatingsystemisalogicalextensionofamultiprogrammingsystemthatenables **multiple** programs simultaneously. It allows a user to perform more than one computer task at the same time.

AdvantagesofMultitaskingoperatingsystem

- o Thisoperatingsystemismoresuitedtosupportingmultipleuserssimultaneously.
- o Themultitaskingoperatingsystemshavewell-definedmemorymanagement.

- o Themultipleprocessorsarebusieratthesametimetocompleteanytaskinamultitasking environment, so the CPU generates more heat.

NETWORKOPERATINGSYSTEM

Network Operating Systems
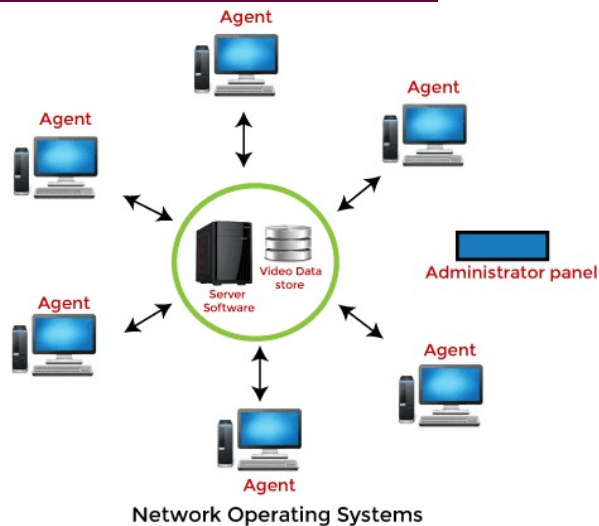
AnOperatingsystem,whichincludessoftwareandassociatedprotocolstocommunicatewithother computers via a network conveniently and cost-effectively, is called Network Operating System.

AdvantagesofNetworkOperatingSystem

- o Inthistypeofoperatingsystem,networktrafficreducesduetothedivisionbetweenclients and the server.
- o Thistypeofsystemislessexpensivetosetupandmaintain.

DisadvantagesofNetworkOperatingSystem

- o Inthistypeofoperatingsystem,thefailureofanynodeinasystemaffectsthewhole system.
- o Securityandperformanceareimportantissues.Sotrained networkadministratorsare required for network administration.

RealTimeOperatingSystem

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

Real - Time Operating System (RTOS)
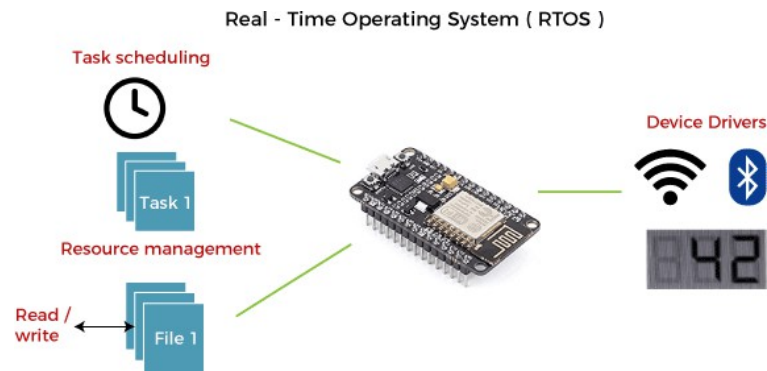
TheApplicationofaReal-Timesystemexistsinthecaseofmilitaryapplications,ifyouwanttodrop a missile, then the missile is supposed to be dropped with a certainprecision.

AdvantagesofReal-timeoperatingsystem:

o Easytolayout,developandexecutereal-timeapplicationsunderthereal-timeoperating system.

o InaReal-timeoperatingsystem,themaximumutilizationofdevicesandsystems.

DisadvantagesofReal-timeoperatingsystem:

o Real-timeoperatingsystems areverycostlytodevelop.

o Real-timeoperatingsystemsareverycomplexandcanconsumecriticalCPUcycles.

TIME-SHARINGOPERATINGSYSTEM

IntheTimeSharingoperatingsystem,computerresourcesareallocatedinatime-dependentfashion toseveralprogramssimultaneously.Thusithelpstoprovidealargenumberofuser'sdirectaccessto themaincomputer.Itisalogicalextensionofmultiprogramming.Intime-sharing,theCPUisswitched among multiple programs given by different users on a scheduledbasis.
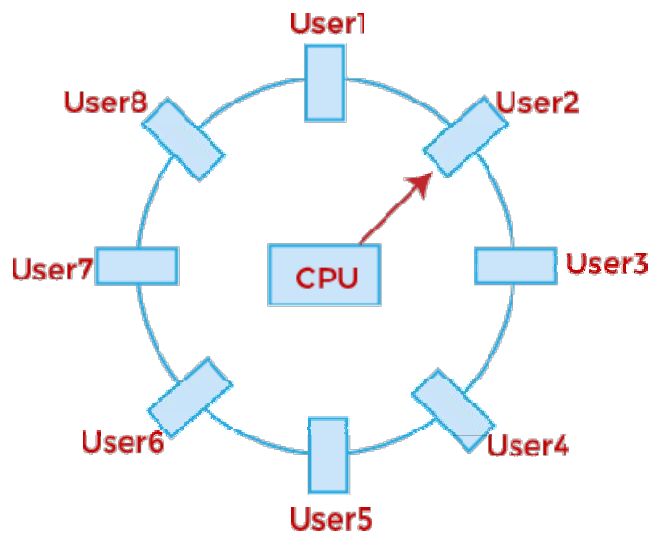
Timesharing in case of 8 users

Atime-sharingoperatingsystemallowsmanyuserstobeservedsimultaneously,sosophisticated CPU scheduling schemes and Input/output management are required.

Time-sharingoperatingsystemsareverydifficultandexpensivetobuild.

AdvantagesofTimeSharingOperatingSystem

- o Thetime-sharingoperatingsystemprovideseffectiveutilizationandsharingofresources.
- o ThissystemreducesCPUidleandresponse time.

DisadvantagesofTimeSharingOperatingSystem

- o Datatransmissionratesareveryhighincomparisontoothermethods.
- o Securityandintegrityofuserprogramsloadedinmemoryanddataneedtobemaintainedas many users access the system at the same time.

DISTRIBUTEDOPERATINGSYSTEM

The Distributed Operating system is not installed on a single machine, it is divided into parts, and these parts are loaded on different machines. A part of the distributed Operating system is installed on each machine to make their communication possible. Distributed Operating systems are much more complex, large, and sophisticated than Network operating systems because they also have to take care of varying networking protocols.
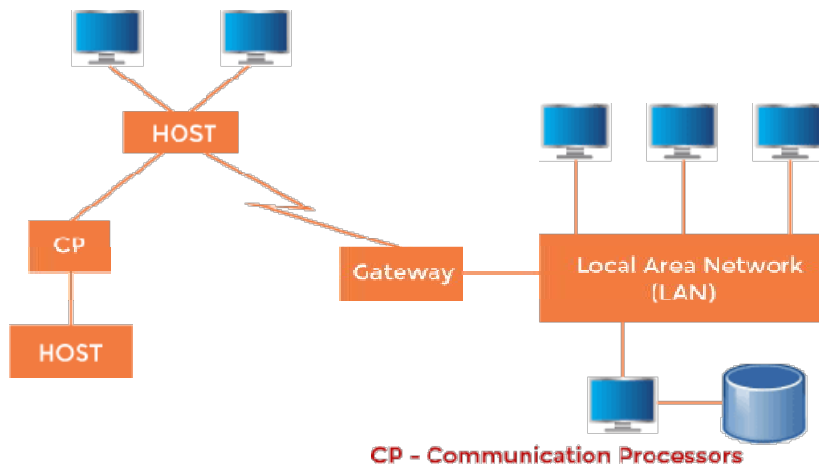
A Typical View of a Distributed System

AdvantagesofDistributedOperatingSystem

- o Thedistributedoperatingsystemprovidessharingofresources.
- o Thistypeofsystemisfault-tolerant.

DisadvantagesofDistributedOperatingSystem

- o Protocoloverheadcandominatecomputationcost.

STRUCTUREOFOPERATINGSYSTEM.

- o **WhatistheOperatingSystemStructure?**
- o AnOperatingSystemhasacomplexstructure,Sotoimplementsuchacomplexthing,weneed agoodpredefinedstructurethatwillhelpustoimplementOperatingsystemsforourspecific tasks.
- o Just like we break down the complex problem into subproblems that are easier to solve, an easier way to design an OS is by making it in parts.
- o AndeachpartisacomponentoftheOperating System.
- o Wecandescribeoperatingsystemstructureasthemethodofinterconnectingandintegrating various components of the operating system into the kernel. There are different types of structures on which operating system is implemented as discussedbelow:

STRUCTUREOFOPERATINGSYSTEM

ItisthesimplestOperatingSystemStructureandisnotwelldefined,Itcanonlybeusedforsmalland   limited systems. In this structure,theinterfaces and levelsof functionalityare well separated, hence programs can access I/O routines which can cause unauthorized access to I/O routines.

ThisstructureisimplementedinMS-DOSoperatingsystem:

- The MS-DOS operating system is made up of various layers, each with its own set of functions. These layers are:
  - o Application Program
  - o System Program
  - o MS-DOS device drivers
  - o ROM BIOS device drivers
- Layering has an advantage in the MS-DOS operating system since all the levels can be defined separately and can interact with each other when needed.
- It is easier to design, maintain, and update the system if it is made in layers. So that's why limited systems with less complexity can be constructed easily using Simple Structure.
- If one user program fails, the entire operating system gets crashed.
- The abstraction level in MS-DOS system is slow, so programs and I/O routines are visible to the end-user, so the user can have unauthorized access.

Advantages of Simple Structure

- It is easy to develop because of the limited number of interfaces and layers.
- Offers good performance due to lesser layers between hardware and applications.

Disadvantages of Simple Structure

- If one user program fails, the entire operating system crashes.
- Abstraction or data hiding is not present as layers are connected and communicate with each other.
- Layers can access the processes going in the Operating System which can lead to data modification and can cause Operating System to crash.
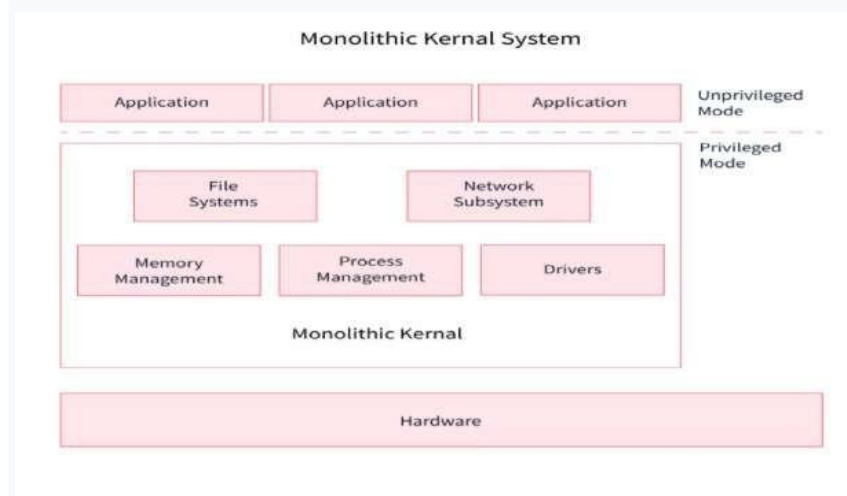
MONOLITHIC STRUCTURE

The monolithic operating system in which kernel acts as a manager by managing all things like file management, memory management, device management, and operational processes of the Operating system.

The kernel is the heart of a computer operating system (OS). Kernel delivers basic services to all other elements of the system. It serves as the primary interface between the operating system and the hardware.

In monolithic systems, kernels can directly access all the resources of the operating system like physical hardware exp Keyboard, Mouse, etc.

The monolithic kernel is another name for the monolithic operating system. Batch processing and time-sharing maximize the usability of a processor by multiprogramming. The monolithic kernel functions as a virtual machine by working on the top of the Operating system and controlling all hardware components. This is an outdated operating system that was used in banks to accomplish minor activities such as batch processing and time-sharing which enables many people, at various terminals to access the Operating system.

A Diagram of the Monolithic structure is shown below:

Advantages of Monolithic structure:

- It is simple to design and implement because all operations are managed by kernel only and layering is not needed.
- As services such as memory management, file management, process scheduling, etc are implemented in the same address space, the execution of the monolithic kernel is relatively fast as compared to normal systems. Using the same address saves time for address allocation for new processes and makes it faster.

Disadvantages of Monolithic structure:

- If any service in the monolithic kernel fails, the entire system fails because, in address space, the services are connected to each other and affect each other.
- It is not flexible and to introduce a new service

LAYEREDAPPROACH

In this type of structure, OS is dividedintolayersorlevels.Thehardwareisonthebottomlayer (layer 0), while the user interface is on the top layer (layer N). These layers are arranged in a hierarchical way in which the top-level layers use the functionalities of their lower-levellevels.

In this approach functionalities of each layer are isolated and abstraction is also available. In layered structure,debuggingiseasierasitisahierarchicalmodel,soalllower-levellayeredaredebuggedand thentheupperlayerischecked.Soallthelowerlayersarealreadycheckedandthecurrentlayeristo   be checked only.

BelowistheImageillustratingtheLayeredstructurein OS:
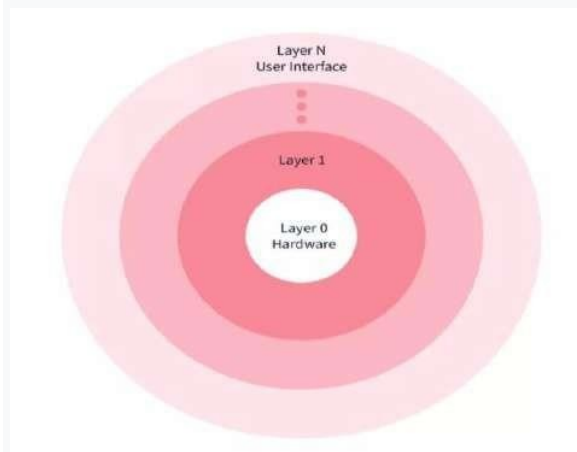
Advantages ofLayeredStructure

- Eachlayerhasitsfunctionalitiessoworktasksareisolatedandabstractionispresentupto some level.
- Debuggingiseasieraslowerlayersaredebuggedandthenupperlayersarechecked.

DisadvantagesofLayeredStructure

- InLayeredStructure,layeringcausesdegradationinperformance.
- Ittakescarefulplanningtoconstructthelayerssincehigherlayersonlyutilizethefunctions of lower layers.

MICRO-KERNEL

Micro-Kernel structure designs the Operating System by removing all non-essential components of the kernel. These non-essential components of kernels are implemented as systems and user programs. Hence these implemented systems are called as Micro-Kernels.

Each Micro-Kernel is made independently and is isolated from other Micro-Kernels. So this makes the system more secure and reliable. If any Micro-Kernel fails then the remaining operating System remains untouched and works fine.

Advantages of Micro-kernel structure:

- It allows the operating system to be portable between platforms.
- As each Micro-Kernel is isolated, it is safe and trustworthy.
- Because Micro-Kernels are smaller, they can be successfully tested.
- If any component or Micro-Kernel fails, the remaining operating system is unaffected and continues to function normally.

Disadvantages of Micro-kernel structure:

- Increased inter-module communication reduces system performance.
- System is complex to be constructed.

Conclusion

- The operating system enables the end-user to interact with the computer hardware. System software is installed and utilized on top of the operating system.
- We can define operating system structure as to how different components of the operating system are interconnected.
- There are many structures of the operating system:
    - Simple Structure
    - Monolithic Approach
    - Layered Approach
    - Microkernels
- All these approaches or structures evolved from time to time making the OS more and more improved than before.
-
-

Microkerna1OperotingSystem

Appfication

Application IPC

Device Driver

Unix Server

File Server

BasicIPC, Virtual*Memory*Scheduling

Hardware
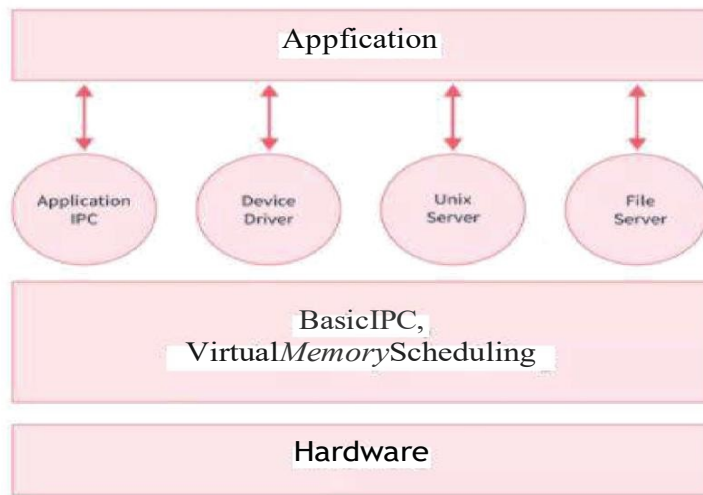
PROCESSMANAGEMENT

UNITII

2. PROCESSMANAGEMENT

Processconcept,processcontrol,interactingprocesse s, inter process messages.

ImplementationissuesofProcesses.

Processscheduling,jobscheduling.

Processsynchronization,semaphore.

Principleofconcurrency,typesofscheduling

2.PROCESSMANAGEMENT

Definitionofprocess:

Aprocessisbasicallyaprograminexecutionorinstanceoftheprogramexecution.The execution of a process must progress in a sequential fashion.

- Processisnotassameasprogramcodebutalotmorethanit.
- Aprocessisan'active'entityasopposedtoprogramwhichisconsideredtobea 'passive' entity.
- Attributesheldbyprocessincludehardwarestate,memory,CPUetc.

Processmemoryisdividedintofoursectionsforefficientworking:
- The***Textsection***is madeupofthecompiledprogramcode,readinfromnon-volatile storage when the program is launched.
- The***Datasection***ismadeuptheglobalandstaticvariables,allocatedand initialized prior to executing the main.
- The***Heap***isusedforthedynamicmemoryallocation,andismanagedviacallsto new, delete, malloc, free, etc.
- The***Stack***isused forlocalvariables.Spaceonthestackisreservedforlocal variables when they are declared.
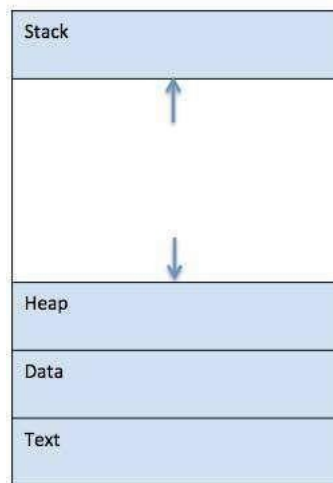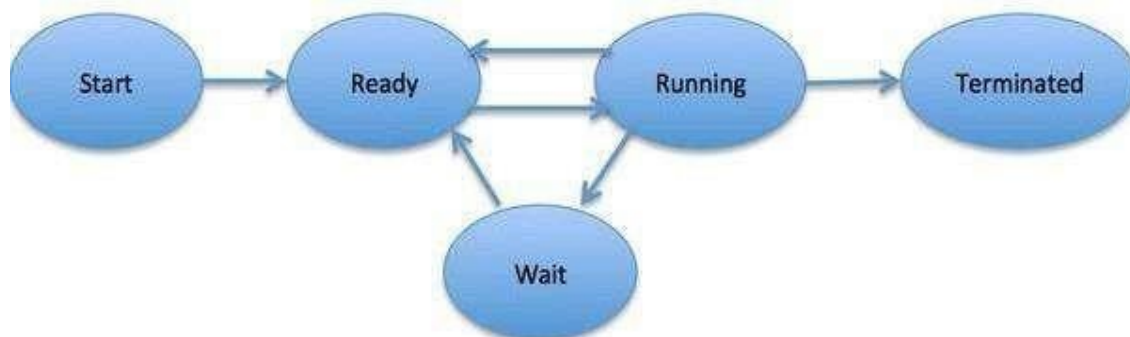
ProcessStates:

Whenaprocessexecutes,itpassesthroughdifferentstates.Thesestages maydifferin different operating systems.

Ingeneral,aprocesscanhaveoneofthefollowingfivestatesata time.

| S.N.	State& Description
1	**Start:**Thisis theinitial state when aprocess isfirst started/created.
2	**Ready:** The process is waiting to be assigned to a processor. Ready processes are waitingtohavetheprocessorallocatedtothembytheoperatingsystemsothatthey canrun.Processmaycomeintothisstateafter**Start**stateorwhilerunningitbybut interrupted by the scheduler to assign CPU to some other process.
3	**Running:**OncetheprocesshasbeenassignedtoaprocessorbytheOSscheduler,the process state is set to running and the processor executes its instructions.
4	**Waiting:**Processmovesintothewaitingstateifitneedstowaitforaresource,suchas waiting for user input, or waiting for a file to become available.
5	**Terminated or Exit:** Once the process finishes its execution, or it is terminated bythe operatingsystem, it is moved to theterminatedstatewhereit waits to beremovedfrom main memory.

ProcessStateDiagram

Process Control Block (PCB):

- AProcessControlBlockisadatastructuremaintainedbytheOperatingSystemfor everyprocess.

- ThePCBisidentifiedbyanintegerprocessID(PID).
- APCBkeepsalltheinformationneededtokeeptrackofaprocessaslistedbelowin the table –

S.N.	Information& Description
1	**ProcessState:**Thecurrentstateoftheprocessi.e.,whetheritisready,running, waiting, or whatever.
2	**Processprivileges:**Thisisrequiredtoallow/disallowaccesstosystemresources.
3	**ProcessID:**Uniqueidentificationforeach oftheprocessintheoperatingsystem.
4	**Pointer:**A pointerto parent process.
5	**ProgramCounter:P**rogramCounterisapointertotheaddressofthenextinstruction to be executed for this process.
6	**CPUregisters:**VariousCPUregisterswhereprocessneedtobestoredforexecution for running state.
7	**CPUSchedulingInformation:**Processpriorityandotherschedulinginformation which is required to schedule the process.
8	**Memory management information:** This includes the information of page table, memorylimits,Segmenttabledependingonmemoryusedbytheoperatingsystem.
9	**Accountinginformation:**ThisincludestheamountofCPUusedforprocess execution, time limits, execution ID etc.
10	**IOstatusinformation:**Thisincludesalistof I/Odevicesallocatedtothe process.

ThearchitectureofaPCBiscompletelydependentonOperatingSystemandmaycontaindifferent information in different operating systems. Here is a simplified diagram of a PCB –
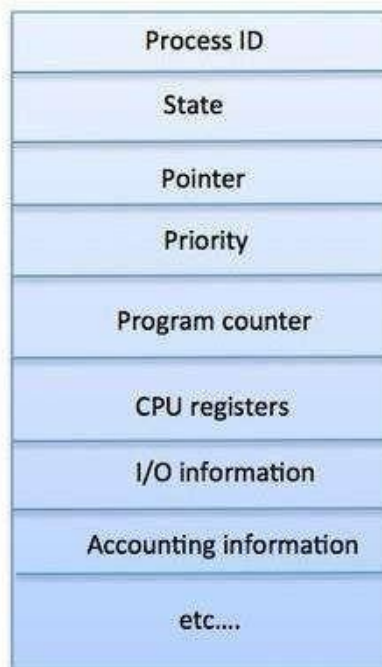
ProcessControlBlock(PCB)Diagram

The PCB is maintained for a process throughout its lifetime, and is deleted once the process terminates.

ProcessSchedulingDefinition

- The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.
- Process scheduling is an essential part of a Multiprogramming operating systems.
- Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

WhatareSchedulingQueues?

- All processes, upon entering into the system, are stored in the **JobQueue**.
- Processes in the Ready state are placed in the **ReadyQueue**.
- Processes waiting for a device to become available are placed in **DeviceQueues**.
  There are unique device queues available for each I/O device.

A new process is initially put in the **Ready queue**. It waits in the ready queue until it is selected for execution (or dispatched). Once the process is assigned to the CPU and is executing, one of the following several events can occur:

- The process could issue an I/O request, and then be placed in the **I/Oqueue**.
- The process could create a new sub-process and wait for its termination.
- The process could be removed forcibly from the CPU, as a result of an interrupt, and be put back in the ready queue.
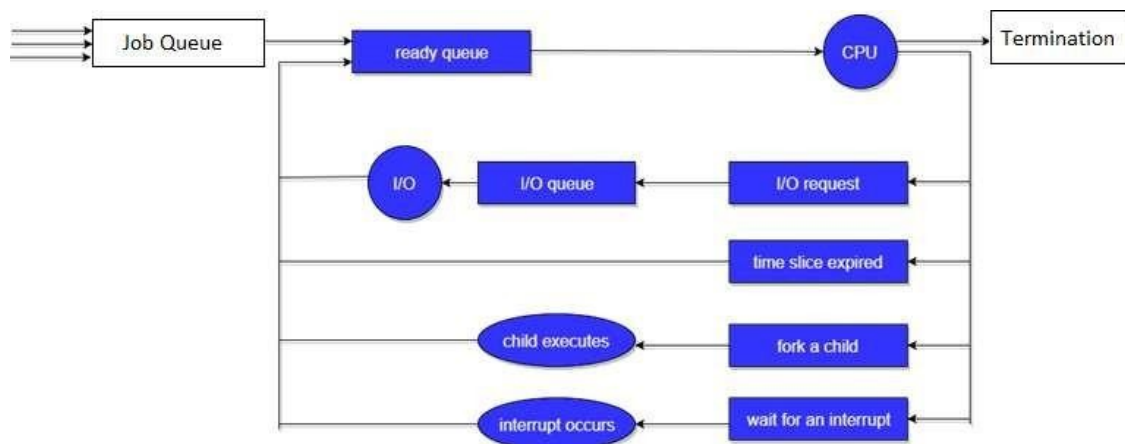
In the first two cases, the process eventually switches from the waiting state to the ready state, and is then put back in the ready queue. A process continues this cycle until it terminates, at which time it is removed from all queues and has its PCB and resources deallocated.

Schedulers:

- Schedulers are special system software which handle process scheduling in various ways.
- Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types −

- Long-TermScheduler
- Short-TermScheduler
- Medium-TermScheduler

LongTermScheduler
- Itisalsocalledajobscheduler.
- Along-termschedulerdetermineswhichprogramsareadmittedtothesystemfor processing.
- Itselectsprocessesfromthequeueandloadsthemintomemoryforexecution.
- ProcessloadsintothememoryforCPUscheduling.
- Theprimaryobjectiveofthejobscheduleristoprovideabalancedmixofjobs,suchas I/O bound and processor bound.

- Italsocontrolsthedegreeofmultiprogramming.
- Ifthedegreeofmultiprogrammingisstable,thentheaveragerateofprocesscreation must be equal to the average departure rate of processes leaving thesystem.
- Onsomesystems,thelong-termschedulermaynotbeavailableorminimal.
- Time-sharingoperatingsystemshavenolongtermscheduler.
- Whenaprocesschangesthestatefromnewtoready,thenthereis useoflong-term scheduler.

ShortTermScheduler:
- ItisalsocalledasCPUscheduler.
- Itsmainobjectiveistoincreasesystemperformanceinaccordancewiththechosenset of criteria.
- Itisthechangeofreadystatetorunningstateoftheprocess.
- CPUschedulerselects aprocessamongthe processesthatarereadytoexecuteand allocates CPU to one of them.
- Short-term schedulers, also known as dispatchers, make the decision of which processtoexecutenext.Short-termschedulersarefaster thanlong-termschedulers.

MediumTermScheduler
- Medium-termschedulingisapartofswapping.
- Itremovestheprocessesfromthememory.
- Itreducesthedegreeofmultiprogramming.
- Themedium-termschedulerisin-chargeofhandlingtheswappedout-processes.
- ArunningprocessmaybecomesuspendedifitmakesanI/Orequest.
- Asuspendedprocesscannotmakeanyprogresstowardscompletion.
- Inthiscondition,toremovetheprocessfrommemoryandmakespaceforother processes, the suspended process is moved to the secondarystorage.
- Thisprocessiscalledswapping,andtheprocessissaidtobe swappedoutor rolled out.
- Swappingmaybenecessarytoimprovetheprocessmix.
-

ComparisonamongScheduler

S.N.	Long-TermScheduler	Short-TermScheduler	Medium-TermScheduler
1	Itisajobscheduler	ItisaCPU scheduler	Itisaprocessswapping scheduler.
2	Speedislesserthanshort term scheduler	Speedisfastestamong other two	Speed is in between both short and long term scheduler.
3	Itcontrolsthedegreeof multiprogramming	Itprovideslessercontrol over degree of multiprogramming	Itreducesthedegreeof multiprogramming.
4	It is almost absent or minimal in time sharing system	Itisalsominimalintime sharing system	ItisapartofTimesharing systems.
5	It selects processes from pool and loads them into memory for execution	Itselectsthoseprocesses which are ready to execute	It can re-introduce the process into memory and execution can becontinued.

CONTEXTSWITCH:

- Acontext switch is the mechanismto store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time.
- Using this technique, a context switcher enables multiple processes to share a single CPU.
- Contextswitchingisanessentialpartofamultitaskingoperatingsystemfeatures.
- WhentheschedulerswitchestheCPUfromexecutingoneprocesstoexecuteanother, the state from the current running process is stored into the process control block.
- After this, the state for the process to run next is loaded from its own PCB and used to set the PC, registers, etc.
- Atthatpoint,thesecondprocesscanstartexecuting.

- Contextswitchesarecomputationallyintensivesinceregisterandmemorystatemustbe saved and restored.
- Toavoidtheamountofcontextswitchingtime,somehardwaresystemsemploytwoor more sets of processor registers.
- Whentheprocessisswitched,thefollowinginformationisstoredforlateruse.
  - ProgramCounter
  - Schedulinginformation
  - Baseandlimitregistervalue
  - Currentlyusedregister
  - ChangedState
  - I/OStateinformation
  - Accounting information

Inter-processCommunication

Inter process 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.

Aprocesscanbeoftwo types:

1. **Independentprocess.**

2. **Co-operatingprocess.**

An independent process is not affected by the execution of other processes

cooperating process can be affected by other executing processes. Though one can think that those processes, which are running independently, will execute very efficiently, in reality, there are many situations when co-operative nature can be utilised for increasing computational speed, convenience and modularity.

ADVANTAGES OF CO-OPERATING PROCESS:-

Following are the advantages.
⇒ Information sharing
⇒ Computation speedup
⇒ Modularity
⇒ Convenience

INFORMATION SHARING:- ⇒ In multiprogramming systems several users may need the same piece of information.

⇒ Co-operating environment allow concurrent access to these type of resources.

COMPUTATION SPEEDUP:- ⇒ To execute a task faster, it must break into subtasks. Each subtasks executes parallel with each other, such a speed can be achieved only if the computer has multiprocessing elements.

MODULARITY:- ⇒ To construct a system in modular approach divide the system functions into no. of separate processes.

CONVENIENCE:- ⇒ A user can perform many task at one time. Ex.   A

user may be editing, printing and compiling in parallel **INTER**

PROCESS COMMUNICATION (IPC)

Interprocess communication (IPC) is a mechanism which allows processes to communicate with each other and synchronize their actions. The communication between these processes can be seen as a method of cooperation between them.

Processes can communicate with each other through both:

1. **Shared Memory**
2. **Message passing**

1. Shared Memory

In shared memory system a region of memory is shared by the co-operating processes.

⇒ Then the processes can share the information by reading and writing data to the shared region.
⇒ Shared m.s. allows maximum speed and convenience of communication.
⇒ In shared memory system the shared region has been created through system calls.
⇒ A shared memory region resides in the address space of the process creating the share region.
⇒ In shared memory regional all access are treated as routine memory access.

⇒Otherprocessesmustattachedthereaddressspacewiththesharedmemorysegmentfor communication
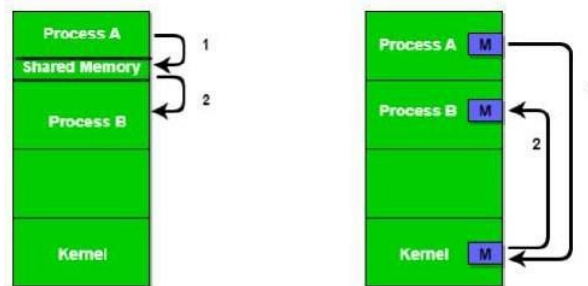

Figure 1 - Shared Memory and Message Passing

THISPROBLEMAREOFTWOCATEGORIES.

     i)   Boundedbufferproblem

     ii)   Unboundedbufferproblem.

i)   Boundedbufferproblem:-

⇒Inthisproblemthebuffersizeisfixed.

⇒Iftheproducerisfaster thantheconsumer, thenaftersome timethe bufferisfullandproducerhastowaittillthebuffer becomesempty.

⇒Whentheconsumingrateishighthanproducing,thenaftersometime buffer becomes emptyand the consumer has to wait.

ii)   Unboundedbufferproblem:-

⇒Herethebuffersizecanbe vary.

⇒Inthis case if the producer is faster than consumer, then the buffer is overloaded withdataandif theconsumerisfasterthanthe producerthen the consumer has to wait till the buffer becomes empty.

2.Messagepassing

⇒Message passing system is a mechanism to allow processes to communicateandsynchroniestheiractivitieswithoutsharingtheaddress space.

⇒Messagepassingmechanism/conceptisgenerallyusedindistributedenvironment.

⇒2primitivesoroperationsarethereinmessagepassingsystem.

     i)   Send():-To send amessage.

     ii)   Receive():-To receivemessage.

⇒Messagesusedforcommunicationareof2types i.e.

⇒Fixedsizedmsg

⇒Variablesizedmsg

⇒Inmessagepassingsystemcommunicationoccursin3 ways.

     i)   Direct/indirectcomm..

     ii)   Synchronous/asynchronouscomm..

     iii)   Explicit/implicit(automaticbuffering)

i) DirectComm:-

⇒Indirectcomm..theprocessesthatwanttocommunicate with each other mustknow each others name.

⇒Indirectcomm.Thereare2typesofaddressingornaming.

   a) Symmetricaddressing
   b) Asymmetric

a) Symmetric:-

⇒Insymmetricaddressingsend()operationwilloccurassend (receiving/ msg processname):

⇒Thereceiveoperationoccurasreceive(senderprocess,msgname);

send(P,message)-SendamessagetoprocessP.
receive(Q,message)-ReceiveamessagefromprocessQ.

Characteristicsofcommunicationlinkindirectcommunication:-

⇒Itisestablishedautomaticallybetweeneverypairofprocessthatwantsto communicate.
⇒Thelinkmustbeestablishedwithexactlytwoprocesses.
⇒Betweeneachpairofprocesses,thereshouldbeexactlyonelink.
⇒Thelinkmaybeunidirectionalorbidirectional.

b) Asymmetric:-

⇒Inthisaddressingonlysendernamesthereceiver.

⇒Thereceiverwillidentifythesenderprocessbyhisprocessid

send(P,message)-SendamessagetoprocessP.
receive(id,message) -Receivea messagefromanyprocess;thevariableidissettothe name of the process with which communication has taken place.

2)Indirectcommunication:-

⇒Inindirectcommunicationthemessagesaresendandreceivedbymailboxorport.
⇒Mailboxisanabstractobjectinwhichthemessagescanbeplacedandremoved.
⇒Twoprocessescancommunicateonlyiftheysharethesamemailbox.

⇒ 2 operation will occur as
Send (mailbox name,msg)
Receive(mailboxname,msg)

send(A,message)-SendamessagetomailboxA.
receive (A, message)-Receive a message from mailbox A.

⇒Indirectcommunicationhasthefollowingproperties

   i) Alinkisestablishedbetweeneverypairofprocessiftheyshareacommon mail box.

ii)   Alinkmaybeunidirectionalorbidirectional

iii)   Alinkmaybeassociatedbasicallywithtwoprocess(more
than two process also possible)

Synchronization:-

Communicationbetweenprocesses takesplacethroughcalls to send()andreceive ()
primitives. There are different design options for implementing each primitive.
Messagepassingmaybeeither**blockingornonblocking**alsoknownas**synchronous and
asynchronous.**

- **Blockingsend**.-Thesendingprocessisblockeduntilthemessageisreceivedby the
receiving process or by the mailbox.
- **Nonblockingsend**-Thesendingprocesssendsthemessageandresumes
operation.

- **Blockingreceive.**-Thereceiverblocksuntilamessageisavailable.
- **Nonblockingreceive**.-Thereceiverretrieveseitheravalidmessageoranull.

Buffering:-

⇒Alinkcapacitydeterminesano.ofmsgscanresideintemporaryqueue.

⇒Alinkcanbeassumeasaqueuelikestructure.

⇒Thislinkcanbeimplementedin3 ways
.i> Zero capacity (no buffing)
ii>   Bounded capacity
iii>   Unboundedcapacity.

⇒Inzerocapacitythequeuemaximumlengthiszero,thelinkcan'tholdnaymsgs waiting in it.
⇒Inboundedcapacity,thequeuehasfinitelength'n'.henceatmost'n'msgscanreside the link.
⇒Inunboundedcapacityanyno.ofmsgscanbereside.

IMPLEMENTATIONOFPROCESSES:

- ProcessModelisimplementedbyProcessTableandProcessControlBlockwhich keep
track all information of process.
- Atthetimeofcreationofanewprocess,operatingsystemallocatesamemoryforit loads a
process code in the allocated memory and setup data space for it .
- Thestateofprocessisstoredas'new'initsPCBandwhenthisprocessmovetoready state its
state is also changes in PCB.
- Whenarunningprocessneedstowaitforaninputoutputdevices,itsstateischanged

to'blocked'.Thevariousqueuesusedforthiswhichisimplementedaslinkedlist.

CPUSCHEDULINGINOPERATINGSYSTEM

CPU schedulingisa process that allowsoneprocess to usethe CPU while theexecution of another process is on hold(in waiting state) due to unavailability of any resource like I/O etc, thereby making full use of CPU. The aim of CPU scheduling is to make the systemefficient,fast,andfair.

Whenever the CPU becomes idle, the operating system must selectone of the processes in the ready queue to be executed. The selection process is carried out by the short-term scheduler (or CPU scheduler). The scheduler selects from among the processes in memorythatarereadytoexecuteandallocatestheCPUtoone ofthem.

TypesofCPUScheduling

HerearetwokindsofScheduling methods:
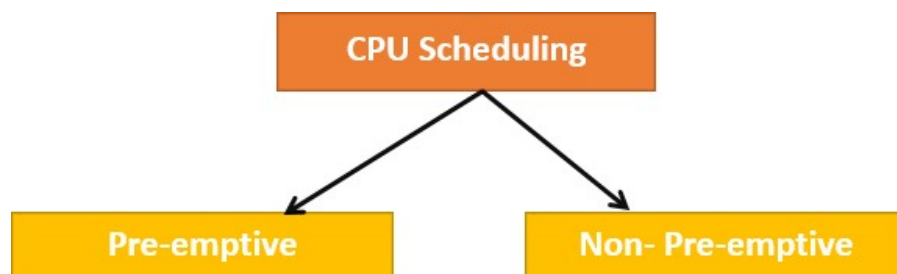
PreemptiveScheduling

In Preemptive Scheduling, the tasks are mostly assigned with their priorities. Sometimes it is important torunataskwithahigher prioritybeforeanother lower prioritytask, evenif thelower priority task is still running. The lower priority task holds for some time and resumes when the higher priority task finishes its execution.

Non-PreemptiveScheduling

In this type of scheduling method, the CPU has been allocated to a specific process. The process that keeps the CPU busy will release the CPU either by switching context or terminating. It is the only method that can be used for various hardware platforms. That's becauseitdoesn'tneedspecialhardware(forexample,atimer)likepreemptivescheduling.

WhenschedulingisPreemptiveorNon-Preemptive?

Todetermineifschedulingispreemptiveornon-preemptive,considerthesefourparameters:

1. A process switches from the running to the waiting state.
2. Specific process switches from the running state to the ready state.
3. Specific process switches from the waiting state to the ready state.
4. Process finished its execution and terminated.

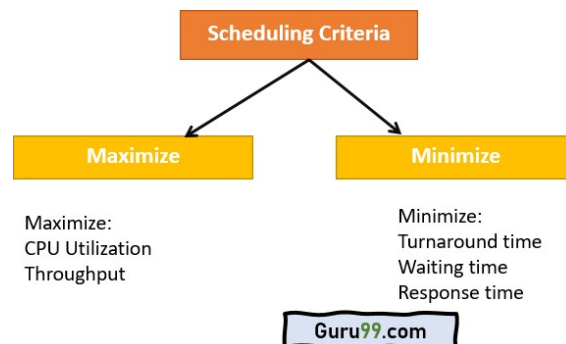**Only conditions 1 and 4 apply, the scheduling is called non-preemptive. All**

other scheduling are preemptiv

ImportantCPUschedulingTerminologies

- **BurstTime/ExecutionTime:**It is a time required by the process to complete execution. It is also called running time.
- **Arrival Time:**when a process enters in a ready state
- **FinishTime:**when process complete and exit from a system
- **Multiprogramming:**A number of programs which can be present in memory at the same time.
- **Jobs:**It is a type of program without any kind of user interaction.
- **User:**It is a kind of program having user interaction.
- **Process:**It is the reference that is used for both job and user.
- **CPU/IOburstcycle:**Characterizes process execution, which alternates between CPU and I/O activity. CPU times are usually shorter than the time of I/O.

CPUSchedulingCriteria

A CPU scheduling algorithm tries to maximize and minimize the following:

Maximize:
CPU Utilization
Throughput

Minimize:
Turnaround time
Waiting time
Response time

Guru99.com

Maximize:

CPU utilization: CPU utilization is the main task in which the operating system needs to make sure that CPU remains as busy as possible. It can range from 0 to 100 percent. However, for the RTOS, it can be range from 40 percent for low-level and 90 percent for the high-level system.

Throughput: The number of processes that finish their execution per unit time is known Throughput. So, whentheCPUis busyexecutingtheprocess, at that time, work is beingdone, and the work completed per unit time is called Throughput.

Minimize:

Waitingtime: Waitingtimeisanamount that specificprocess needs towaitintheready queue.

Responsetime: It is an amount totimein which therequest was submitted untilthe first response is produced.

TurnaroundTime: Turnaroundtimeis anamount of timetoexecuteaspecific process. It isthe calculation of the total time spent waiting to get into the memory, waiting in the queue and, executing on the CPU. The period between the time of process submission to the completion time is the turnaround time.

IntervalTimer

Timer interruptionis amethodthat is closelyrelatedtopreemption.Whenacertainprocess gets the CPU allocation, a timer may be set to a specified interval. Both timer interruption and preemption force a process to return the CPU before its CPU burst is complete.

Most of themulti-programmedoperatingsystem uses someform of atimertoprevent aprocess from tying up the system forever.

WhatisDispatcher?

It is amodule that provides control of the CPUto the process. The Dispatcher should be fastso that it can run on every context switch. Dispatch latency is the amount of time needed by the CPU scheduler to stop one process and start another.

Functions performedbyDispatcher:

- ContextSwitching
- Switchingtousermode
- Movingtothecorrectlocationinthenewlyloadedprogram.

TypesofCPUschedulingAlgorithm

Therearemainlysixtypes ofprocess schedulingalgorithms

1. FirstComeFirstServe(FCFS)
2. Shortest-Job-First(SJF)Scheduling
3. ShortestRemainingTime
4. PriorityScheduling
5. RoundRobinScheduling
6. MultilevelQueueScheduling

FIRSTCOMEFIRSTSERVE (FCFS)

- Jobsareexecutedonfirstcome,firstservebasis.
- Itisanon-preemptive,pre-emptiveschedulingalgorithm.
- Easytounderstandandimplement.
- ItsimplementationisbasedonFIFOqueue.
- Poorinperformanceasaveragewaittimeishigh.

Process	Arrival Time	Execute Time	Service Time
P0	0	5	0
P1	1	3	5
P2	2	8	8
P3	3	6	16
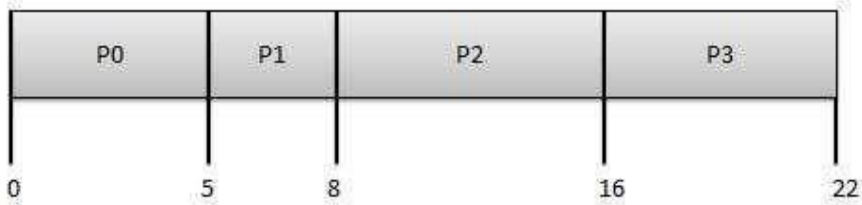
Waittimeofeachprocessisasfollows −

Process	WaitTime:ServiceTime-ArrivalTime
P0	0 - 0= 0

|	
P1	5 - 1= 4
P2	8 - 2= 6
P3	16- 3=13

AverageWaitTime:(0+4+6+13)/4= 5.75

SHORTEST JOBNEXT(SJN)

- Thisisalsoknownas**shortestjobfirst**,or SJF
- Thisisanon-preemptive,pre-emptiveschedulingalgorithm.
- Bestapproachtominimizewaitingtime.
- EasytoimplementinBatchsystemswhererequiredCPUtimeisknowninadvance.
- ImpossibletoimplementininteractivesystemswhererequiredCPUtimeisnotknown.
- Theprocessershould knowinadvancehowmuchtimeprocesswilltake.

Given: Table of processes, and their Arrival time, Execution time

Process	ArrivalTime	ExecutionTime	ServiceTime
P0	0	5	0
P1	1	3	5
P2	2	8	14
P3	3	6	8

Process	Arrival Time	Execute Time	Service Time
P0	0	5	3
P1	1	3	0
P2	2	8	16
P3	3	6	8

P1	P0	P3	P2
0 3	8	16	22
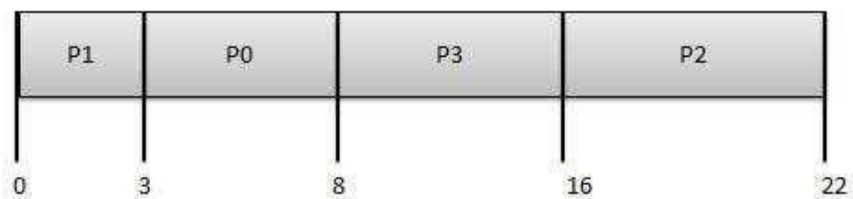
Waitingtimeofeachprocessisasfollows−

Process	WaitingTime
P0	0 - 0= 0
P1	5 - 1= 4
P2	14- 2=12
P3	8 - 3= 5

AverageWaitTime:(0 +4+12 + 5)/4=21 /4 = 5.25

## PRIORITYBASEDSCHEDULING

- Priorityschedulingisanon-preemptivealgorithm andoneofthemostcommonschedulingalgorithms in batch systems.

- Eachprocessisassignedapriority.Processwithhighestpriorityistobeexecutedfirstandsoon.

- Processeswithsamepriorityareexecutedonfirstcomefirstservedbasis.

- Prioritycanbedecided basedonmemoryrequirements,timerequirements oranyotherresource requirement.

Given:Table ofprocesses,andtheirArrivaltime,Executiontime,andpriority.Here weareconsidering1is the lowest priority.

Process	ArrivalTime	ExecutionTime	Priority	ServiceTime
P0	0	5	1	0
P1	1	3	2	11
P2	2	8	1	14
P3	3	6	3	5

Process	Arrival Time	Execute Time	Priority	Service Time
P0	0	5	1	9
P1	1	3	2	6
P2	2	8	1	14
P3	3	6	3	0

P3	P1	P0	P2
0	6	9	14 22
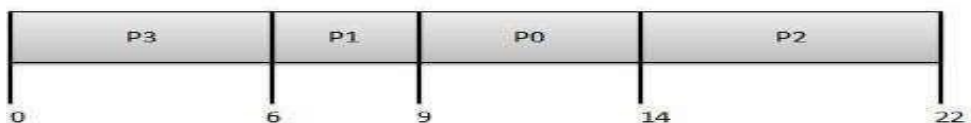
Waitingtimeofeachprocessisasfollows−

Process	WaitingTime
P0	0 - 0= 0
P1	11- 1=10
P2	14- 2=12
P3	5 - 3= 2

AverageWaitTime:(0 +10+12+2)/4=24/ 4=6

SHORTESTREMAININGTIME

- Shortestremainingtime(SRT)isthepreemptiveversionoftheSJNalgorithm.
- Theprocessoris allocatedtothe jobclosest tocompletion but itcanbe preempted bya newerready job with shorter time to completion.
- ImpossibletoimplementininteractivesystemswhererequiredCPUtimeisnotknown.
- Itisoftenusedinbatchenvironmentswhereshortjobsneedtogivepreference.

ROUNDROBINSCHEDULING

- RoundRobinisthepreemptiveprocessschedulingalgorithm.
- Eachprocessisprovidedafixtimetoexecute,itiscalleda**quantum**.
- Onceaprocess is executedforagiventimeperiod, it is preemptedandother process executes fora given time period.
- Contextswitchingisusedtosavestatesofpreemptedprocesses.

Quantum = 3
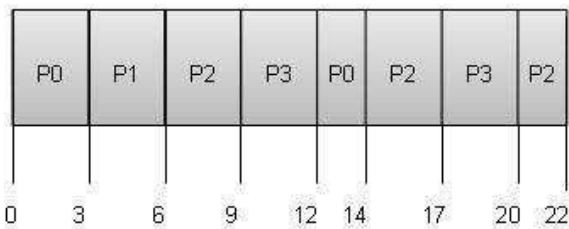
Waittimeofeachprocessisasfollows −

Process	WaitTime:ServiceTime-ArrivalTime
P0	(0-0) + (12- 3)= 9
P1	(3-1) =2

P2	(6- 2)+(14 -9)+ (20-17)=12
P3	(9- 3)+(17 -12)=11

AverageWaitTime:(9+2+12+11)/4=8.5

MULTIPLE-LEVELQUEUESSCHEDULING

Multiple-levelqueuesarenotanindependentschedulingalgorithm.Theymakeuseofotherexistingalgorithms   to group and schedule jobs with common characteristics.

- Multiplequeuesaremaintainedforprocesseswithcommoncharacteristics.
- Eachqueuecanhaveitsownschedulingalgorithms.
- Prioritiesareassignedtoeach queue.

For example, CPU-bound jobs can be scheduled in one queue and all I/O-bound jobs in another queue. The Process Scheduler then alternately selects jobs from each queue and assigns them to the CPU based on the algorithm assigned to the queue.

ConcurrencyinOperatingSystem

Inthisarticle, youwill learnthe concurrency in the operatingsystemwith itsprinciples, issues,advantagesand disadvantages.

WhatisConcurrency?

It refers to the execution of multiple instruction sequences at the same time.It occurs in an operating system when multiple process threadsare executing concurrently. These threads can interact with one another via shared memory or message passing. Concurrency results in resource sharing, which causes issues like deadlocks and resource scarcity. It aidswithtechniques such as process coordination, memory allocation,and execution schedule to maximizethroughput.

PrinciplesofConcurrency

Today'stechnology,likemulti-coreprocessorsandparallelprocessing,allowsmultipleprocessesandthreadsto be executed simultaneously. Multiple processes and threads can access the same memory space, the same declaredvariableincode,orevenreador write tothesamefile.

Theamount of timeit takes a process to execute cannot be simplyestimated, and you cannot predictwhich process will complete first, enabling you to build techniques to deal with the problems that concurrency creates.

Interleaved and overlapping processes are two types of concurrent processes with the same problems.It is impossibletopredicttherelativespeedofexecution,andthefollowingfactorsdetermineit:

1. **Thewayoperatingsystemhandlesinterrupts**

2. **Otherprocesses'activities**
3. **Theoperatingsystem'sschedulingpolicies**

ProblemsinConcurrency

Therearevariousproblemsinconcurrency.Someofthemareasfollows:

1. Locatingtheprogrammingerrors

It's difficult to spot a programming error because reports are usually repeatable due to the varying states of shared components each time the code is executed.

2. SharingGlobalResources

Sharingglobalresourcesisdifficult.Iftwoprocessesutilizeaglobalvariableandbothalterthevariable'svalue, theorderinwhichthe manychangesareexecutedis critical.

3. Lockingthechannel

ItcouldbeinefficientfortheOStolocktheresourceandpreventotherprocessesfromusingit.

4. OptimalAllocationofResources

ItischallengingfortheOStohandleresourceallocationproperly.

IssuesofConcurrency

Variousissuesofconcurrencyareasfollows:

1. Non-atomic

Operations that are non-atomic but interruptible by several processes may happen issues. A non-atomic operation depends on other processes, and an atomic operation runs independently of other processes.

2. Deadlock

In concurrent computing, itoccurswhenonegroupmemberwaitsforanothermember, including itself, to send a messageand releasea lock.Software and hardware locks are commonly usedto arbitrate shared resources andimplementprocesssynchronizationinparallelcomputing,distributedsystems,andmultiprocessing.

3. Blocking

A blocked process is waiting for some event, like the availability of a resource or completing an I/O operation. Processes may blockwaiting for resources, and a process may be blockedfor a long time waitingfor terminal input.Iftheprocessisneededtoupdatesomedataperiodically,itwillbeveryundesirable.

4. RaceConditions

A race problem occurs whenthe output of a software application is determined bythe timing or sequencing of other uncontrollable events. Race situations can also happen in multithreadedsoftware, runs in a distributed environment, or is interdependent on shared resources.

5. Starvation

A problem in concurrent computing is where a process is continuously denied the resources it needsto completeitswork.It couldbecausedbyerrorsinscheduling ormutualexclusion algorithm, butresourceleaks may also causeit.

Concurrent system design frequently requires developing dependable strategies for coordinating their execution, data interchange, memory allocation, and execution schedule to decrease response time and maximize throughput.

AdvantagesandDisadvantagesofConcurrencyinOperatingSystem

VariousadvantagesanddisadvantagesofConcurrencyinOperatingsystemsareasfollows:

Advantages

1. BetterPerformance

It improves the operating system's performance.When one application only utilizes the processor, and another only uses the disk drive, the time it takesto perform both apps simultaneously is lessthanthe timeit takes to run themsequentially.

2. BetterResourceUtilization

Itenablesresourcesthatarenotbeingusedbyoneapplicationtobeusedbyanother.

3. RunningMultipleApplications

Itenablesyoutoexecutemultipleapplicationssimultaneously.

Disadvantages
1. Itisnecessarytoprotectmultipleapplicationsfromeachother.
2. Itisnecessarytouseextratechniquestocoordinateseveralapplications.
3. AdditionalperformanceoverheadsandcomplexitiesinOSareneededforswitchingbetween applications.

PROCESSSYNCHRONIZATION

Inthistutorial,wewillbecoveringtheconceptofProcesssynchronizationinanOperatingSystem.   Process

Synchronizationwas   introducedto   handle   problemsthat   arose   while   multiple   processexecutions.

Processiscategorizedintotwotypesonthebasisofsynchronizationandthesearegivenbelow:

- IndependentProcess
- CooperativeProcess

IndependentProcesses

Twoprocessesaresaidtobeindependentiftheexecutionofoneprocessdoesnotaffecttheexecutionof another process.

CooperativeProcesses

Twoprocessesaresaidtobecooperativeiftheexecutionofoneprocessaffectstheexecutionofanother process.Theseprocessesneedtobesynchronizedsothattheorderofexecutioncanbeguaranteed.

Process Synchronization

Itisthetaskphenomenonofcoordinatingtheexecutionofprocessesinsuchawaythatnotwoprocessescan have access to the same shared data and resources.

- Itis aprocedure that isinvolvedinordertopreservetheappropriateorderofexecutionofcooperative processes.
- Inordertosynchronizetheprocesses,therearevarioussynchronizationmechanisms.
- ProcessSynchronizationismainlyneededinamulti-processsystemwhenmultipleprocessesare runningtogether,andmore thanoneprocessestrytogainaccesstothesamesharedresourceorany data at the sametime.

RaceCondition

Atthetimewhenmorethanoneprocessiseitherexecutingthesamecodeoraccessingthesamememoryor anysharedvariable;Inthatcondition,thereisapossibilitythattheoutputorthevalueofthesharedvariableis wrongsoforthatpurposealltheprocessesaredoingtheracetosaythatmyoutputiscorrect.Thiscondition iscommonlyknownasaracecondition. Asseveralprocessesaccessandprocessthemanipulationsonthe samedata ina concurrent manner anddue to which the outcomedependsonthe particular order inwhich the accessofdatatakesplace.

Mainly this condition is a situation that may occur inside the critical section. Race condition in the critical sectionhappenswhentheresultofmultiplethreadexecutiondiffersaccordingtotheorderinwhichthe threadsexecute.Butthis condition is criticalsections can beavoided if the criticalsection is treatedas an atomicinstruction.Properthreadsynchronizationusinglocksoratomicvariablescanalsopreventrace conditions.

CriticalSectionProblem

ACriticalSectionisacodesegmentthataccessessharedvariablesandhastobeexecutedasanatomicaction. Itmeansthatinagroupofcooperatingprocesses,atagivenpointoftime,onlyoneprocessmustbe executing its critical section.If any other processalso wants to execute its critical section, it must wait until thefirstonefinishes.Theentrytothecriticalsectionismainlyhandledbywait()functionwhiletheexitfromthe critical section is controlled by the signal()function.

```
do {                            controls the entry into critical
                                section and gets a LOCK on
                                required resources

        entry section    ◄──────┘

            critical section  ◄──────  the critical part
removes the LOCK
from the resources
and let the others  ──────►  exit section
know that its critical
section is over
                remainder section  ◄──── rest of the section

} while (TRUE);
```
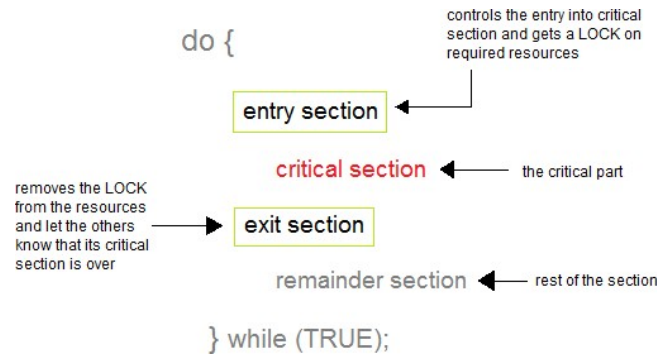
EntrySection

Inthissectionmainlytheprocessrequestsforitsentryinthecriticalsection.

ExitSection

Thissectionisfollowedbythecriticalsection.

ThesolutiontotheCriticalSectionProblem

Asolutiontothecriticalsectionproblemmustsatisfythefollowingthreeconditions:

1. MutualExclusion

Outofagroupofcooperatingprocesses,onlyoneprocesscanbeinitscriticalsectionatagivenpointof time.

2. Progress

Ifnoprocessisinitscriticalsection,andifoneormorethreadswanttoexecutetheircriticalsectionthenany oneofthesethreadsmustbeallowedtogetinto itscriticalsection.

3. BoundedWaiting

Afteraprocessmakesarequestforgetting into itscriticalsection,thereisalimit forhowmanyother processescanget intotheir criticalsection, beforethis process's request isgranted.So afterthelimit is reached,thesystemmustgranttheprocesspermissiontogetintoitscriticalsection.

SolutionsfortheCriticalSection

ThecriticalsectionplaysanimportantroleinProcessSynchronizationsothattheproblemmustbesolved.

Somewidelyusedmethodtosolvethecriticalsectionproblemareasfollows:

1. Peterson'sSolution

Thisiswidelyusedandsoftware-basedsolutiontocriticalsectionproblems.Peterson'ssolutionwasdeveloped by a computer scientist Peterson that's why it is named so.

Withthehelpofthissolutionwheneveraprocessisexecutinginanycriticalstate,thentheotherprocessonly executes therestofthe code,andvice-versacanhappen. Thismethodalsohelpstomakesureofthethingthat onlyasingleprocesscanruninthecriticalsectionataspecifictime.

Thissolutionpreservesallthreeconditions:

- MutualExclusioniscomfortedasatanytimeonlyoneprocesscanaccessthecritical section.
- Progressisalsocomforted,asa processthatis outsidethecriticalsectionis unabletoblockother processes from entering into the critical section.
- BoundedWaitingisassuredaseveryprocessgetsafairchancetoentertheCritical section.

Theaboveshowsthestructure ofprocessPi in Peterson's solution.

- SupposethereareNprocesses(P1,P2, ... PN) andasatsomepointoftimeeveryprocessrequiresto enter in the CriticalSection
- A FLAG[] arrayofsizeN ismaintained herewhich isbydefault false. Whenever aprocessrequires to enterinthecriticalsection,ithastosetitsflagastrue.Example:IfPiwantstoenteritwill setFLAG[i]=TRUE.
- Anothervariableiscalled TURN andisusedto indicatetheprocessnumber thatiscurrently waitingto enterintothecriticalsection.
- TheprocessthatentersintothecriticalsectionwhileexitingwouldchangetheTURN toanother number from the list of processes that are ready.
- Example:Iftheturnis3thenP3enterstheCritical sectionandwhileexitingturn=4andthereforeP4 breaksoutofthewaitloop.

SynchronizationHardware

Manysystemsprovidehardwaresupportforcriticalsectioncode.Thecriticalsectionproblemcouldbesolvedeasilyinasingle-processorenvironmentifwecoulddisallowinterruptstooccurwhileasharedvariableor resourceisbeingmodified.

Inthismanner,wecouldbesurethatthecurrentsequenceofinstructionswouldbeallowedtoexecutein orderwithoutpre-emption.Unfortunately,thissolutionisnotfeasibleinamultiprocessorenvironment.

Disablinginterruptonamultiprocessorenvironmentcanbetime-consumingasthemessageispassedtoall theprocessors.

Thismessagetransmissionlagdelays theentry of threadsintothecriticalsection,andthesystemefficiency decreases.

MutexLocks

Asthesynchronizationhardwaresolutionisnoteasytoimplementforeveryone,astrictsoftwareapproach calledMutexLockswasintroduced.Inthisapproach,intheentrysectionofcode,aLOCKisacquiredoverthe criticalresourcesmodifiedandusedinsidethecriticalsection,andintheexitsectionthatLOCKisreleased.

Astheresourceislockedwhileaprocessexecutesitscriticalsectionhencenootherprocesscanaccess it.

SEMAPHORES

In1965,Dijkstraproposedanewandverysignificanttechniqueformanagingconcurrentprocessesbyusing thevalueofasimpleintegervariabletosynchronizetheprogressofinteractingprocesses.Thisintegervariable iscalledasemaphore.Soitisbasicallyasynchronizingtoolandisaccessedonlythroughtwolowstandard atomic operations,wait and signal designatedby P(S) and V(S)respectively.

Semaphoresareintegervariablesthatareusedtosolvethecriticalsectionproblembyusingtwo atomic operations, wait and signal that are used for process synchronization.

In very simple words, the semaphoreis a variable that can hold only a non-negative Integer value, shared betweenallthethreads,withoperationswaitandsignal,whichworkasfollow:

```
P(S):ifS>=1thenS:=S-1

     else<blockandenqueuetheprocess>;



V(S):if<someprocessisblockedonthequeue> then

     <unblock a process>

     elseS:=S+1;
```

Copy

Theclassicaldefinitionsofwaitandsignalare:

- **Wait:Thisoperationdecrementsthevalueofitsargument**S**,assoonasitwouldbecomenonnegative(greater than orequal to** 1**). ThisOperation mainly helpsyouto control the entryof atask into thecriticalsection.Inthecaseofthenegativeorzerovalue,nooperationisexecuted.**wait()**operation wasoriginallytermedasP;soitisalsoknownas**P(S)operation.**Thedefinitionofwaitoperationisas follows:**

```
wait(S)

{

    while(S<=0);//nooperation

    S--;

}
```

Note:

When oneprocessmodifiesthevalueofasemaphorethen, nootherprocesscan simultaneouslymodifythat same semaphore's value.In the above case the integervalue ofS(S<=0)as wellasthe possible modification that is S-- mustbe executed without any interruption.

- **Signal:** Increments thevalue ofits argument S**, as there isno moreprocess blocked on the queue. This Operationismainlyusedtocontroltheexitofataskfromthecriticalsection.**signal()** operationwas originallytermedasV;soitisalsoknownas**V(S)operation.**Thedefinitionofsignaloperationisas follows:**

```
signal(S)

{

S++;

}
```

Also, note that all the modificationsto the integer value of semaphore in thewait()and signal()operations must be executedindivisibly.

PropertiesofSemaphores

1. **It'ssimpleandalwayshaveanon-negativeintegervalue.**
2. **Workswithmany processes.**
3. **Canhavemanydifferentcriticalsectionswithdifferentsemaphores.**
4. **Eachcriticalsectionhasuniqueaccesssemaphores.**
5. **Canpermitmultipleprocessesintothecriticalsectionatonce,ifdesirable.**

WewillnowcoverthetypesofsemaphoresintheOperatingsystem;

TypesofSemaphores

SemaphoresaremainlyoftwotypesinOperatingsystem:

1. **BinarySemaphore:**

   Itisaspecialformofsemaphoreusedforimplementingmutualexclusion,henceitisoftencalled aMutex.Abinarysemaphoreisinitializedto1 andonlytakesthevalues0 and1 duringtheexecution of aprogram. In Binary Semaphore, the wait operation works onlyif the value of semaphore = 1, and the signal operation succeeds whenthe semaphore= 0. Binary Semaphores are easier to implement thancountingsemaphores.

2. **CountingSemaphores:**

   Theseareusedtoimplement bounded concurrency.TheCountingsemaphores canrangeover anunrestricteddomain.Thesecanbeusedtocontrolaccesstoagivenresourcethatconsistsofa finitenumberofInstances.Herethesemaphorecountisusedtoindicatethenumberofavailable resources.Iftheresourcesareaddedthenthesemaphorecountautomaticallygetsincrementedandif theresourcesareremoved,thecountisdecremented.CountingSemaphorehasnomutualexclusion.

ExampleofUse

Hereisasimplestep-wiseimplementationinvolvingdeclarationandusageofsemaphore.

```
Sharedvarmutex:semaphore=1;

Processi

   begin

   .

   .

   P(mutex);

   executeCS;

   V(mutex);

   .

   .

   End;
```

AdvantagesofSemaphores

BenefitsofusingSemaphoresareasgivenbelow:

- Withthehelpofsemaphores,thereisaflexiblemanagementofresources.

- Semaphores are machine-independent and they should be run in the machine-independent code of the microkernel.
- Semaphoresdonotallowmultipleprocessestoenterinthecriticalsection.
- Theyallowmorethanonethreadtoaccessthecriticalsection.
- Assemaphoresfollowthemutualexclusionprinciplestrictlyandthesearemuchmoreefficientthan some other methods ofsynchronization.
- Nowastageofresources in semaphores becauseofbusywaiting insemaphores asprocessortimeis not wastedunnecessarilyto check if any conditionis fulfilledin order toallow a process to accessthe critical section.

DisadvantagesofSemaphores

- Oneofthebiggestlimitationsisthatsemaphoresmayleadtopriorityinversion;wherelowpriority processes may access the critical section first and high priority processes may access the critical sectionlater.
- To avoiddeadlocks inthesemaphore,theWaitandSignaloperationsarerequiredtobeexecutedin the correct order.
- Usingsemaphoresatalargescale   isimpractical;astheiruseleadstolossofmodularity   andthis happensbecausethe wait()andsignal()operations prevent thecreation of thestructuredlayoutforthe system.
- Theiruseisnotenforcedbutisbyconventiononly.
- Withimproper use, a processmay block indefinitely. Sucha situation is calledDeadlock. We will be studying deadlocks in detail in coming lessons.

UNIT-

3MEMORYMANAGEMEN

T

MemoryallocationTechniques
   1. Contiguousmemoryallocation

2. **Non-contiguousmemoryallocation**
Swapping
Paging,Segmentation,virtualmemoryusingpaging, Demand paging, page fault handling.

MEMORYMANAGEMENT

Memory is the important part of the computer that is used to store the data. Its management iscriticaltothecomputersystembecausetheamountofmainmemoryavailableinacomputer system is very limited. At any time, many processes are competing for it. Moreover, to increase performance, several processes are executed simultaneously. For this, we must keep several processes in the main memory, so it is even more important to manage them effectively.

Memory Management in OS
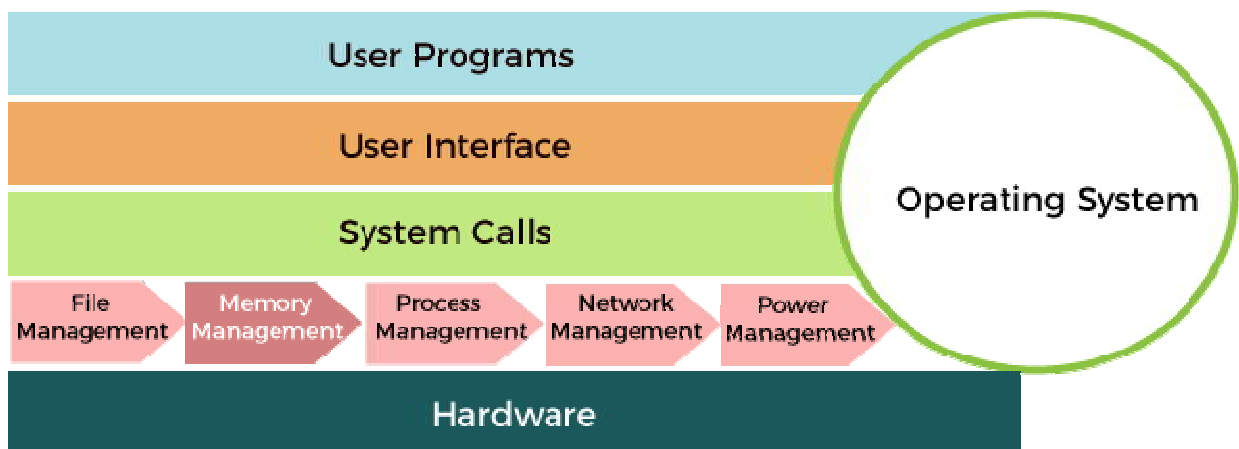
in a uni-programming system, main memory is divided into two parts: one part for the operatingsystem(residentmonitor,kernel)andonepartfortheuserprogramcurrentlybeing executed.

In a multiprogramming system, the "user" part of memory must be further subdivided to accommodate multiple processes. The task of subdivision is carried out dynamically by the operating system and is known as **memory management**

BINDINGOFINSTRUCTIONSANDDATATOMEMORY

Addressbindingof instructions anddata to memoryaddresses canhappenatthreedifferent stages.

1. **Compiletime:**The compile timeisthetime takento compile the program orsource code. During compilation, if memory location known a priori, then it generates absolute codes.

2. **Load time**: It is the timetaken to link all related program file and load into the main memory.Itmust generate relocatable codeif memorylocationisnotknown atcompile time.

3. **Executiontime**:Itisthetimetakentoexecutetheprograminmainmemorybyprocessor. Binding delayed until run time if the process can be moved during its execution from one memory segment to another. Need hardware support for address maps (e.g., base and limit registers).
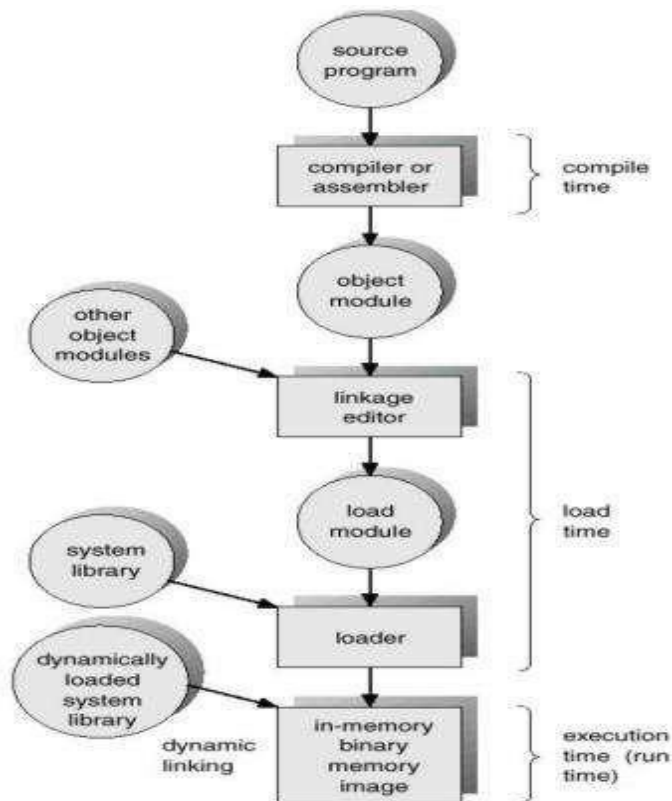
(Multistepprocessingofauserprogram.)

LOGICAL-VERSUSPHYSICAL-ADDRESSSPACE

⇒An address generated by the CPU is commonly referred to as a logical address or a virtual address whereas anaddressseen bythemainmemoryunit is commonlyreferred to asa physicaladdress.

⇒The set of all logical addresses generated by a program is a logical-address space whereasthe setof allphysicaladdressescorrespondingtotheselogicaladdressesisa physicaladdressspace.

⇒ Logical and physical addresses arethe samein compile-time and load-time addressbindingschemes;logical(virtual)andphysicaladdressesdifferinexecution-time addressbindingscheme.

⇒The Memory Management Unit is a hardware device that maps virtual to physical address.In MMU scheme,thevalue in the relocation register is added to everyaddress generatedbyauser process atthetimeitis senttomemoryasfollows:
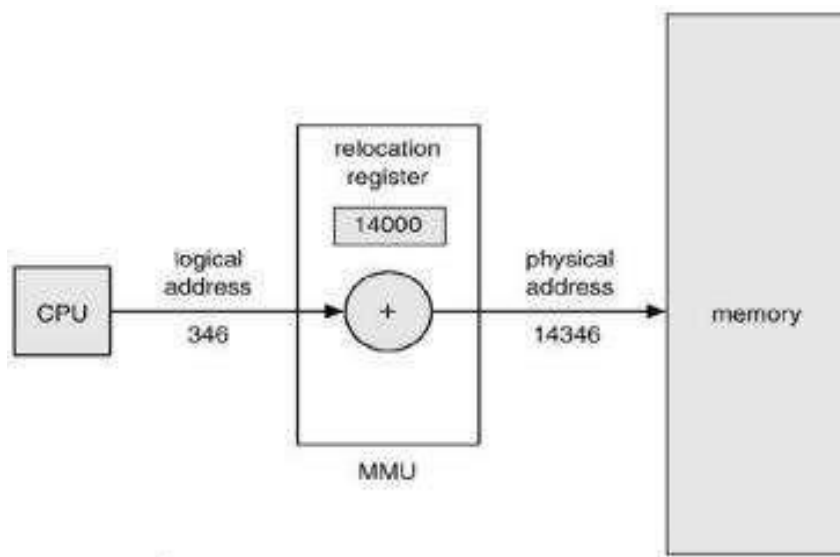
(Dynamicrelocationusingarelocationregister)

DYNAMICLOADING

⇒Itloadstheprogramanddatadynamicallyintophysicalmemorytoobtain better memory-space utilization.

⇒Withdynamicloading,aroutineisnotloadeduntilitiscalled.

⇒Theadvantageofdynamicloadingisthatanunusedroutineisneverloaded.

⇒Thismethodisusefulwhenlargeamountsof codeareneededtohandleinfrequently occurring cases, such as error routines.

⇒Dynamicloadingdoesnotrequirespecialsupportfromtheoperatingsystem.

DYNAMIC LINKING

⇒Linkingpostponeduntilexecutiontime.

⇒Smallpieceofcode(stub)usedtolocatetheappropriatememory-residentlibraryroutine.

⇒Stubreplacesitselfwiththeaddressoftheroutineandexecutestheroutine.

⇒Operatingsystemneededtocheckifroutineisinprocessesmemoryaddress.

⇒Dynamiclinkingisparticularlyusefulforlibraries. OVERLAYS

⇒Keepinmemoryonlythoseinstructionsanddatathatareneededatanygiven time.

⇒Neededwhenprocessislargerthanamountofmemoryallocatedtoit.

⇒Implementedbyuser,nospecialsupportneededfromoperatingsystem,programming design of overlay structure is complex.

⇒A process can be swapped temporarily out of memory to a backing store (large disc), and then brought back into memory for continued execution.

⇒**Roll out, roll in**: A variant of this swapping policy is used for priority-based scheduling Algorithms. If a higher-priority process arrives and wants service, the memory manager can swapoutthelower-priorityprocesssothatitcanloadandexecutethehigher-priorityprocess. When the higher-priority processfinishes, thelower-priority process can be swapped back in and continued. This variant of swapping is called roll out, roll in.

⇒Major part of swap time is transfer time; total transfer time is directly proportional to the amount of memory swapped.

⇒Modifiedversionsofswappingarefoundonmanysystems(UNIX,Linux,andWindows).
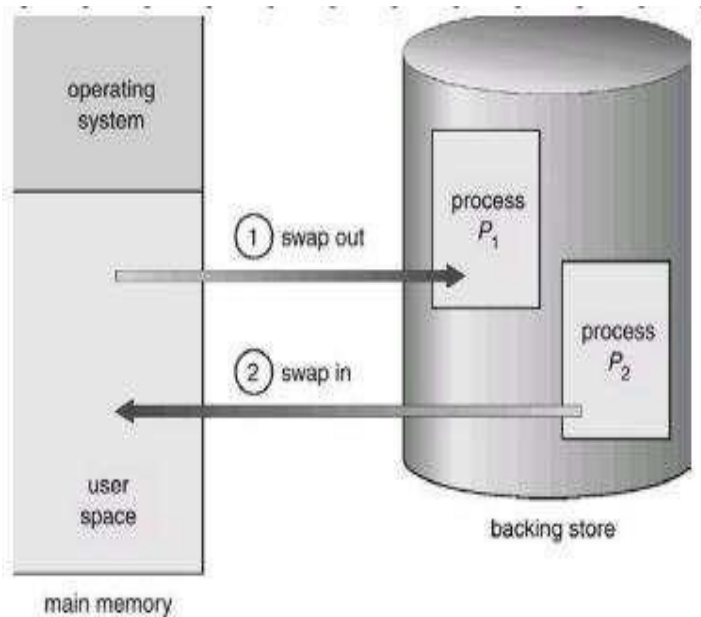
(SchematicViewof Swapping

MEMORYMANAGEMENTTECHNIQUES:

The Memorymanagement Techniquescanbe classified intofollowing main categories:

- o Contiguousmemorymanagementschemes
- o Non-Contiguousmemorymanagementschemes
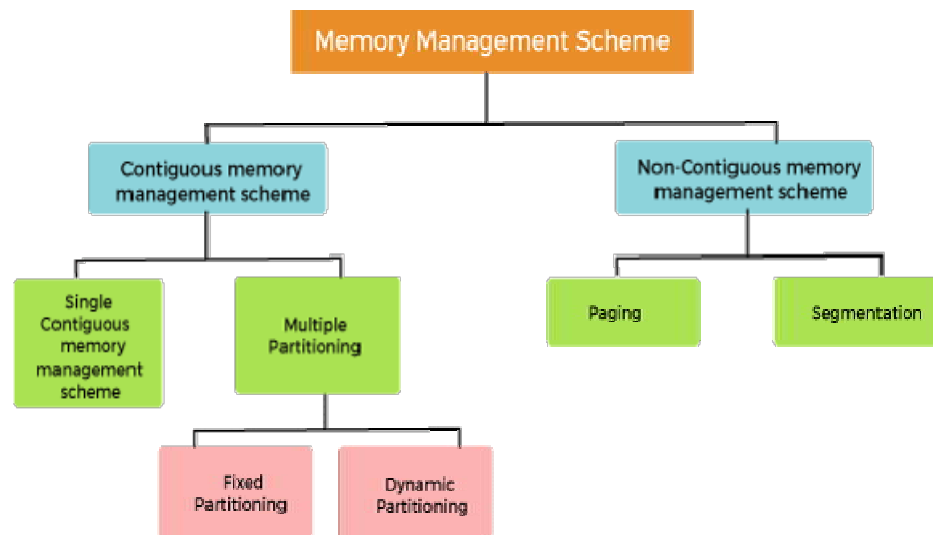
Classification of memory management schemes

CONTIGUOUSMEMORYMANAGEMENTSCHEMES:

In a Contiguousmemorymanagementscheme,eachprogram occupiesasinglecontiguous block of storage locations, i.e., a set of memory locations with consecutive addresses.

SINGLECONTIGUOUSMEMORYMANAGEMENT SCHEMES:

The Single contiguous memory management scheme is the simplest memory management scheme used in the earliest generation of computer systems. In this scheme, the main memory is divided into two contiguous areas or partitions. The operating systems reside permanently in one partition, generally at the lower memory, and the user process is loaded into the other partition.

Advantagesof Singlecontiguous memorymanagementschemes:

- o   Simpletoimplement.
- o   Easytomanageanddesign.
- o   In a Single contiguousmemorymanagementscheme,oncea process is loaded, itis given full processor's time, and no other processor will interrupt it.

DisadvantagesofSinglecontiguousmemorymanagementschemes:

- o   Wastage of memoryspacedue tounusedmemoryas theprocess is unlikelyto use all the available memory space.
- o   The CPUremains idle,waitingforthe disk toloadthebinaryimage into themain memory.

- It cannot be executedif theprogram istoo large tofit theentireavailablemain memory space.
- Itdoesnotsupportmultiprogramming,i.e.,itcannothandlemultipleprograms simultaneously.

MULTIPLE PARTITIONING:

The single Contiguous memory management scheme is inefficient as it limits computers to executeonlyoneprogramatatimeresultinginwastageinmemoryspaceandCPUtime.The problem of inefficient CPU use can be overcome using multiprogramming that allows more than one program to run concurrently. To switch between two processes, the operating systems need to load both processes into the main memory. The operating system needs to divide the available main memoryinto multiple partsto load multiple processes into the main memory. Thus multiple processes can reside in the main memory simultaneously.

Themultiple partitioningschemescanbeoftwo types:

- FixedPartitioning
- DynamicPartitioning

FIXED PARTITIONING

The main memory is divided into several fixed-sized partitions in a fixed partition memory management scheme or static partitioning. These partitions can be of the same size or differentsizes.Eachpartitioncanholdasingleprocess.Thenumberofpartitionsdetermines the degree of multiprogramming, i.e., the maximum number of processes in memory. These partitions are made at the time of system generation and remain fixed after that.

AdvantagesofFixedPartitioningmemorymanagementschemes:

- Simpletoimplement.
- Easytomanageanddesign.

DisadvantagesofFixedPartitioningmemorymanagement schemes:

- Thisschemesuffersfrominternalfragmentation.
- Thenumberofpartitionsisspecifiedatthe timeofsystemgeneration.

Fixed Partitioning

(Contiguous memory allocation)
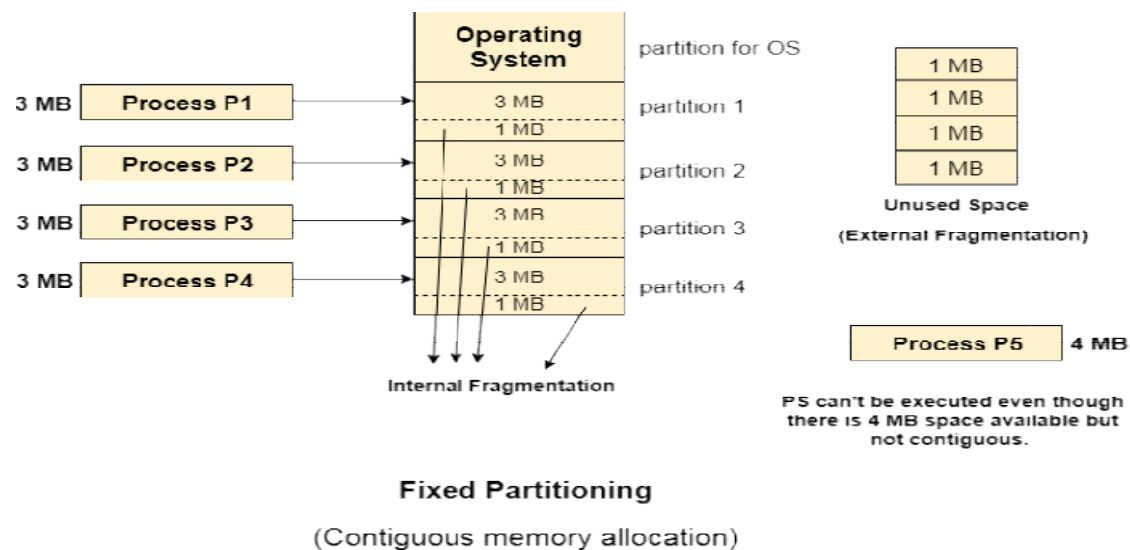
DYNAMICPARTITIONING

The dynamic partitioning was designed to overcome the problems of a fixed partitioning scheme. In a dynamic partitioning scheme, each process occupies only as much memory as they require when loaded for processing. Requested processes are allocated memory until the entire physical memory is exhausted or the remaining space is insufficient to hold the requestingprocess.Inthisschemethepartitionsusedareofvariablesize,and thenumberof partitions is not defined at the system generation time.

Dynamic Partitioning

(Process Size = Partition Size)
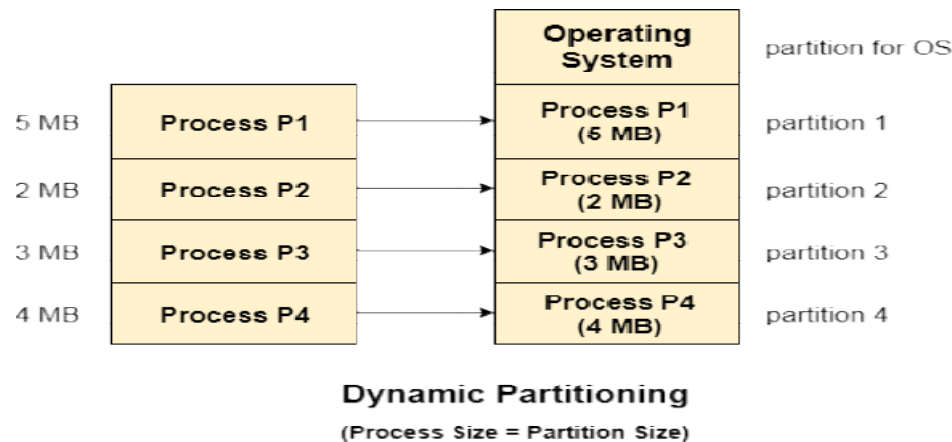
PARTITIONSELECTION POLICY:

When the multiple memory holes (partitions) are large enough to contain a process, the operatingsystem mustuseanalgorithm to select in which hole the processwillbe loaded.

Thepartitionselectionalgorithmareasfollows:

⇒ **First-fit:**TheOSlooksatallsectionsoffreememory.Theprocessisallocatedtothe firsthole found that is big enough size than the size of process.

⇒**NextFit:**ThenextfitsearchstartsatthelastholeallocatedandTheprocessisallocated tothe next hole found that is big enough size than the size of process.

⇒**Best-fit:**TheBestFitsearchestheentirelistofholestofindthesmallestholethatisbig enough size than the size of process.

⇒**Worst-fit**:TheWorstFitsearchestheentirelistof holes tofindthelargestholethatisbig enough size than the size of process.
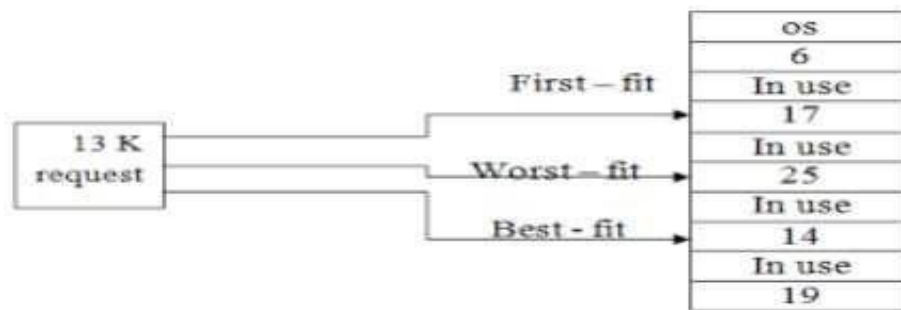
Fragmentation: Thewastingof memoryspace is calledfragmentation.Thereare two types of fragmentation as follows:

1. **External Fragmentation**: The total memory space exists to satisfy a request, but it is not contiguous.Thiswasted space not allocated to any partition iscalledexternalfragmentation. The external fragmentation can be reduce by compaction. The goal is to shuffle the memory contents to place all free memory together in one large block. Compaction is possible only if relocation is dynamic, and is done at execution time.

2. **Internal Fragmentation:** The allocated memory may be slightly larger than requested memory. The wasted space within a partition is called internalfragmentation. One method to reduce internal fragmentation is to use partitions of different size
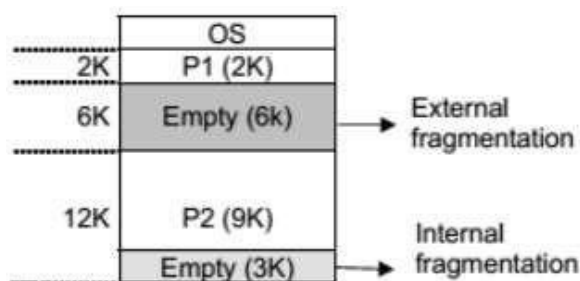
RE-ALLOCATIONOFMEMORY(OR)COMPACTION:

Thefollowingdiagramshowshowfragmentation can cause wasteof memoryanda compaction technique can be used to create more free memory out of fragmented memory −

Fragmented memory before compaction

Memory after compaction
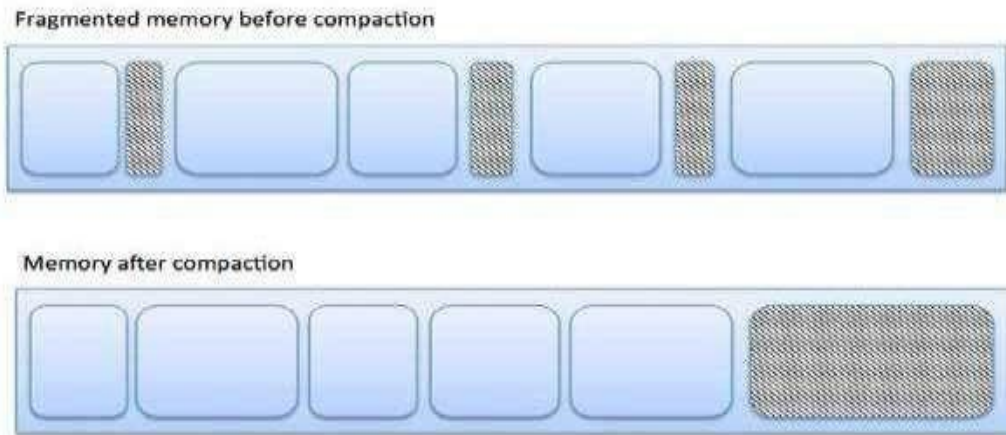
- Compactionisaprocessin which thefree space iscollected in a large memorychunk to make some space available for processes.
- Inmemorymanagement, swappingcreatesmultiplefragments inthememory because of the processes moving in and out.
- Compactionrefersto combiningalltheemptyspacestogetherandprocesses.
- Compaction helps to solve the problem of fragmentation, but it requires too much of CPU time. It moves all the occupied areas of store to one end and leaves one large free space for incoming jobs, instead of numerous small ones.
- In compaction, the system also maintains relocation information and it must be performed on each new allocation of job to the memory or completion of job from memory.
- Compactionisnot always easy. If relocation isstaticand isdone atassemblyorload time, compaction cannot be done. Compaction is possible only if relocation is dynamic and performed at execution time.

NON-CONTIGUOUSMEMORYMANAGEMENTSCHEMES:

In a Non-Contiguous memory management scheme, the program is divided into different blocksandloadedatdifferentportionsofthememorythatneednotnecessarilybeadjacent to one another. This scheme can be classified depending upon the size of blocks and whether the blocks reside in the main memory or not.

Whatispaging?

Paging is a technique that eliminates the requirements of contiguous allocation of main memory.Inthis,themainmemoryisdividedintofixed-sizeblocksofphysicalmemorycalled frames.Thesizeofaframeshouldbekeptthesameasthatofapagetomaximizethemain memory and avoid external fragmentation.

Advantagesof paging:

- o Pagesreduceexternalfragmentation.
- o Simpletoimplement.
- o Memoryefficient.
- o Duetotheequalsizeofframes,swapping becomesveryeasy.
- o Itisused forfasteraccessof data.

WhatisSegmentation?

Segmentation is a technique that eliminates the requirements of contiguous allocation of main memory. In this, the main memory is divided into variable-size blocks of physical memory called segments. It is based on the way the programmer follows to structure their programs.Withsegmentedmemoryallocation,eachjobisdividedintoseveralsegmentsof differentsizes,oneforeachmodule.Functions,subroutines,stack,array,etc.,areexamples of such modules.

PAGING

- In OperatingSystems,Pagingisa storagemechanism usedtoretrieveprocesses from the secondary storage into the main memory in the form ofpages.
- Themain ideabehindthe pagingisto divide each processintheform of **pages**. The main memory will also be divided in the form of **frames**.
- The size of aframe shouldbe **keptsame**asthatof a page tohavemaximum utilization of the main memory and to avoid external fragmentation
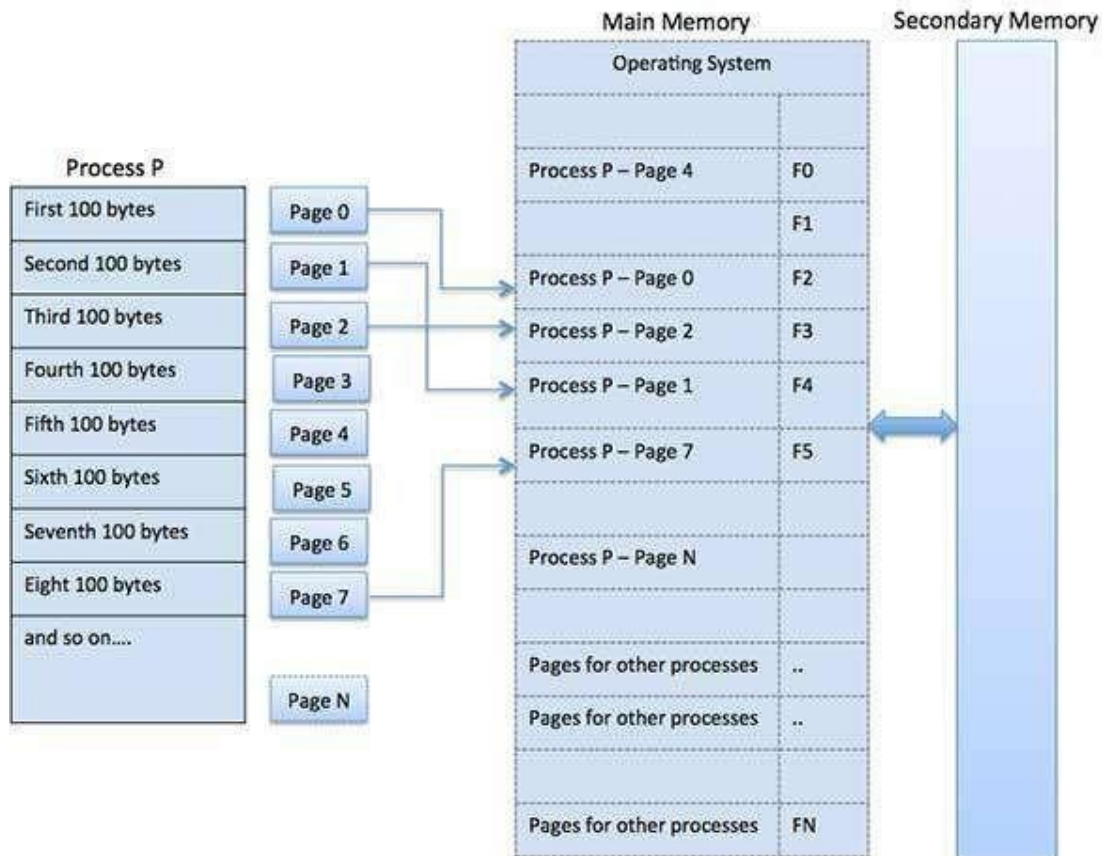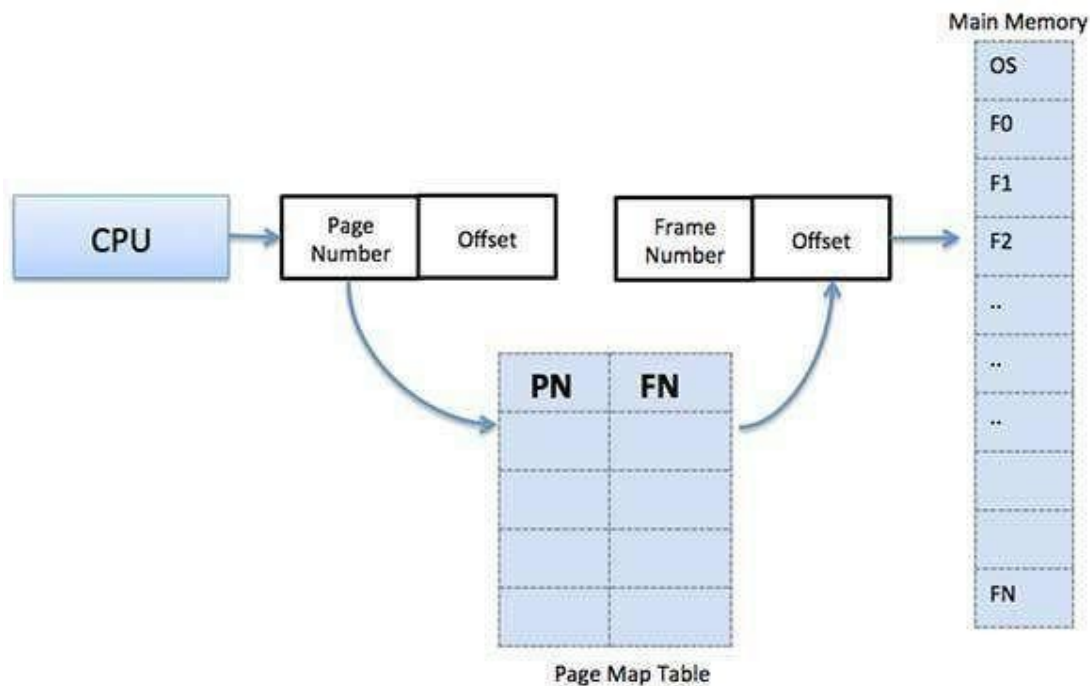

ADDRESSTRANSLATION

Page address is called **logical address** and represented by **page number**and the **offset**.

LogicalAddress=Pagenumber+pageoffset

Frameaddressis called **physicaladdress**and representedbya **framenumber**and the **offset**.

PhysicalAddress=Framenumber+pageoffset

A data structurecalled **pagemaptable** isused to keep trackof the relationbetweena page of a process to a frame in physical memory.

Page Map Table

When the system allocates a frame to any page, it translates this logical address into a physical address and create entry into the page table to be used throughout execution of the program.

When a process is to be executed, its corresponding pages are loaded into any available memoryframes.Supposeyouhaveaprogram of8Kbbutyourmemorycanaccommodate only 5Kb at a given point in time, then the paging concept will come into picture. When a computerrunsoutof RAM,theoperatingsystem (OS)willmoveidleorunwantedpagesof memory to secondary memory to free up RAM for other processes and brings them back when needed by the program.

This process continues during the whole execution of the program where the OS keeps removing idle pages from the main memory and write them onto the secondary memory and bring them back when required by the program.
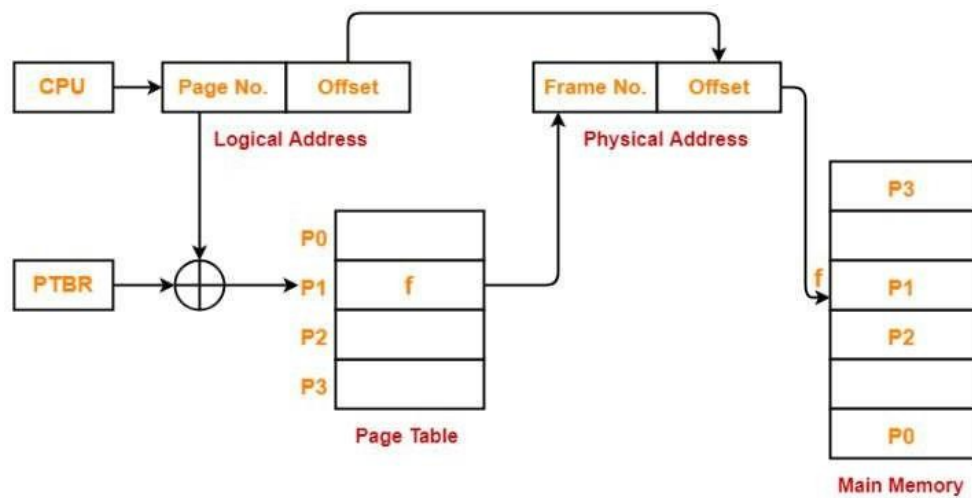
Translating Logical Address into Physical Address

- The paging model of memory is shown in below figure. The page size is defined by the hardware. The size of a page is typically of a power of 2, varying between 512 bytes and 16 MBper page, depending on the computer architecture. The selectionof apowerof 2asa page size makesthe translationof alogicaladdressintoapage number and page offset particularly easy.

- If the size of logical address is 2^m, and a page size is 2^n addressing units, then thehigh orderm-n bitsof alogical address designate thepage number, and then low order bits designate the page offset.
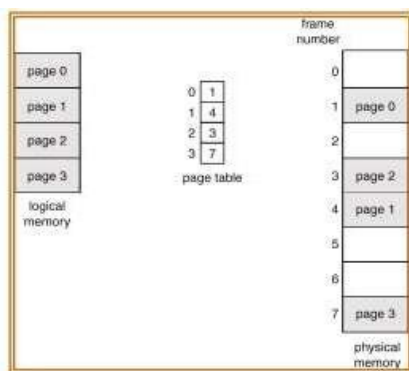
- Keeptrack ofallfreeframes.

- Torun aprogram ofsizenpages, needtofindn free framesandloadprogram.
- Setupapagetabletotranslatelogicaltophysicaladdresses.
- infernalfragmentationmayoccur.

- **Letus take anexample**.

Supposea program ofsize 4B.

Pagesize=2b

Then total no of page =2

Supposethememorysize=16b

Frame size =page size=2BThen

frame =8

PAGETABLE

F2	p0
F4	p1

p0

0	1
2	3

P1

F0		0	1	
F1		2	3	
F2		4	5	P0
F3		6	7	
F4		8	9	P1
F5		10	11	
F6		12	13	
F7		14	15	

CPUwillgeneratelogicaladdress

– letsitwant byte3ofthe processwhich is page no1

Cpu will generate 2 baddress 1b for page no+ 1bfor pageoffset So

byte 3 can be represent as 1 1 –i.e page no 1 and offset 1

In pagetablepage1 isinframe4 formula.

- Physicaladdress =page size oflogicalmemoryXframenumber+ offset
- Physicaladdress=2x4+1=9
- Sobyteno9

ADVANTAGESOFPAGING

• Pagingreducesexternalfragmentation

• Pagingissimple to implementand assumedasanefficientmemorymanagement technique.

• Duetoequalsizeofthe pagesandframes, swappingbecomesveryeasy.

DisadvantagesofPaging

• Pagetable requires extra memoryspace, so maynotbe goodfor asystem havingsmall RAM.

• Suffersfrominternalfragmentation.

• Thereis an increase intimetaken tofetch theinstructionsincenowtwomemory accesses are required.

SEGMENTATION

It is another schemeof memorymanagementandit generallysupports theuser viewof memory.

The Logicaladdressspace isbasicallythe collectionof segments.Each segment hasa name and a length.

Basically,aprocessisdividedintosegments.Likepaging,segmentationdividesor segments the memory.

But thereisa difference and that iswhile the **paging**dividesthememoryinto a **fixed size** and on the other hand,

segmentationdividesthe **memoryintovariablesegments**thesearethenloadedinto logical memory space.

A program isa collection of segmentsA segment isa logicalunit such as: main

program

Procedure

functionmethod

object

localvariables,

globalvariables

common block

stack

symboltable

arrays
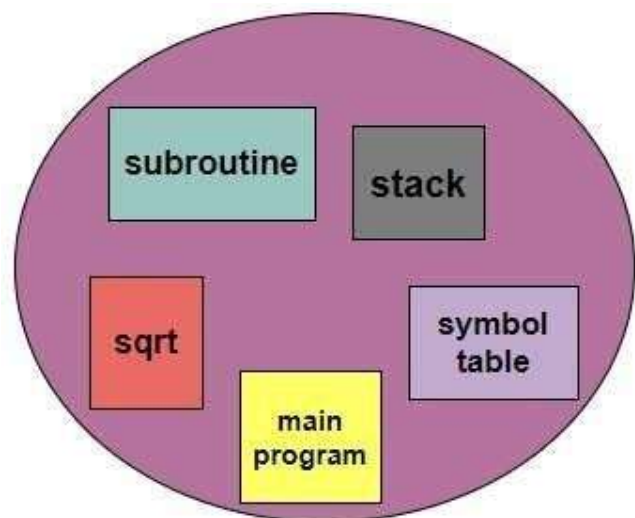
<u>Segment Table-</u>

- Segment table isa table thatstores the informationabouteachsegmentoftheprocess.
- Ithastwocolumns.
- First columnstoresthesizeor lengthofthesegment.
- Second column stores the base address or starting address of the segment in themain memory.
- Segment table isstored asa separate segment in themainmemory.
- Segmenttablebaseregister(STBR)storesthebaseaddressofthesegmenttable.

Fortheaboveillustration,considerthesegmenttableis-

	Limit	Base
Seg-0	1500	1500
Seg-1	500	6300
Seg-2	400	4300
Seg-3	1100	3200
Seg-4	1200	4700

Segment Table

Here,

- Limitindicatesthelength orsizeofthesegment.
- Baseindicatesthe baseaddress orstartingaddress of the segmentin themainmemory.

In accordance to the above segment table,the segmentsare storedin themainmemory as-
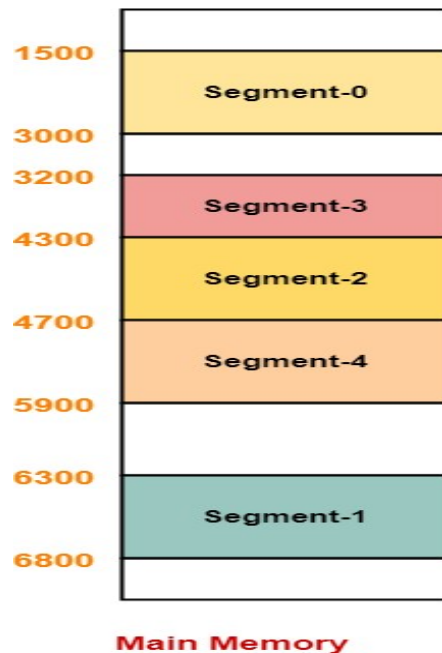
Main Memory

Translating Logical Address into Physical Address-

- CPU always generates a logical address.
- A physical address is needed to access the main memory.

Following steps are followed to translate logical address into physical address-

Step-01:

CPU generates a logical address consisting of two parts-

1. Segment Number
2. Segment Offset

Logical Address

Segment Number   **Segment Offset**
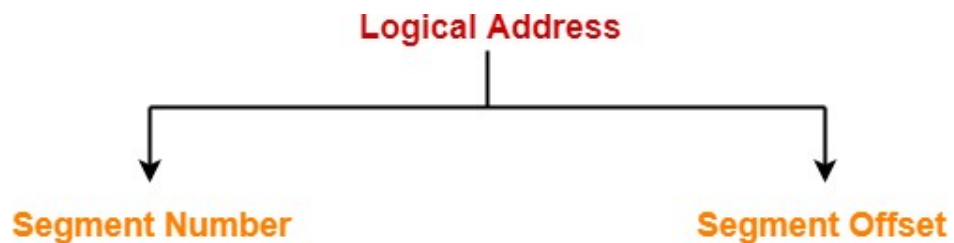
- SegmentNumberspecifiesthespecificsegmentof theprocessfromwhich CPUwants to read the data.
- SegmentOffsetspecifiesthespecificwordinthesegmentthat CPUwantstoread.

Step-02:

- Forthegenerated segmentnumber,correspondingentryislocatedinthe segmenttable.
- Then,segmentoffsetiscomparedwiththelimit(size)ofthesegment.

Now, two casesarepossible-

Case-01:SegmentOffset>=Limit

- If segmentoffsetisfoundtobe greater thanorequal tothe limit,atrap isgenerated.

Case-02:SegmentOffset<Limit

- Ifsegmentoffset isfound to be smallerthan the limit,then request istreated asavalid request.
- Thesegmentoffsetmustalwayslieintherange[0,limit-1],
- Then, segmentoffset isaddedwith the baseaddressofthesegment.
- Theresultobtained afteraddition istheaddressof thememorylocation storingthe required word.

Diagram-

Thefollowingdiagramillustratestheabovestepsoftranslatinglogicaladdressinto physical address-
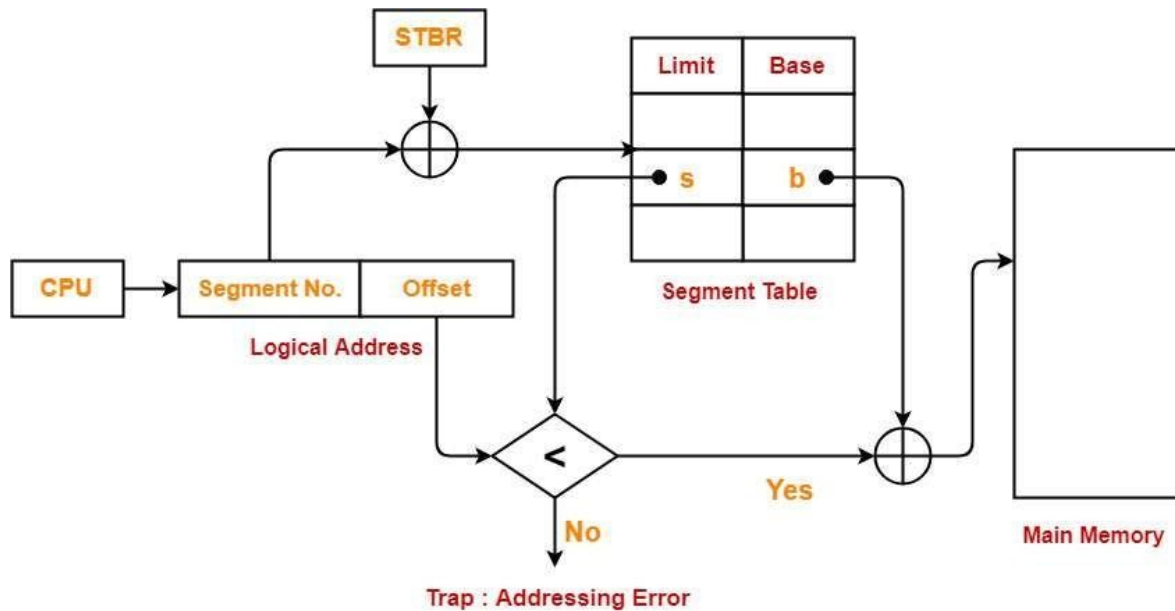
Translating Logical Address into Physical Address

Advantages-

Theadvantagesofsegmentationare-

- Itallowstodivide theprogramintomoduleswhichprovidesbettervisualization.
- Segmenttable consumeslessspaceascompared to**PageTable**inpaging.
- Itsolvestheproblemofinternalfragmentation.

Disadvantages-

Thedisadvantagesofsegmentationare-

- Thereis an overheadofmaintainingasegment tableforeachprocess.
- Thetime taken tofetchthe instruction increases sincenowtwo memoryaccessesare required.
- Segments of unequalsize arenot suitedforswapping.
- It suffersfrom externalfragmentationas thefree space getsbroken down intosmaller pieces with the processes being loaded and removed from the mainmemory.

PAGINGVSSEGMENTATION

Sr No.	Paging	Segmentation

1	Non-Contiguousmemoryallocation	Non-contiguousmemoryallocation
2	Pagingdividesprogramintofixedsize pages.	Segmentationdividesprogramintovariable size segments.
3	OSisresponsible	Compilerisresponsible.
4	Pagingisfasterthan segmentation	Segmentationisslowerthanpaging
5	PagingisclosertoOperatingSystem	SegmentationisclosertoUser
6	Itsuffersfrominternalfragmentation	It suffersfromexternal fragmentation
7	Thereis no externalfragmentation	Thereis no externalfragmentation
8	Logicaladdressisdividedintopage number and page offset	Logical addressis divided into segment number and segment offset
9	Pagetableisusedtomaintainthe page information.	Segment Table maintains the segment information
10	Pagetableentryhastheframenumber and some flag bits to represent details about pages.	Segment table entry has the base address of the segment and some protection bits for the segments.

VirtualMemoryusingpaging

Virtual Memoryis a storage mechanism which offers user an illusion of havinga verybig mainmemory. It is donebytreatinga part of secondary memory as the main memory.

- In Virtualmemory,theusercanstoreprocesseswitha biggersize than the available mainmemory.
- Insteadof loadingonelongprocessinthe main memory, theOS loads the various parts ofmore than one process in the main memory.
- Virtualmemoryisneededwheneveryourcomputerdoesn'thave space in the physicalmemory.
- Virtual memoryis mostlyimplemented with demand paging anddemand segmentation.
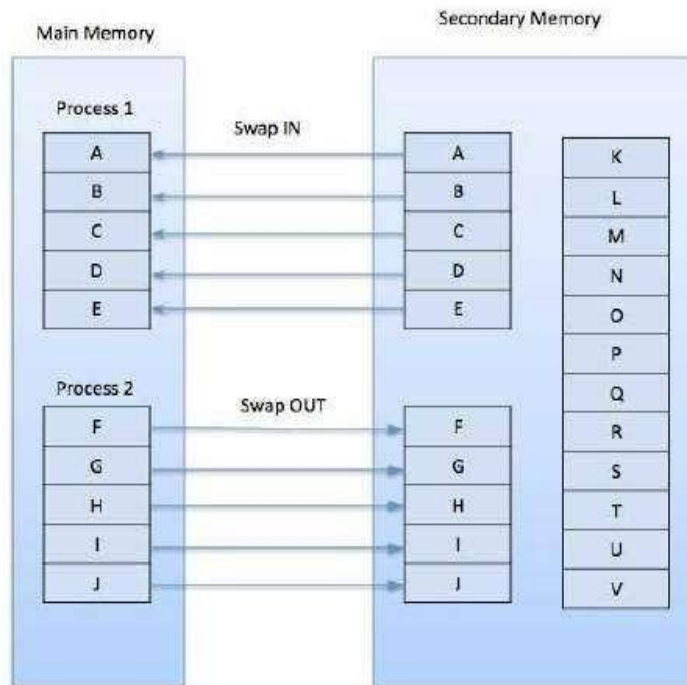
## AdvantagesofVirtual Memory

1. ThedegreeofMultiprogrammingwillbeincreased.
2. UsercanrunlargeapplicationwithlessrealRAM.
3. There isnoneed tobuymorememoryRAMs.

DisadvantagesofVirtual Memory

1. Thesystembecomesslowersinceswappingtakestime.
2. Ittakesmoretimeinswitchingbetweenapplications.
3. Theuserwillhave the lesserhard diskspaceforitsuse.
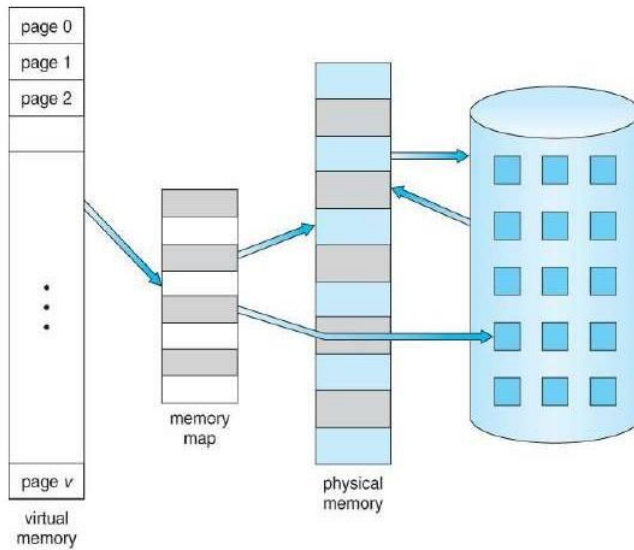
DEMANDPAGING

A demandpagingsystem is similar tothepagingsystem with swappingfeature.When we want to execute a process we swap it into the memory.

A swappermanipulatesentireprocesswhereasapagerisconcerned with theindividual pages of a process. The demand paging concept is using pager rather than swapper. Whenaprocessis tobe swapped in, thepager guesses which pages will be used before the process is swapped out again. Instead of swapping in a whole process, the pager brings only those necessary pages into memory. The transfer of a paged memory to contiguous disk space

isshown inbelowfigure.
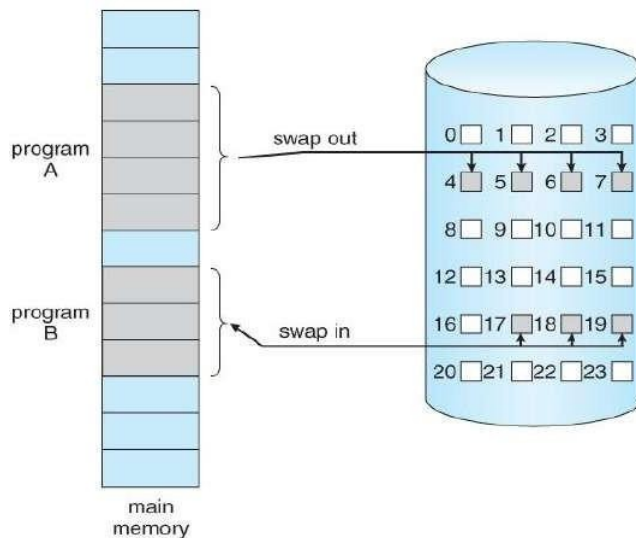
Thus it avoids readingintomemorypages that will not usedanyway decreasingthe swap time and the amount of physical memory needed.

In thistechnique we need somehardwaresupport todistinctbetween the pagesthat arein memory and those that are on the disk.

A valid and invalid bit is used for this purpose. When this bit is set to valid it indicates that theassociate page isin memory. If thebit isset toinvalid it indicatesthatthe page iseither not valid or is valid but currently not in the disk.

logical memory

page table

physical memory
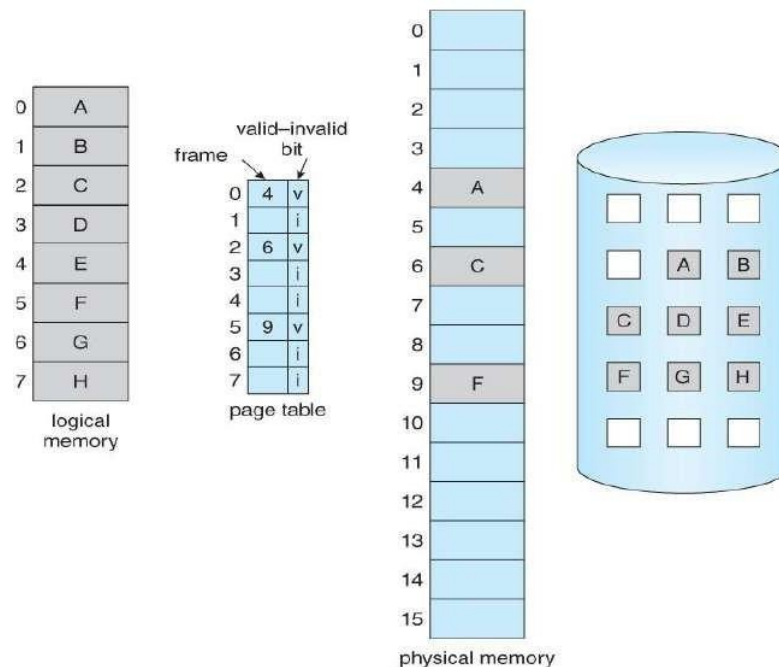
Marking a page invalid will have no effect if the process never attempts to access that page. So while a processexecutesandaccesspagesthatarememoryresident, execution proceeds normally.

Access toa pagemarked invalid causes apage fault trap. It is the result of theOS'sfailure to bring the desired page into memory.

ROCEDURETOHANDLEPAGEFAULT

- The computer hardware traps to the kernel and program counter (PC) is saved on the stack. Current instruction state information is saved in CPUregisters.
- An assembly program is started to save the general registers and other volatile information to keep the OS from destroying it.
- Operating system finds that a page fault has occurred and tries to find out which virtual page is needed. Sometimes, hardware register contains this required information.Ifnot,theoperatingsystemmustretrievePC,fetchinstructionandfind out what it was doing when the fault occurred.
- Once virtual address caused page fault is known, system checks to see if address is valid and checks if there is no protection access problem.
- If the virtualaddressisvalid, the system checks toseeif apageframeisfree.If no frames are free, the page replacement algorithm is run to remove apage.
- Ifframeselectedisdirty,pageisscheduledfortransfertodisk,contextswitch

takesplace,fault process is suspendedandanother process ismade to rununtil disk transfer is completed.

- Assoon aspageframe isclean, operatingsystem looksupdiskaddress where needed page is, schedules disk operation to bring it in.
- Whendiskinterruptindicatespagehasarrived,pagetablesareupdated to reflect its position, and frame marked as being in normalstate.
- Faulting instruction is backed up to state it had when it began and PC is reset.Faultingisscheduled,operatingsystemreturnstoroutinethatcalledit.

AssemblyRoutinereloadsregisterandotherstateinformation,returnstouserspacetocontinue execution.
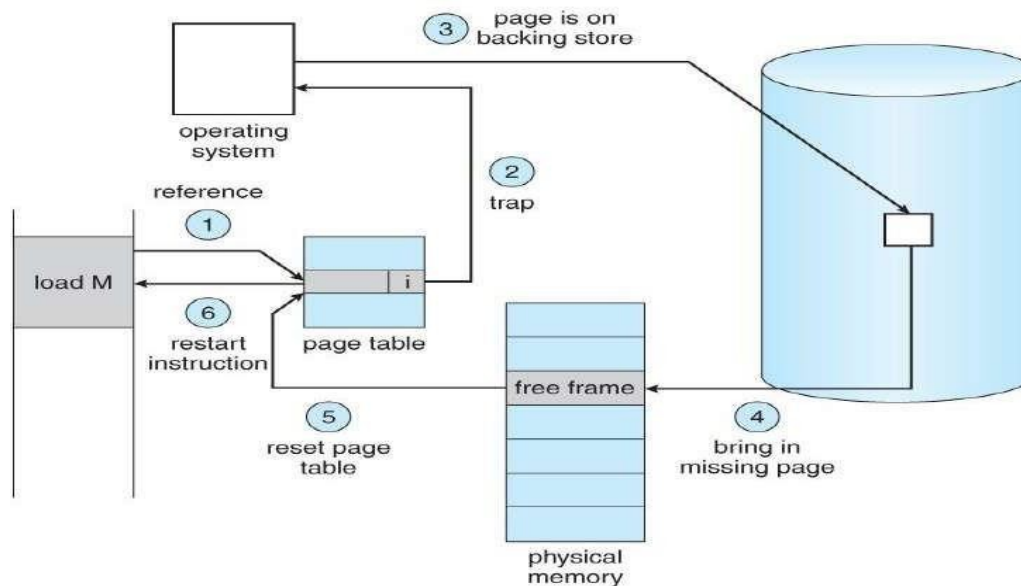
TechniquesforDeviceManagement
- **Dedicated,**
- **sharedand**
- **virtual.**

Deviceallocationconsiderationsl/Otrafficcontrol&l/OSchedule,l/O Devicehandlers.
SPOOLING.

Device management in an operating system means controlling the Input/Output devices like disk, microphone, keyboard, printer, magnetic tape, USB ports, camcorder, scanner, other accessories,andsupporting unitslikesupportingunitscontrol channels.

The system has multiple devices, and in order to handle these physical or virtual devices, the operating system requires a separate program known as an ad device controller.Italsodetermineswhethertherequesteddeviceisavailable.

Thefundamentalsofl/Odevicesmaybedividedintothree categories:

1. **BootDevice**
2. **CharacterDevice**
3. **NetworkDevice**

- Boot Device

Itstoresdatainfixed-sizeblocks,eachwithitsuniqueaddress.Forexample-Disks.

- CharacterDevice

Ittransmitsoracceptsastreamofcharacters,noneofwhichcanbeaddressed individually. For instance, keyboards, printers, etc.

- NetworkDevice

Itisusedfortransmittingthedatapackets.

Functions of the device management in the operating system

The operating system (OS) handles communication with the devices via their drivers. TheOScomponentgivesauniforminterfaceforaccessingdeviceswithvariousphysical

features. There are various functions of device management in the operating system. Someofthemareasfollows:

1. Itkeepstrackofdata,status,location,uses,etc.Thefilesystemisatermusedto defineagroupoffacilities.

2. It enforces the pre-determined policies and decides which process receives the devicewhenandforhowlong.

3. Itimprovestheperformanceofspecificdevices.

4. It monitors the status of every device, including printers, storage drivers, and other devices.

5. It allocatesand effectively deallocates the device.De-allocatingdifferentiatesthe devices at two levels: first, when an I/O command is issued and temporarily freed.Second,whenthejobiscompleted,andthedeviceispermanentlyrelease

Types ofdevices

There are three types of Operating system peripheral devices: dedicated, shared, and virtual.Theseareasfollows:

1. DedicatedDevice

In device management, some devices are allocated or assigned to only one task ata time until that job releases them. Devices such as plotters, printers, tape drivers, and other similar devicesnecessitate such an allocation mechanism because it will be inconvenient if multiple people share them simultaneously. The disadvantage of such devicesistheinefficiencycausedby allocatingthe devicetoa single userforthewhole durationoftaskexecution,evenifthedeviceisnotused100%ofthetime.

2. SharedDevices

These devices could be assigned to a variety of processes. By interleaving their requests, disk-DASD could be shared by multiple processes simultaneously. The Device Manager carefully controls the interleaving, and pre-determined policies must resolve all difficulties.

3. VirtualDevices

Virtualdevicesareahybridofthetwodevices,andtheyarededicateddevicesthat have been transformed into shared devices. Forexample, a printer can be transformed into a shareable deviceby usinga spoolingprogram that redirectsall print requests to adisk.Aprintjobisnot sentdirectlytotheprinter;however,itisroutedtothedisk untilitisfullyprepared withall oftherequiredsequencesandformatting, atwhich point it is transmitted to the printers. The approach can transform a single printer into numerousvirtualprinters,improvingperformanceandeaseofuse.

I/OschedulinginOperatingSystems

Before understanding the I/O scheduling, It's important to get an overview of I/O operations.

- Operating System hasa certain portion of code that is dedicated to managing Input/Output in order to improve the reliability and the performance of the system.
- AcomputersystemcontainsCPUsand morethanonedevicecontrollers connectedtoacommonbuschannel,generallyreferredtoasthedevicedriver.
- These device drivers provide an interface to I/O devices for communicating with the system hardware promoting ease of communication providing access to **shared** memory.
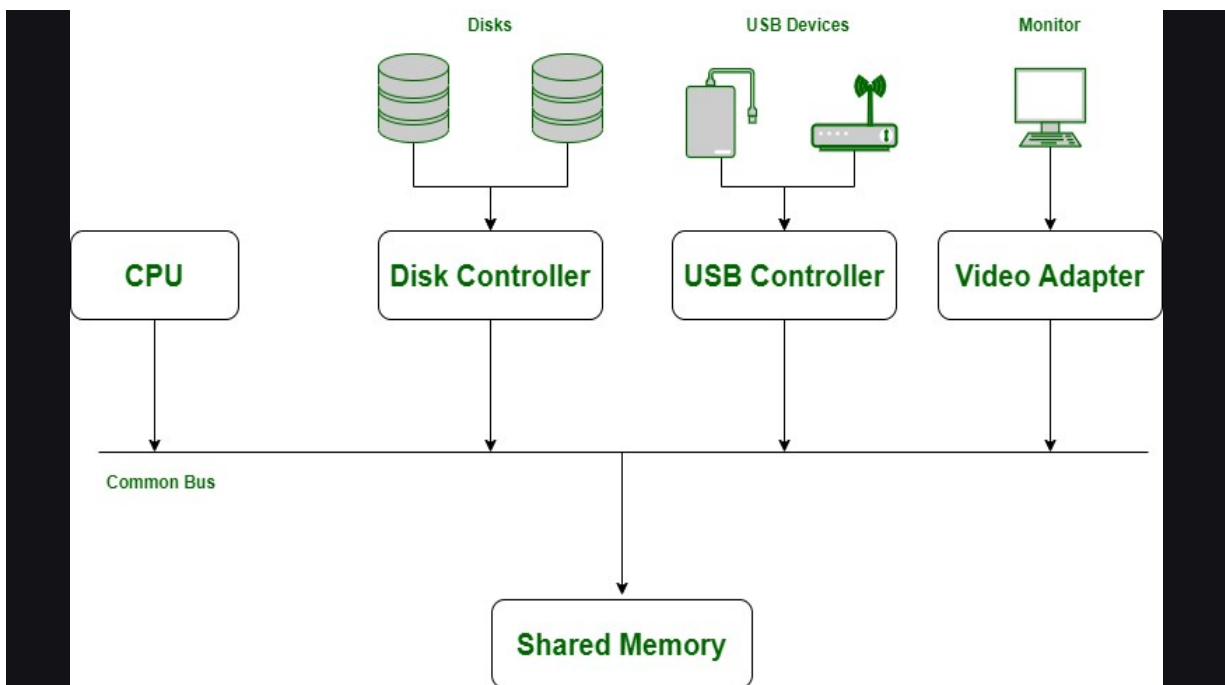
I/ORequestsinoperatingsystems:

I/O Requests are managed by Device Drivers in collaboration with some system programsinsidetheI/Odevice.TherequestsareservedbyOSusingthreesimple segments :

- **I/O TRAFFIC CONTROLLER :**Keeps track of thestatus of all devices,control units, and communication channels.
- **I/O SCHEDULER :**Executes the policiesusedbyOS toallocateandaccess the device, control units, and communication channels.
- **I/O DEVICE HANDLER :**Serves thedevice interrupts andheads thetransferof data.

I/OSCHEDULINGINOPERATINGSYSTEMS:

Schedulingisusedforefficientusageofcomputerresourcesavoidingdeadlockand serving all processes waiting in the queue.

I/OTrafficControllerhas3maintasks:

Theprimarytaskistocheckifthere'satleast onepathavailable.

Ifthereexistsmorethanonepath,itmustdecidewhichonetoselect.

Ifallpathsareoccupied,itstaskistoanalyzewhichpathwillbeavailableatthe earliest.

I/O Scheduler functions similar to Process scheduler, it allocates the devices, control units, and communication channels. However, under heavy load of I/O requests, Scheduler must decide what request should be served first and for that we multiple queues to be managed by OS.

The major difference between **Process scheduler**< and **I/O scheduler** is that I/O requestsarenotpreempted:Oncethechannelprogramhasstarted,it'sallowedto continue to completion. Although it is feasible because programs are relatively short (50 to 100 ms). Some modern OS allows I/O Scheduler to serve and higher priority requests. In simpler words, If an I/O request has higher priority then they are served before other I/O requests with lower priority. I/O scheduler works in coordination

with the I/O traffic controller to keep track of which path is being served for the current I/O request.

I/O Device Handler manages the I/O interrupts (if any) and scheduling algorithms. Few I/O handling algorithms are :

- FCFS[Firstcomefirstserver].
- SSTF[Shortestseektime first].
- SCAN
- Look
- N-StepScan
- C-SCAN
- C-LOOK

Every scheduling algorithm aims to minimize arm movement, mean response time, varianceinresponsetime.An overviewofallI/Oschedulingalgorithmsaredescribed below :

FirstComeFirstServe[FCFS]

It is one of the simplest device-scheduling algorithmsince it is easy to program and essentiallyfairtousers(I/Odevices).Theonlybarriercouldbethehighseektime,so any other algorithm that can surpass the minimum seek timeis suitable for scheduling.

ShortestSeekTimeFirst[SSTF]:

It uses the same ideology as the Shortest Job Firstin process scheduling, where shortestprocessesareservedfirstandlongerprocesseshavetowaitfortheirturn. Comparing the SJF concept in I/O scheduling, the request with track closest to one being served (The one with the shortest distance to travel on disk) is next to be satisfied. The main advantage over FCFS is that it minimizes overall seek time. It favors easy-to-reach requests and postpone traveling to those that are out of the way.

SCANAlgorithm:

SCAn uses a status flag which tells the direction of the arm, it tells whether the arm is moving towards the center of the disk or to the other side. This algorithm moves arm from the end of the disk to the center track servicing every request in its way. Whenitreachesinnermosttrack,itreversesthedirectionandmovestowardsouter tracks on the disk, again servicing every request in its path.

LOOK[ElevatorAlgorithm]:
It'savariationofSCANalgorithm,herearmdoesn'tnecessarilygoallthewayto either side on disk unless there are requests pending. It looks ahead for a request before servicing it. A big question arises is "Why should we use LOOK over SCAN?". The major advantage of LOOK over SCAN is that it discards the indefinite delay ofI/O requests.

OthervariationsofSCAN:

N-Step Scan: It holds all the pending requests until the arm starts its way back. New requests are grouped for the next cycle ofrotation.

C-SCAN [Circular SCAN] : It provides a uniform wait time as the arm serves requests on its way during the inward cycle. To know more, refer Differencebetween SCANand C-SCAN.

C-LOOK [Optimized version of C-SCAN] : Arm doesn't necessarily return to the lowest-numberedtrack,it returns from thelowestrequestto beserved. Itoptimized the C-SCAN as the arm doesn't move to the end of the disk if not required. To know more, refer to the Difference between C-LOOK and C-SCAN.

What isSpooling

Spooling is a process in which data is temporarily held to be used andexecutedby a device, program, or system. Data is sent to and stored in memory or other volatile storageuntiltheprogramorcomputerrequestsitforexecution.

SPOOL is an acronym for *simultaneous peripheral operations online*. **Generally, the spool is maintained on the computer's physical memory, buffers, or the I/O device-specificinterrupts. The spool isprocessedinascendingorder, working basedon aFIFO (first-in,first-out)algorithm.**

Spoolingreferstoputtingdataofvariousl/Ojobsinabuffer.Thisbufferisaspecial area in memory or hard disk which is accessible to I/O devices. An operating system does the following activities related to the distributed environment:

- o HandlesI/Odevicedataspoolingasdeviceshavedifferentdataaccessrates.
- o Maintains the spooling buffer, which provides a waiting station where data canrest while the slower device catches up.

- Maintains parallel computation because of the spooling process as a computer can perform I/O in parallel order. It becomes possible to have the computer read datafromatape,writedatatodisk,and writeouttoatapeprinterwhileitis doingitscomputingtask.

HowSpoolingWorksinOperatingSystem

Inanoperatingsystem,spoolingworksinthefollowingsteps,suchas:

1. Spooling involves creating a buffer called SPOOL, whichis used to hold off jobs and data till the device in which the SPOOL is created is ready to make use and execute that job or operate on the data.

2. Whenafaster devicesendsdatato aslowerdeviceto perform some operation,it uses any secondary memory attached asa SPOOLbuffer. This data iskept inthe SPOOL until the slowerdevice is ready to operate on this data. When the slower device is ready, then the data in the SPOOL is loadedonto the main memory for the required operations.
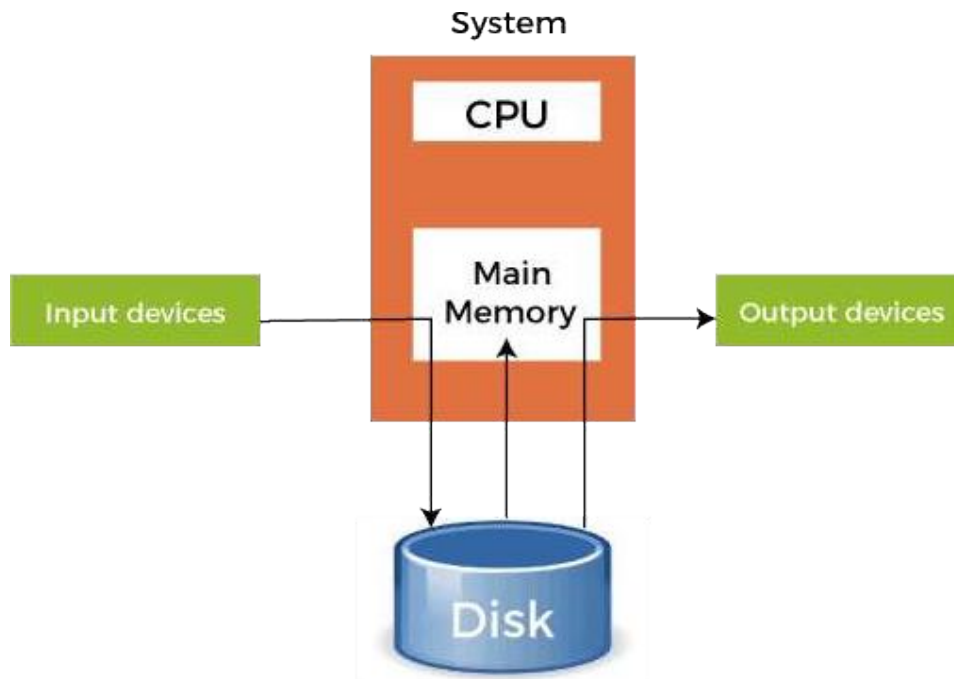
3. Spooling considers the entire secondary memory asa huge buffer that canstore manyjobsanddatafor manyoperations. The advantage ofSpoolingisthatit can create a queue of jobs that execute in FIFO order to execute the jobs one by one.

4. **A device can connect to many input devices, which may require some operation on their data. So, all of these input devices may put their data onto the secondary memory (SPOOL), which can then be executed one by one by the device. This will make sure that the CPU is not idle at any time.So, we can say thatSpoolingisacombination ofbuffering andqueuing.**

5. **After the CPU generates some output, this output is first saved in the main memory. This output is transferred to the secondary memory from the main memory,andfromthere,theoutputissenttotherespectiveoutputdevices.**

Exampleof Spooling

ThebiggestexampleofSpoolingis *printing*.Thedocumentswhicharetobeprinted are storedin the SPOOL and then added tothe queue for printing.During thistime, many processes can perform their operations and use the CPU without waiting while theprinterexecutestheprintingprocessonthedocumentsone-by-one.

Spooler/Memory
(Print Jobs in Queue)

Many features can also be addedtothe Spoolingprintingprocess,like setting priorities or notification when the printing process has been completed or selecting the different typesofpapertoprintonaccordingtotheuser'schoice.

AdvantagesofSpooling

Herearethefollowingadvantagesofspoolinginanoperatingsystem,such as:

o **The number of I/O devices oroperations doesnotmatter. Many I/O devices can work together simultaneously without any interference or disruption to each other.**

o **In spooling, there is no interaction between the I/O devices and the CPU. That means there is no need for the CPU to waitfor the I/O operations to take place.**

Such operations take along time to finishexecuting, sothe CPU will notwait for them to finish.

o   CPU in the idle state is notconsidered very efficient. Most protocols are created toutilizethe CPU efficiently inthe minimumamountof time. In spooling, the CPU is kept busy most of the time and only goes to the idle state when the queue is exhausted. So,all the tasks are added to the queue, and the CPU will finishallthosetasksandthengointotheidlestate.

o   It allows applications to run at the speed of the CPU while operating the I/O devices at their respective full speeds.

DisadvantagesofSpooling

Inanoperatingsystem,spoolinghasthefollowingdisadvantages,suchas:

o   Spooling requiresa large amount of storage depending on thenumberof requestsmadebytheinputandthenumberofinputdevicesconnected.

o   Because the SPOOL is created in the secondary storage, having many input devicesworkingsimultaneouslymaytakeup   alotofspaceonthesecondary storage and thus increase disk traffic. This results in the disk getting slowerand slower asthe traffic increasesmore and more.

o   Spooling is used for copying and executingdata from a slower device to a faster device.TheslowerdevicecreatesaSPOOLtostorethe datatobe operatedupon in a queue,and the CPU workson it. This process in itself makes Spooling futile to use in real-time environments where we need real-time results from the CPU. This is because theinputdevice is slowerand thus produces its data at a slower pace while the CPU can operate faster, so it movesonto the next process in the queue. This is whythe final result or output is produced at alatertimeinstead of in real-time.

DifferencebetweenSpoolingandBuffering

Spooling and buffering arethetwo ways bywhich I/O subsystemsimprove the performance and efficiency of the computer by using a storage space in the main memoryoronthedisk.

Spooling

Overlap the i/o operation of one job with the execution of the another job

Buffering

Overlap the i/o operation of one job with the execution of the same job
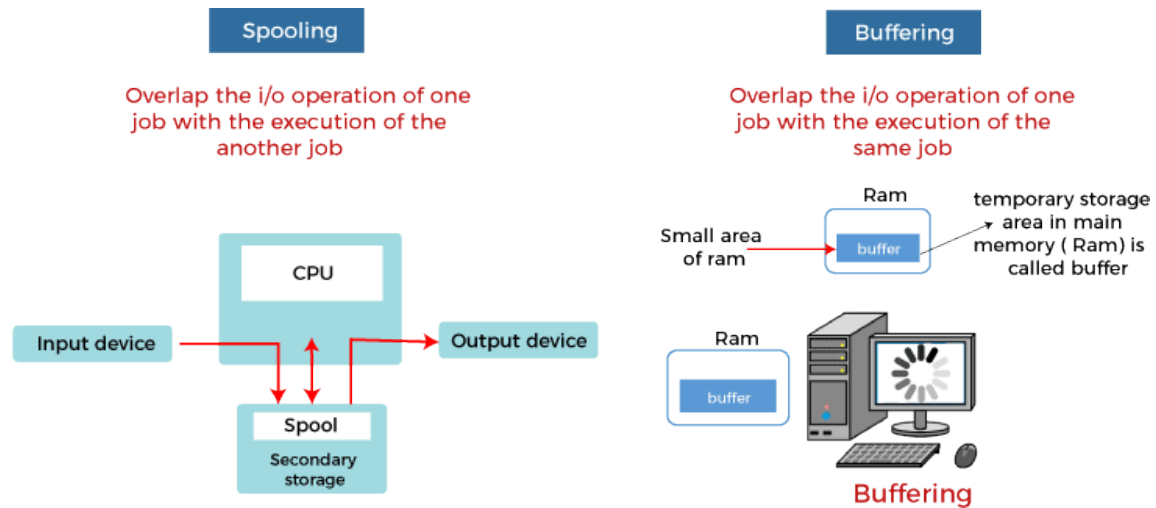
The basic difference between Spooling and Buffering is thatSpooling overlaps the I/Oof one job with theexecution of another job. In comparison, the buffering overlaps the I/Oofonejobwiththeexecutionofthesamejob.Belowaresomemoredifferences between Spooling and Buffering, such as:

Terms	Spooling	Buffering
Definition	Spooling, an acronymof Simultaneous Peripheral Operation Online (SPOOL), puts data into a temporary working area to be accessed and processed by another program or resource.	Bufferingisanactof storing data temporarily in the buffer. It helps in matchingthespeedof the data stream betweenthesender andreceiver.
Resource requirement	Spooling requires less resource management asdifferent resources manage the process for specific jobs.	Bufferingrequires more resource managementasthe same resource managesthe process ofthesamedivided job.

Internal implementation	Spooling overlaps the input and output of one job with the computation of another job.	Bufferingoverlapsthe inputandoutputof one job with the computationofthe samejob.
Efficient	Spoolingismoreefficientthan buffering.	Buffering is less efficientthanspooling.
Processor	Spooling can also process data at remotesites.Thespooler onlyhas tonotifywhenaprocessgets completed at the remote site to spoolthenextprocesstothe remote sidedevice.	Buffering does not support remote processing.
Sizeonmemory	Itconsidersthediskasahuge spoolorbuffer.	Buffer is a limited area inthemainmemory.

DEADLOCKS

UNIT-5

5. DEADLOCKS

   Conceptofdeadlock.

SystemModel

DeadLockDetection

ResourcesallocationGraph

MethodsofDeadlockhandling

Recovery&Prevention,ExplainBankersAlgorithm&SafetyAlgorithm

Conceptofdeadlock.

Deadlockhappensin operatingsystemwhentwoormoreprocessesneedsome resource to complete their execution that is held by the other process.
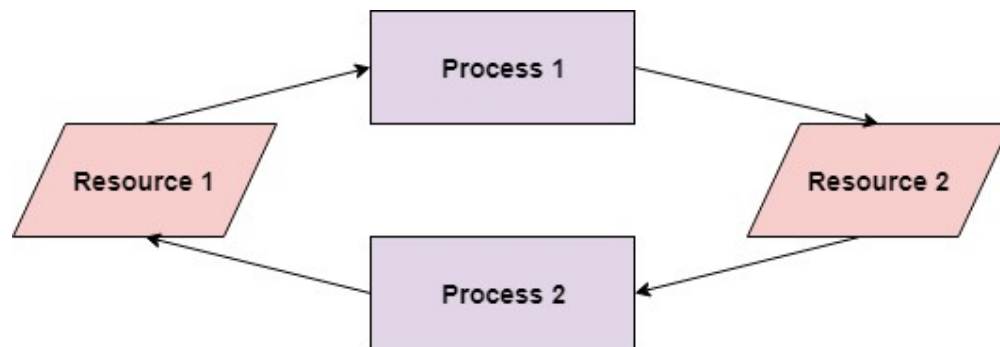
Deadlock in Operating System

In the above diagram, the process 1 has resource 1 and needs to acquire resource 2. Similarly process 2 has resource 2 and needs to acquire resource 1. Process 1 and process2areindeadlockaseachofthemneedstheother'sresourcetocomplete their execution but neither of them is willing to relinquish theirresources.

Coffman Conditions

A deadlock occurs if the four Coffman conditions hold true. But these conditions are not mutually exclusive.

TheCoffmanconditionsaregivenasfollows−

▢ **MutualExclusion**

There should be a resource that can only be held by one process at a time. In the diagram below, there is a single instance of Resource 1 and it is held by Process 1 only.
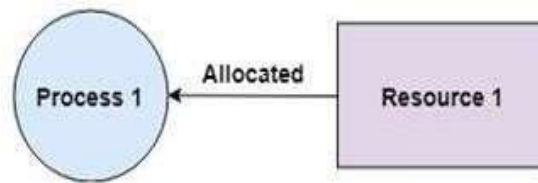
▢ Hold and Wait

Aprocesscanholdmultipleresourcesandstillrequestmoreresourcesfromother processes which are holding them. In the diagram given below, Process 2 holds Resource 2 and Resource 3 and is requesting the Resource 1 which is held by Process 1.
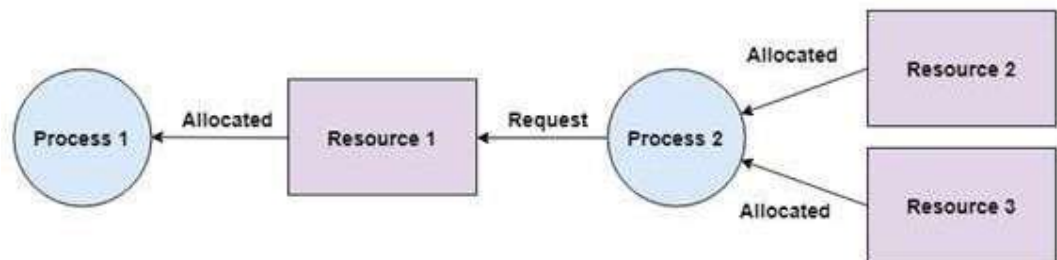

▢ No Preemption

A resource cannot be preempted from a process by force. A process can only release a resource voluntarily. In the diagram below, Process 2 cannot preempt Resource 1 from Process 1. It will only be released when Process 1 relinquishes it voluntarily after its execution is complete.
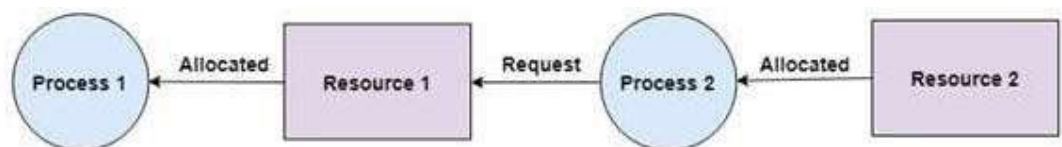
▢ Circular Wait

A process is waiting for the resource held by the second process, which is waiting fortheresourceheldby thethirdprocess andsoon,tillthelast processis waiting for a resource held by the first process. This forms a circular chain. For example: Process 1 is allocated Resource2 and it is requesting Resource 1. Similarly,

Process2isallocatedResource1anditisrequestingResource2.This formsa circular wait loop.
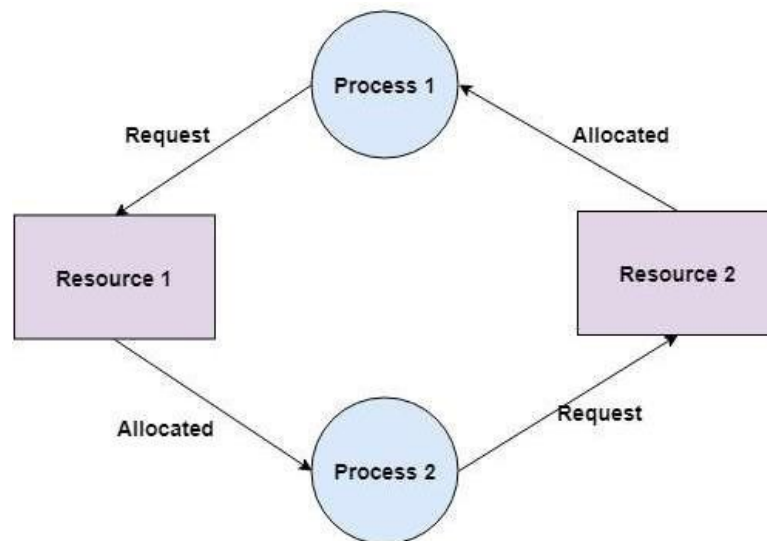
- ## **5.2SYSTEMMODEL**

- Forthepurposesofdeadlockdiscussion,asystemcanbemodeledasacollection of limited resources(r1, r2,r3,....rn) which can be partitioned into different categories, to be allocated to a number of processes(P1, P2 ,P3 ,...Pn) each having different needs.

- Resourcecategoriesmayincludememory,printers,CPUs,openfiles,tapedrives, CD-ROMS, etc.

- Bydefinition,alltheresourceswithinacategoryareequivalent,andarequestof thiscategorycanbeequallysatisfiedbyanyoneoftheresourcesinthatcategory. Ifthisisnotthecase(i.e.ifthereissomedifferencebetweentheresourceswithin a category), then that category needs to be further divided into separate categories. For example, "printers" may need to be separated into "laser printers" and "colour inkjet printers".

- Somecategoriesmayhaveasingle resource.

- In normal operation a process must request a resource before using it, and release it when it is done, in the following sequence:
  1. *Request* - If the request cannot be immediately granted, then the processmustwait until the resource(s) it needs become available.For example the system calls open( ) , malloc( ), new( ), and request().

2. ***Use***-Theprocessusesthe resource,e.g.*prints totheprinter*or *reads from thefile.*
3. ***Release*** - The process relinquishes the resource. so that it becomes available for other processes. For example, close( ), free( ), delete( ), and release( ).
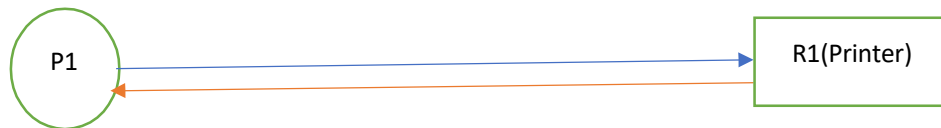
- ## **5.3DEADLOCKDETECTION**

Adeadlockcanbedetectedbyaresourceschedulerasitkeepstrackof   all the resources that are allocated to different processes. After a deadlockisdetected,itcanberesolvedusingthefollowingmethods:

1. All the processes that are involved in the deadlock are terminated. Thisisnotagoodapproachasalltheprogressmadebytheprocesses   is destroyed.
2. Resourcescanbe pre-emptedfromsome processesandgiven to others till thedeadlock is resolved.
3. **Ifresourceshavesingleinstance:**
4. InthiscaseforDeadlockdetectionwecanrunanalgorithmtocheck   for cycle in the Resource Allocation Graph. Presence of cycle in the graph is the sufficient conditionfor deadlock.
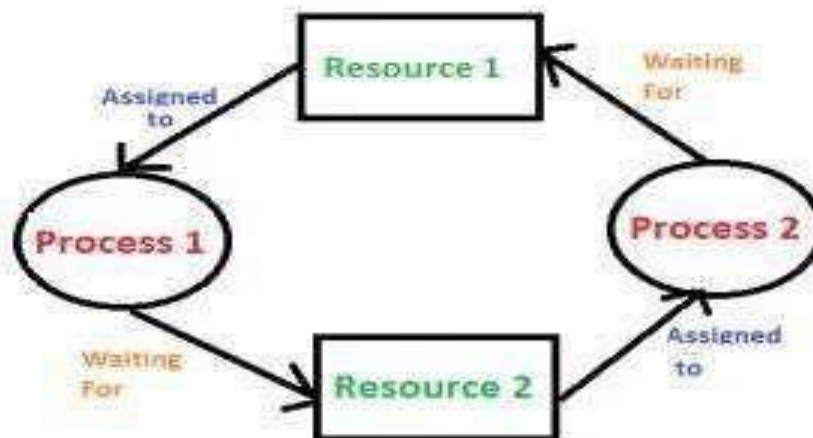
Intheabovediagram,resource1andresource2havesingleinstances.Thereis a cycle

R1→P1→ R2 →P2.So,DeadlockisConfirmed.

1. **Iftherearemultipleinstancesofresources**:
   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.

- ## **5.4RESOURCEALLOCATIONGRAPH**

Theresourceallocationgraphis the pictorial representationof thestateof a system. Asitsnamesuggests,theresourceallocationgraphisthecompleteinformationabout all the processes which are holding some resources or waiting for some resources.

It also contains the information about all the instances of all the resources whether they are available or being used by the processes.

InResourceallocationgraph,theprocessisrepresentedbyaCirclewhiletheResource is represented by a rectangle. Let's see the types of vertices and edges in detail.
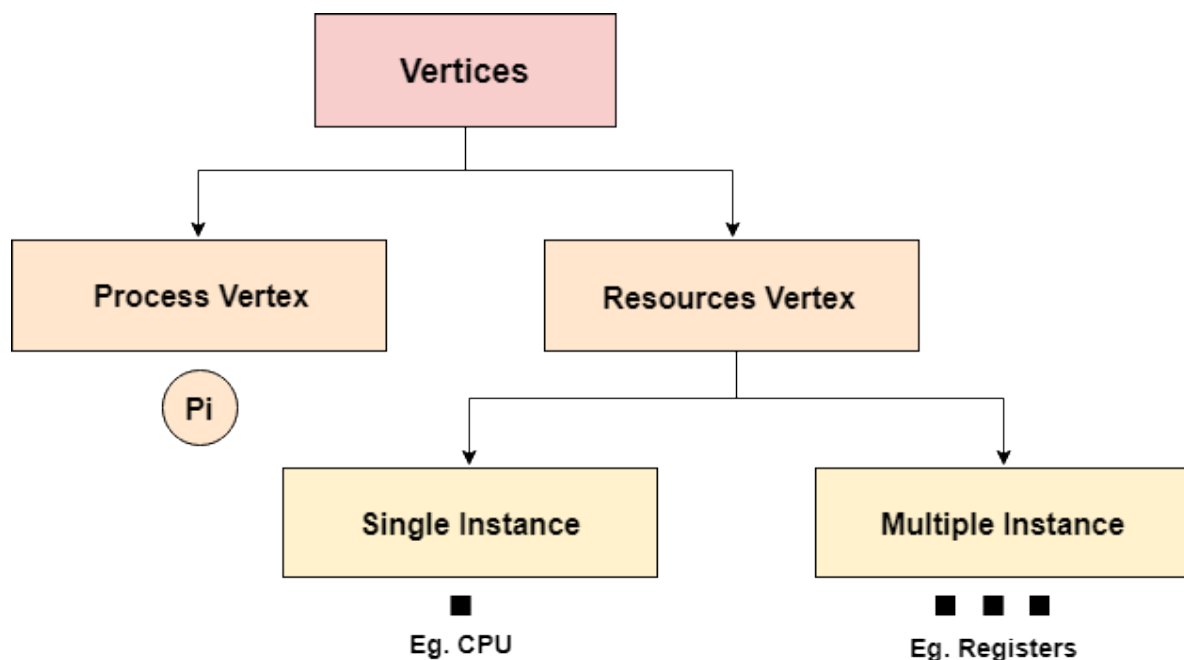
Vertices are mainly of two types, Resource and process. Each of them will be representedbyadifferentshape.Circlerepresentsprocesswhilerectanglerepresents resource.

A resource can have more than one instance. Each instance will be represented by a dot inside the rectangle.
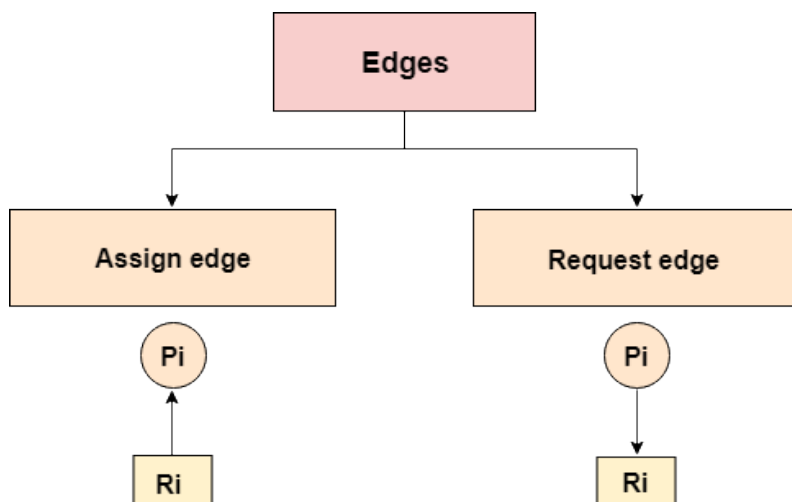
Edges inRAG are also of two types,one represents assignment andother represents the wait of a process for a resource. The above image shows each of them.

A resource is shown as assigned to a process if the tail of the arrow is attached to an instance to the resource and the head is attached to a process.

A process is shown as waiting for a resource if the tail of an arrow is attached to the process while the head is pointing towards the resource.

Process is requesting for a resource

Resource is assigned to process
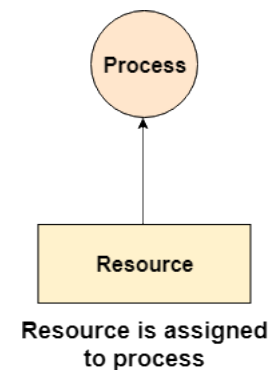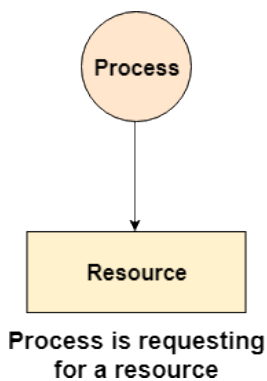
Example

Let'sconsider 3 processes P1, P2 and P3, and two types of resources R1 and R2. The resources are having 1 instance each.

According to the graph, R1 is being used by P1,P2 is holding R2 and waiting for R1, P3 iswaitingforR1aswellasR2.

Thegraphisdeadlockfreesincenocycleisbeingformedinthe graph.

R1   R2
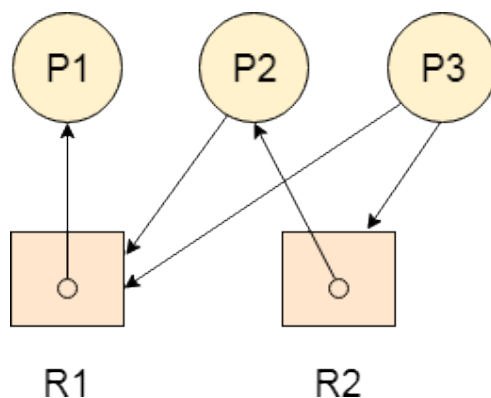
5.5METHODSFORHANDLINGDEADLOCKS

- Generally,speakingtherearethreewaysofhandlingdeadlocks:
  1. *Deadlockpreventionoravoidance*-Donotallowthesystemtogetintoa deadlocked state.
  2. *Deadlockdetectionandrecovery*-Abortaprocessorpreemptsome resourceswhen deadlocks are detected.
  3. *Ignoretheproblemalltogether* -Ifdeadlocksonlyoccuroncea yearorso, it may be better to simply let them happen and reboot as necessary than

to incur the constant overhead and system performance penalties associatedwithdeadlockpreventionor detection. This is theapproachthat both Windows and UNIX take.

- Inordertoavoiddeadlocks,thesystemmusthaveadditionalinformationabout allprocesses.Inparticular,thesystemmustknowwhatresourcesaprocesswill or may request in the future. (Ranging from a simple worst-case maximum to a complete resource request and release plan for each process, depending on the particular algorithm.)
  - Deadlock detection is fairly straightforward, but deadlock recovery requires either aborting processes or preempting resources, neither of which is an attractive alternative.
  - If deadlocks are neither prevented nor detected, then when a deadlock occurs the system will gradually slow down, as more and more processes becomestuckwaitingforresourcescurrentlyheldbythedeadlockandby other waiting processes. Unfortunately, this slowdown can be indistinguishable from a general system slowdown when a real- time process has heavy computing needs.

5.6 DEADLOCKPREVENTION

WecanpreventDeadlockbyeliminatinganyoftheabovefourconditions.

1. EliminateMutualExclusion
It is not possible to dis-satisfy the mutual exclusion because some resources, such as the tape drive and printer, are inherently non-shareable.

2. EliminateHoldandwait
Allocate all required resources to the process before the start of its execution, this wayholdandwaitconditioniseliminatedbutitwillleadtolowdeviceutilization. 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.

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

HOLD AND WAIT
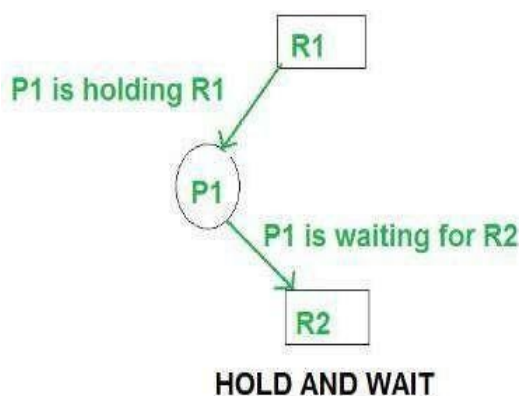
3. **EliminateNoPre-emption**

Pre-emptresourcesfromthe process when resources required by otherhigh priorityprocesses.

4. **EliminateCircularWait**

Eachresourcewillbeassignedwithanumericalnumber.Aprocesscanrequestthe resources increasing/decreasing. order of numbering.

ForExample,if P1process is allocatedR5resources,nownexttimeifP1askforR4, R3 lesser than R5 such request will not be granted, only request for resources more than R5 will be granted.

DEADLOCKAVOIDANCE

The general idea behind deadlock avoidance is to prevent deadlocks from ever happening, by preventing at least one of the aforementioned conditions.

- This requires more information about each process, and tends to lead to low device utilization. (i.e. it is a conservative approach.)
- In some algorithms the scheduler only needs to know the *maximum* number of each resource that a process might potentially use. In more complex algorithms the scheduler can also take advantage of the *schedule* of exactly what resources may be needed in what order.
- Whenaschedulerseesthatstartingaprocessorgrantingresourcerequests may lead tofuture deadlocks, then that process is just not started or the request is not granted.
- A resource allocation **state** is defined by the number of available and allocatedresources,andthemaximumrequirementsofallprocessesinthe system.

Safe State- A state is **safe** if the system can allocate all resources requested by all processes (up to their stated maximums) without entering adeadlock state. If the system cannot fulfill the request of all processes then the state of the system is called **unsafe**.

More formally, a state is safe if there exists a **safe sequence(the order of process termination)** of processes { P0, P1, P2, ..., PN } such that all of the resource requests for Pi can be granted using the resources currently allocated to Pi and all processes Pj where j < i. (i.e. if all the processes prior to Pi finish and free up

theirresources,thenPiwillbeabletofinishalso,usingtheresourcesthatthey have freed up.)

⊡ If a safe sequence does not exist, then the system is in an unsafe state, whichmay leadto deadlock. (All safestates are deadlockfree,butnot all unsafe states lead to deadlocks.)

⊡
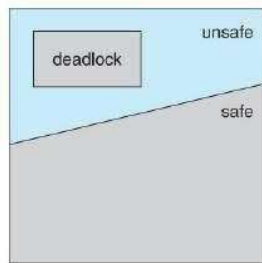
Figure-Safe,unsafe,anddeadlockedstate spaces.

NOTE: Deadlockpreventionensuresthatatleastoneofthenecessaryconditionsto cause deadlock will never occur while deadlock avoidance ensures that the system never enters anunsafe state.

BANKER'S ALGORITHMINOPERATINGSYSTEM

Banker'salgorithmisa deadlockavoidancealgorithm.**Itisnamedso becausethis** algorithm is used in banking systems to determine whether a loan can be granted or not.

Considertherearenaccountholdersinabankandthesumofthemoneyinalloftheir accountsisS.Everytimealoanhastobegrantedbythebank, itsubtractsthe**loan amount**fromthe**totalmoney**thebankhas.Thenitchecksifthatdifferenceisgreater than S. It is done because, only then, the bank would have enough money even if all the n account holders draw all their money atonce.

Banker'salgorithmworksinasimilarwayincomputers.

Whenever a new process is created, it must specify the maximum instances of each resource type that it needs, exactly.

CHARACTERISTICSOFBANKER'SALGORITHM

ThecharacteristicsofBanker'salgorithmareasfollows:

- Ifanyprocessrequestsforaresource,thenithastowait.
- This algorithm consists of advanced features for maximum resource allocation.
- Therearelimitedresourcesinthesystemwehave.
- Inthisalgorithm,ifanyprocessgets alltheneededresources,thenitisthatit should return the resources in a restricted period.
- Variousresourcesaremaintainedinthisalgorithmthatcanfulfilltheneedsof at least one client.

Letusassumethattherearenprocessesandmresourcetypes.

DATASTRUCTURESUSEDTOIMPLEMENTTHEBANKER'SALGORITHM

Somedatastructuresthatareusedtoimplementthebanker'salgorithmare:

1. Available

Itisan**array**oflengthm.Itrepresentsthenumberofavailableresourcesofeachtype. If Available[j] = k, then there are k instances available, of resource type Rj.

2. Max

It is an n x m matrix which represents the maximum number of instances of each resource that a process can request. If Max[i][j] = k, then the process Pi can request atmost k instances of resource type Rj.

3. Allocation

Itisannxmmatrixwhichrepresentsthenumberofresourcesofeachtypecurrently allocated to each process. If Allocation[i][j] = k, then process Pi is currently allocated k instances of resource type Rj.

4. Need

It is a two-dimensional array. It is an n x m matrix which indicates the remaining resource needs of each process. If Need[i][j] = k, then process Pi may need k more instances of resource type Rj to complete its task.

Need[i][j]=Max[i][j]-Allocation[i][j]

Banker'salgorithmcomprisesoftwo algorithms:

1. Safetyalgorithm
2. Resourcerequestalgorithm

SAFETYALGORITHM

Asafetyalgorithmisanalgorithmusedto findwhetherornotasystemisinitssafe state. The algorithm is as follows:

1. Let$Work$and$Finish$bevectorsoflength**m**and**n**,respectively. Initially,

2. Work=Available

3. Finish[i]=falsefori=0,1,...,n-1.

Thismeans,initially,noprocesshasfinishedandthenumberofavailable resources is represented by the **Available** array.

4.Findanindex**i**suchthatboth

5. Finish[i]==false

6. Needi<=Work

Ifthereisnosuchipresent,thenproceedtostep 4.

Itmeans,weneedtofindanunfinishedprocesswhoseneedscanbesatisfied by the available resources. If no such process exists, just go to step 4.

7.Performthefollowing:

8. Work=Work+Allocationi

9. Finish[i]=true

Gotostep2.

When an unfinished process is found, then the resources are allocated andthe process is marked finished. And then, the loop is repeated to check the same for all other processes.

10. If `Finish[i]==true` foralli,thenthesystemisinasafestate.

That means if all processes are finished, then the system is in safe state.

Thisalgorithmmayrequireanorderof **mxn²operations**inordertodetermine whether a state is safe or not.

Now the next algorithm is a resource-request algorithm and it is mainly used to determine whether requests can be safely granted or not.

Let Requesti be the request vector for the process Pi. If Requesti[j]==k, then process Pi wants k instance of Resource type Rj.Whena request for resources is made by the process Pi, the following are the actions that will be taken:

1. IfRequesti<=Needi,thengotostep2;elseraiseanerrorcondition,sincethe process has exceeded its maximum claim.

2. IfRequesti<=Availableithengotostep3;elsePimusthavetowaitasresourcesare not available.

3. NowwewillassumethatresourcesareassignedtoprocessPi and thusperform the following steps:

Available= Available-Requesti;

Allocationi=Allocationi+Requesti;

Needi =Needi - Requesti;

Iftheresultingresourceallocationstatecomesouttobesafe,thenthetransactionis completed and, process Pi is allocated its resources. But in this case, if the new state isunsafe,thenPiwaitsforRequesti,andtheoldresource-allocationstateisrestored.

Disadvantagesof Banker'sAlgorithm

Somedisadvantagesofthisalgorithmareasfollows:

1. DuringthetimeofProcessing,thisalgorithmdoesnotpermitaprocessto change its maximum need.
2. Anotherdisadvantageofthisalgorithmisthatalltheprocessesmustknow in advance about the maximum resource needs.
3. Thisalgorithmpermitstherequeststobe providedinconstrainedtime,but for one year which is a fixed period.

DEADLOCKRECOVERY

In order to recover the system from deadlocks, either OS considers resources or processes.

FOR RESOURCE

Preempttheresource

We can snatch one of the resources from the owner of the resource (process) and give it to the other process with the expectation that it will complete the execution andwillreleasethisresourcesooner.Well,choosingaresourcewhichwillbesnatched is going to be a bit difficult.

Rollbacktoasafe state

System passes through various states to get into the deadlock state. The operating system canrollback the systemto the previous safe state. For this purpose, OS needs to implement check pointing at every state.

Themoment, we get into deadlock,we will rollbackall theallocations to getinto the previous safe state.

FOR PROCESS

Kill aprocess

Killing a process can solve our problem but the bigger concern is to decide which process to kill. Generally, Operating system kills a process which has done least amount of work until now.

Killallprocess

This is not a suggestible approach but can be implemented if the problem becomes very serious. Killing all process will lead to inefficiency in the system because all the processes will execute again from starting.
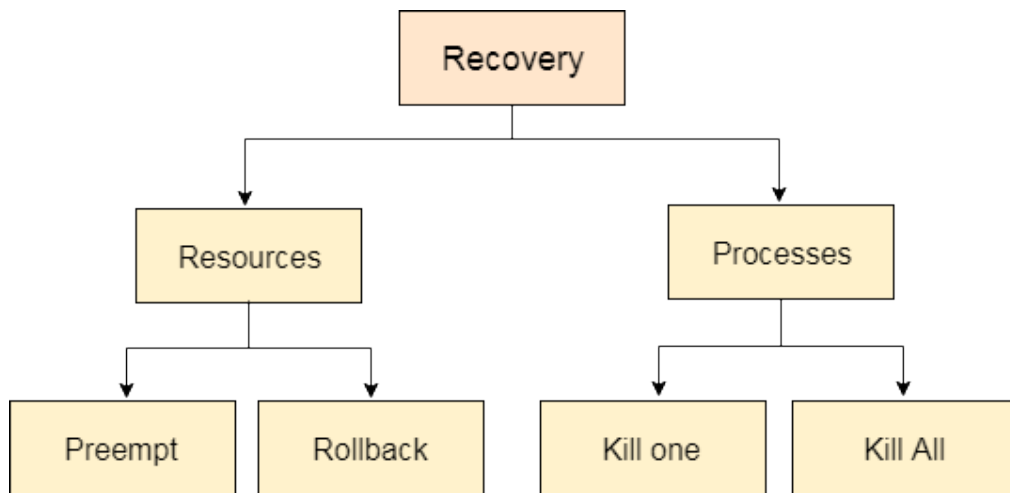
FILEMANAGEMENT
UNIT-6

6. FILEMANAGEMENT
   Fileorganization,Directory&filestructure,sharingoffiles
   Fileaccessmethods,filesystems,reliability
   Allocationofdiskspace
   Fileprotection,secondarystoragemanagement.
6.FILEMANAGEMENT

le management is one of the basic and important features of operating system. Operating system is used to manage files of computer system. All the files with different extensions are managed by operating system.

A file is collection of specific information stored in the memory of computer system. File management is defined as the process of manipulating files in computer system, it management includes the process of creating, modifying and deleting the files.

The following are some of the tasks performed by file management of operating system of any computer system:

1. Ithelpstocreatenewfilesincomputersystemandplacingthematthespecific locations.

2. Ithelpsineasilyandquicklylocatingthesefilesincomputersystem.

3. It makes the process of sharing of the files among different users very easy and user friendly.

4. It helps to stores the files in separate folders known as directories. These directorieshelpuserstosearchfilequicklyortomanagethefilesaccordingtotheir types or uses.

5. Ithelpstheusertomodifythedataoffilesortomodifythenameofthefileinthe directories.

FILEORGANIZATION,DIRECTORY&FILESTRUCTURE,SHARINGOFFILES

FILE- A file is a collection of related information that is recorded on secondary storage. From user's point of view, a file is the smallest allotment of logical data in secondary storage.

 Orfileisacollectionoflogicallyrelatedentities.

FILE ORGANIZATION refers to the way data is stored in a file. File organization is veryimportantbecauseitdeterminesthe methodsofaccess,efficiency,flexibility

and storage devices to use. Or we can say, File organization is the "logicalstructuring'' as well as the access method(s)of files.

Commonfileorganizationschemesare:

- sequential,
- random,
- serialand
- indexed-sequential

FILEDIRECTORIES:

Collectionoffilesisafiledirectory.Thedirectorycontainsinformationaboutthe files, including attributes, location and ownership. Much of this information, especially that is concerned with storage, is managed by the operating system. The directory is itself a file, accessible by various filemanagement routines.

Informationcontainedinadevicedirectoryare:

- Name
- Type
- Address
- Currentlength
- Maximumlength
- Datelastaccessed
- Datelastupdated
- Ownerid
- Protectioninformation

Operationperformedondirectoryare:

- Searchforafile
- Createafile
- Deleteafile
- Listadirectory
- Renameafile
- Traversethefilesystem

Advantagesofmaintainingdirectoriesare:

- **Efficiency:**Afilecanbelocatedmorequickly.
- **Naming:**Itbecomesconvenientforusersastwouserscanhavesamename for different filesor may have different name for samefile.

**Grouping:**Logicalgroupingoffilescanbedonebypropertiese.g.alljavaprograms, all
1. **SINGLE-LEVEL DIRECTORY**
   Inthisasingledirectoryismaintainedforalltheusers.

- **Namingproblem:**Userscannothavesamenamefor twofiles.
- **Groupingproblem:**Userscannotgroupfilesaccordingtotheirneed.

| Directory | cat | xkcd | a | dtr | ims | trs | dog | imp | sos |

2.   TWO-LEVEL DIRECTORY

In this, separate directories for each user is maintained.

- **Pathname:** Due to two levels there is a pathname for every file to locate that file.
- Now, we can have same file name for different user.
- Searching is efficient in this method.
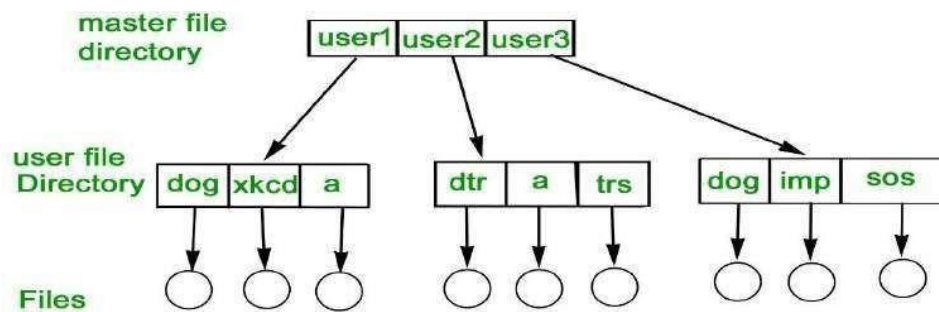
3.   TREE-STRUCTURED DIRECTORY:

Directory is maintained in the form of a tree. Searching is efficient and also there is grouping capability. We have absolute or relative path name for a file.

When operating system defines different file structures, it also contains the code to support these file structure. Unix, MS-DOS support minimum number of file structure.
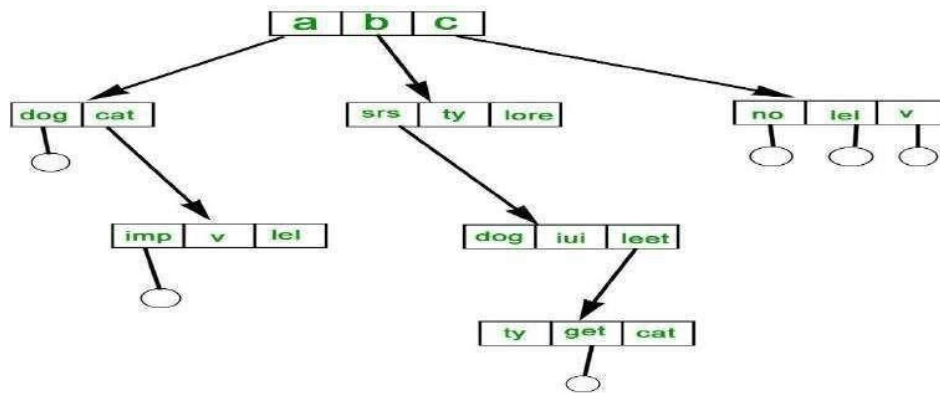
FILESTRUCTURE

AFileStructureshouldbeaccordingtoarequiredformatthattheoperating system canunderstand.

- Afilehasacertaindefinedstructureaccordingtoitstype.
- Atextfileisasequenceofcharactersorganizedintolines.
- Asourcefileisasequenceofproceduresandfunctions.
- An object file is a sequence of bytes organized into blocks that are understandable bythemachine.
- When operating system defines different file structures, it also contains the codeto supportthesefile structure. Unix,MS-DOS supportminimumnumber of file structure.

FILESHARING

Allowing users to share files raises a major issue: protection. A general approach is to provide controlledaccess to files througha set of operations such asread,write,delete,list,andappend.Thenpermituserstoperformoneormore operations.

One popular protection mechanism is a condensed version of access list, where the syst emrecognizes three classifications of users with each file and directory:

- user
- group
- other

FILEACCESSMETHODS,FILESYSTEMS,RELIABILITY

FILEACCESSMETHODS

Fileaccessmethodsreferstothemannerinwhichtherecords ofafilemaybe accessed.

There are several ways to access files −

- Sequential access
- Direct/Random access
- Indexed sequential access

SEQUENTIAL ACCESS

- A sequential access is that in which the records are accessed in some sequence, i.e., the information in the file is processed in order, one record after the other.
- This access method is the most primitive one.
- Example: **Compilers** usually access files in this fashion.

DIRECT/RANDOM ACCESS

- Records are stored randomly but accessed directly.
  Each record has its own address on the file with by the help of which it can be directly accessed for reading or writing
- The records need not be in any sequence within the file and they need not be in adjacent locations on the storage medium.
- **Magnetic** and **optical** disks allow data to be stored and accessed randomly.

INDEXED SEQUENTIAL ACCESS

- This mechanism is built upon base of sequential access.
- An index is created for each file which contains pointers to various blocks.
- Index is searched sequentially and its pointer is used to access the file directly.
- For example, on a **magnetic drum**, records are stored sequential on the tracks. However, each record is assigned an index that can be used to access it directly.

RELIABILITY

Regular file operations often involve changing several disk blocks. For example, What can happen if disk loses power or machine software crashes?

- Some operations in progress may complete
- Some operations in progress may be lost
- Overwrite of a block may only partially complete

what if the computer crashes between two of these operations? Then the file system

could enter an *inconsistent state*, confusing the OS to the point that the filesystem can't be used at all. In order to make file system reliable we need to keep Backup/restore(disasterscenarios)andhaveaconsistentcheckonFilesystem(i.e., maintain consistency) (e.g., UNIX fsck).

Filesystemwantsdurability

–Data previously stored can be retrieved (maybe after some recovery step), regardless offailure.

ALLOCATIONOFDISKSPACE

FileAllocationMethods

There are different kinds of methods that are used to allocate disk space. We must selectthebestmethodforthefileallocationbecauseitwilldirectlyaffectthesystem performance and system efficiency. With the help of the allocation method, we can utilize the disk, and also files can be accessed.

Therearevarioustypesoffileallocationsmethod:

1. Contiguousallocation
2. Extents
3. Linkedallocation
4. Clustering
5. FAT
6. Indexedallocation
7. LinkedIndexedallocation
8. MultilevelIndexedallocation
9. Inode

Therearedifferenttypesoffileallocationmethods,butwemainlyusethreetypesof   file allocation methods:

1. Contiguousallocation
2. Linkedlistallocation
3. Indexedallocation

These methods provide quick access to the file blocks and also the utilization of disk space in an efficient manner.
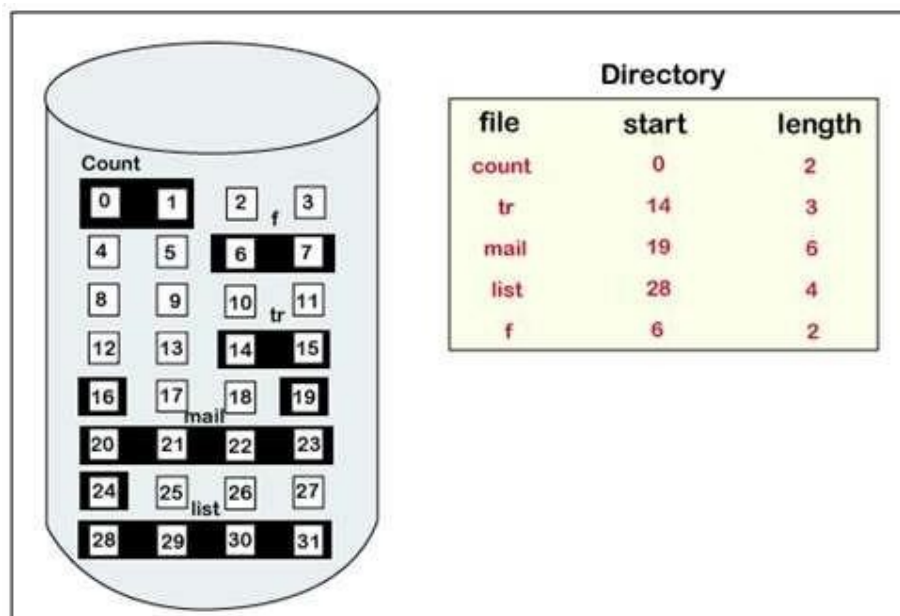
CONTIGUOUSALLOCATION:

Contiguous allocation is one of the most used methods for allocation. Contiguous allocation means we allocate the block in such a manner, so that in the hard disk, all the blocks get the contiguous physical block.

Wecan seein the below figurethatin the directory,wehavethreefiles. Inthetable, wehavementionedthestartingblockandthelengthofallthefiles.Wecanseeinthe table that for each file, we allocate a contiguous block.

Contiguous Allocation
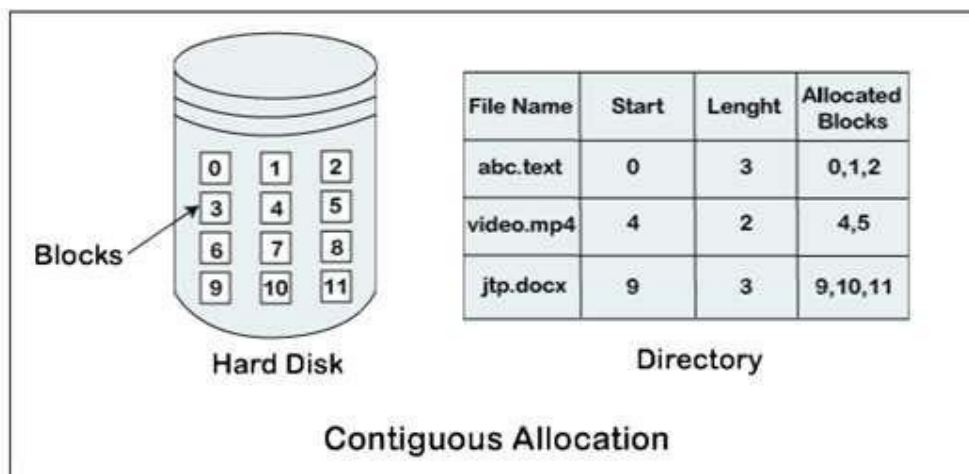
Exampleofcontiguousallocation

We can see in the given diagram,thatthereisafile.Thenameofthefileis'mail.' Thefilestartsfromthe19thblockand thelengthofthefileis6.So,thefileoccupies6 blocks in a contiguous manner. Thus, it will hold blocks 19, 20, 21, 22, 23,24.

AdvantagesofContiguousAllocation

Theadvantagesofcontiguousallocationare:

1. Thecontiguousallocationmethodgivesexcellentreadperformance.
2. Contiguousallocationiseasyto implement.
3. Thecontiguousallocationmethodsupportsbothtypesoffileaccessmethods that are sequential access and direct access.
4. TheContiguousallocationmethodisfastbecause,inthismethodnumber of seeks is less due to the contiguous allocation of file blocks.

DisadvantagesofContiguousallocation

Thedisadvantagesofcontiguousallocationmethodare:

1. Inthecontiguousallocationmethod,sometimesdiskcanbefragmented.
2. Inthismethod,itis difficulttoincreasethesizeofthefileduetothe availability of the contiguous memory block.

LINKEDLISTALLOCATION

The linked list allocation method overcomes the drawbacks of the contiguous allocationmethod.In this fileallocationmethod,eachfileis treatedas a linked list of disks blocks. In the linked list allocation method, it is not required that disk blocks assignedto a specific file are in the contiguous order on the disk. The directory entry comprises of a pointer for starting file block and also for the ending file block. Each disk blockthat is allocatedor assignedto a fileconsists of a pointer, and that pointer point the next block of the disk, which is allocated to the same file.

Exampleoflinkedlistallocation

Wecanseeinthebelowfigurethatwehaveafilenamed'jeep.'Thevalueofthe start is 9. So, we have to start the allocation from the 9th block, and blocks are allocated in a random manner. The value of the end is 25. It means the allocation is finished on the 25th block. We can see in the below figure that the block (25) comprised of -1, which means a null pointer, and it will not point to anotherblock.
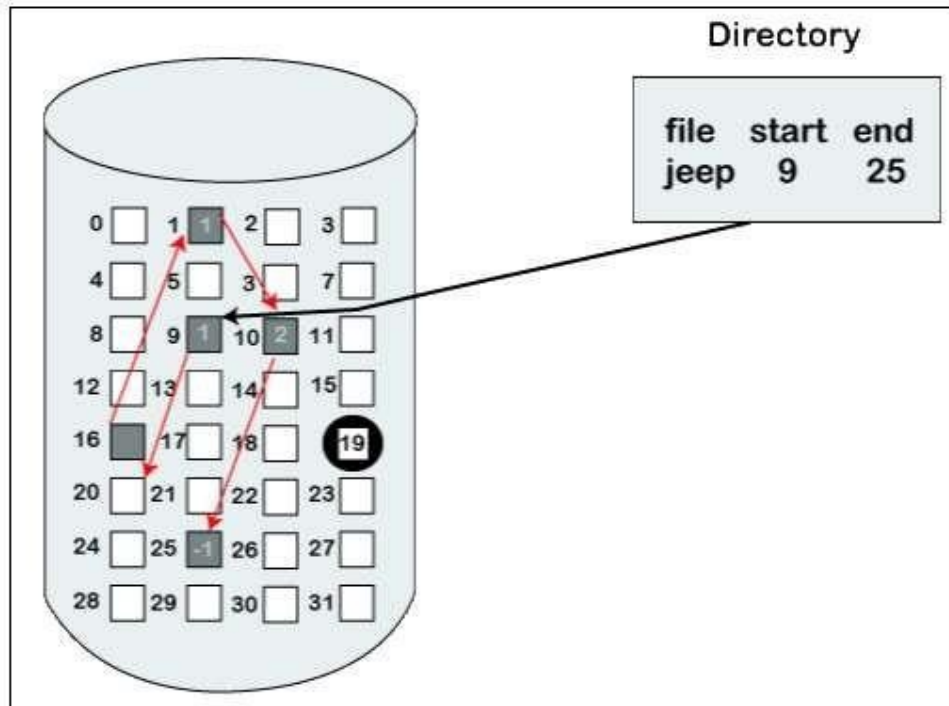
AdvantagesofLinkedlistallocation
Therearevariousadvantagesoflinkedlistallocation:

1. In liked list allocation, there is no external fragmentation. Due to this, we can utilize the memory better.
2. In linked list allocation, a directory entry only comprises of the starting block address.
3. Thelinkedallocationmethodisflexiblebecausewecanquicklyincreasethesize   of the file because, in this to allocate a file, we do not require a chunk of memory in a contiguous form.

DisadvantagesofLinkedlistAllocation
Therearevariousdisadvantagesoflinkedlistallocation:

1. Linkedlistallocationdoesnotsupportdirectaccessorrandomaccess.
2. Inlinkedlistallocation,weneedtotraverseeachblock.
3. Ifthepointerinthelinkedlistbreakinlinkedlistallocation,thenthefilegets corrupted.
4. Inthediskblockforthepointer,itneedssomeextraspace.

INDEXEDALLOCATION

The Indexed allocation method is another method that is used for file allocation. In theindex allocationmethod,wehaveanadditional block,andthatblockis knownas theindexblock.Foreachfile,thereisanindividualindexblock.Intheindexblock,the ith entry holds the disk address of the ith file block. We can see in the below figure that the directory entry comprises of the address of the index block.
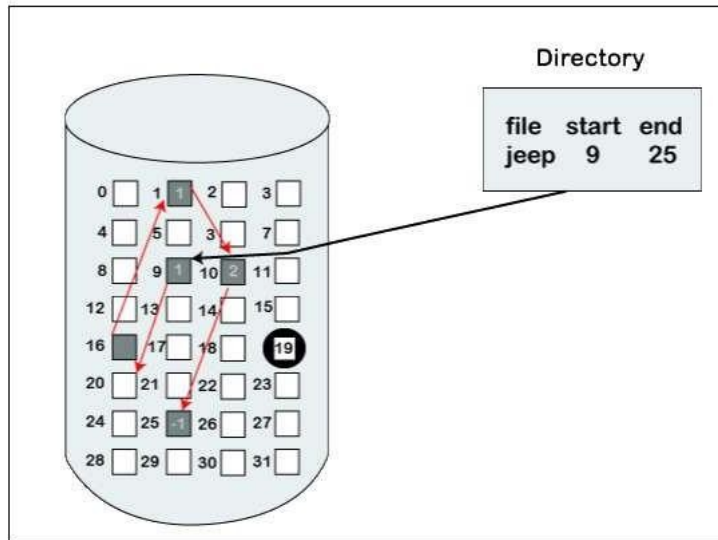
AdvantagesofIndexAllocation
Theadvantagesofindexallocationare:

1. Theindexallocationmethodsolvestheproblemofexternalfragmentation.
2. Indexallocationprovidesdirectaccess.

DisadvantagesofIndexAllocation
Thedisadvantagesofindexallocationare:

1. Inindexallocation,pointeroverheadismore.
2. Wecanlosetheentirefileifanindexblockisnotcorrect.
3. Itistotallyawastagetocreateanindexforasmallfile.

FILEPROTECTION,SECONDARYSTORAGEMANAGEMENT.

FILE PROTECTION

Wheninformationiskeptinacomputersystem,amajorconcernisitsprotection from physical damage (reliability) as well as improper access.

Typesofaccess: Incaseofsystemsthatdon'tpermitaccesstothefilesofother users. Protection is not needed. So, one extreme is to provide protection by prohibitingaccess.Theotherextremeistoprovidefreeaccesswithnoprotection. Both these approaches are too extreme for general use.

So, we need controlled access. It is provided by limiting the types of file access. Access is permitted depending on several factors. One major factor is type of access requested. The different type of operations that can be controlled are:

• Read

• Write

• Execute

• Append

• Delete

• List

Accesslistsand groups:

Various users may need different types of access to a file or directory. So, we can associate an access lists with each file and directory to implement identity dependent access. When a user access requests access to a particular file, the OS checkstheaccesslistassociatedwiththatfile.Ifthatuserisgrantedtherequested access, then the access is allowed. Otherwise, a protection violation occurs & the user is denied access to the file. But the main problem with access lists is their length.Itisverytedioustoconstructsuchalist.So,weuseacondensedversionof the access list by classifying

theusersinto3categories:

• Owners:Theuserwhocreatedthefile.

• Group:Asetofuserswhoaresharingthefiles.

• Others:Allotherusersinthesystem.

Hereonly3fieldsarerequiredtodefineprotection.Eachfieldisacollectionofbits each of which

eitherallowsorpreventstheaccess.E.g.TheUNIXfilesystemdefines3fieldsof3 bits each: rwx

• r(readaccess)

- w(writeaccess)

- x(executeaccess)

Separatefieldsarekeptforfileowners,group&otherusers.So,abitisneeded to record protection information for each file.

NEEDFORPROTECTINGFILES

- Need forprotectionisduetotheabilitytoaccessfiles
- Twowaystotackletheproblem
    1. Prohibitaccessprovidingcompleteprotection
   2. Providefreeaccesswithoutprotection
- Bothapproachesaretooextremeforgeneraluse.

SECONDARYSTORAGEANDDISKSCHEDULINGALGORITHMS

Secondary storage devices are those devices whose memory is non volatile, meaning,thestoreddatawillbeintactevenifthesystemisturnedoff.Herearea few things worth noting about secondary storage.

- Secondarystorageisalsocalledauxiliarystorage.
- Secondarystorageislessexpensivewhencomparedtoprimarymemorylike RAMs.
- Thespeedofthesecondarystorageisalsolesserthanthatofprimarystorage.
- Hence,thedatawhichislessfrequentlyaccessediskeptinthe secondary storage.
- Afewexamplesaremagneticdisks,magnetictapes,removablethumbdrives etc.

MAGNETICDISKSTRUCTURE

In modern computers, most of the secondary storage is in the form of magnetic disks.Hence,knowingthestructureofamagneticdiskisnecessarytounderstand how the data in the disk is accessed by the computer.
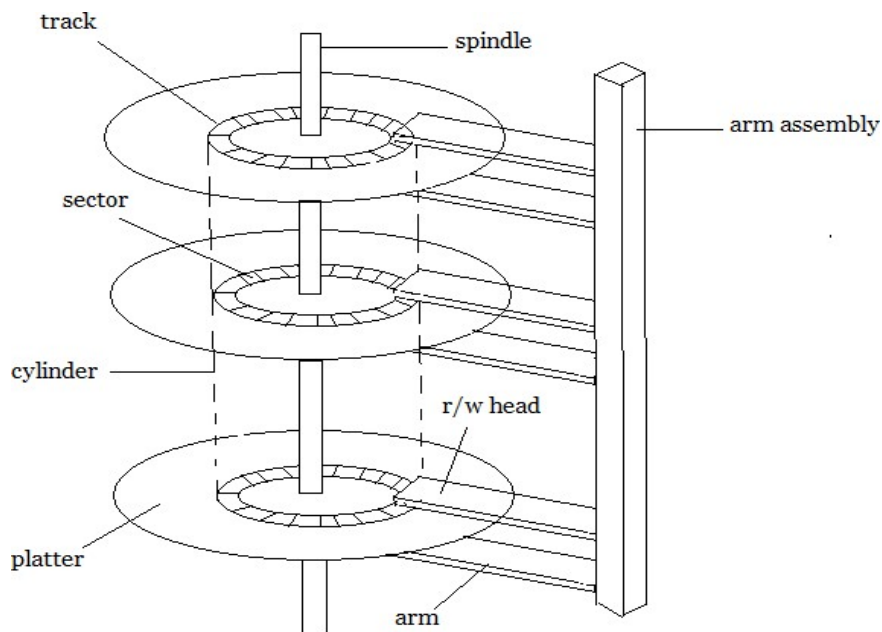
Structure of a magnetic disk

A magnetic disk contains several **platters**. Each platter is divided into circular shaped **tracks**. The length of the tracks near the centre is less than the length of the tracks farther from the centre. Each track is further divided into **sectors**, as shown in the figure.

Tracks of the same distance from centre form a cylinder. A read-write head is used to read data from a sector of the magnetic disk.

The speed of the disk is measured as two parts:

- **Transfer rate:** This is the rate at which the data moves from disk to the computer.

- **Random access time:** It is the sum of the seek time and rotational latency.

Seek time is the time taken by the arm to move to the required track. **Rotational latency** is defined as the time taken by the arm to reach the required sector in the track.

Even though the disk is arranged as sectors and tracks physically, the data is logically arranged and addressed as an array of blocks of fixed size. The size of a block can be **512** or **1024** bytes. Each logical block is mapped with a sector on the disk, sequentially. In this way, each sector in the disk will have a logical address.

SYSTEMPROGRAMMING

UNIT-7

systemprogramming
conceptsofsystemprogrammingandshowdifferencefromapplication
complier:
   compiler,functionsofcompiler.
   comparecompilerandinterpreter.
   sevenphasesofcompiler,briefdescriptionofeachphase

CONCEPTSOFSYSTEMPROGRAMMING
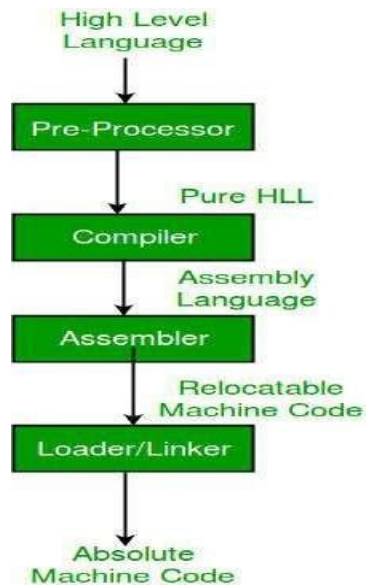
DEFINITION

System programming (or systems programming) is the activity of programming system software. The primary distinguishing characteristic of systems programming whencomparedtoapplicationprogrammingisthat applicationprogrammingaimsat producing software which provides services to the user (e.g. word processor, Spreadsheets, databases, Accounting packages)

Systems programming aims at producing software which provides services to the computerhardware(e.g.diskdefragmenter,OperatingSystems...).Italsorequiresa greaterdegreeofhardwareawarenessthatistosayitismachinedependent and the programmer needs to know the hardware within which the software will operate.

Languageprocessingsystems(usingCompiler)–

We know a computer is a logical assembly of Software and Hardware. The hardware knowsalanguage,thatishardforustograsp,consequently,wetendtowriteprogramsinahigh-levellanguage,thatismuchlesscomplicatedforustocomprehend and maintain in thoughts. Now, these programs go through a series of transformationssothattheycanreadilybeusedbymachines.Thisiswherelanguage procedure systems come in handy.

High Level
Language

Pre-Processor

Pure HLL

Compiler

Assembly
Language

Assembler

Relocatable
Machine Code

Loader/Linker

Absolute
Machine Code

- **High-LevelLanguage**–Ifaprogramcontains#defineor#includedirectivessuch as #include or #define it is called HLL. They are closer to humans but far from machines.These(#)tagsarecalledpreprocessordirectives.Theydirectthepre-processor about what to do.
- **Pre-Processor** –The pre-processor removes all the #include directives by includingthefilescalledfileinclusionandallthe#definedirectivesusingmacro expansion. It performs file inclusion, augmentation, macro-processing, etc.
- **Assembly Language** –It's neither inbinary form nor high level. It is an intermediatestatethatis acombinationof machineinstructions andsome otherusefuldataneededforexecution.
- **Assembler** –For every platform (Hardware + OS) we will have an assembler. They are not universal since for each platform we have one. The output of the assembler is called an object file. Its translates assembly language to machine code.
- **Interpreter** –An interpreter converts high-level language into low-level machine language, just like a compiler. But they are different in the way they read the input. The Compiler in one go reads the inputs, does the processing, and executes the source code whereas the interpreter does the same line by line. Compiler scans the entire program and translates it as a whole into machinecodewhereasaninterpretertranslatestheprogramonestatementat atime.Interpretedprogramsareusuallyslowerwithrespecttocompiledones.

- **RelocatableMachine Code** –It can be loaded atany point and can be run. The addresswithintheprogramwillbeinsuchawaythatitwillcooperatewiththe program movement.
- **Loader/Linker** –It converts the relocatable code into absolute code and tries to run the program resulting in a running program or an error message (or sometimes both can happen). Linker loads a variety of object files into a single file to make it executable. Then loader loads it in memory and executesit.

**DIFFERENCESBETWEENSYSTEMPROGRAMMINGANDAPPLICATIONCOMPILER:**

SYSTEM PROGRAMMING	APPLICATIONCOMPILER
1. System programming aims at producingsoftwareandsoftware platforms(such as OS or web browsers or application programming interface) which provides services to others platform.	1. A compiler like all applications relies on the OS to load it into memory and start its execution.
2. System programming provides an environment where program can be developed and executed.	2. Application compiler is used to converthighlevellanguagetolower level language or machine code.
3. examples of system software includes os, computational sciences s/w, device driver, assembler,interpreter,compiler etc.	3. Examples of application compiler areC,C++,Javaetc.

COMPILER
FUNCTIONSOFCOMPILER.

INTRODUCTIONTOCOMPILER

- ○ Acompilerisatranslatorthatconvertsthehigh-levellanguageintothe machine language.
- ○ High-levellanguageiswrittenbyadeveloperandmachinelanguagecanbe understood by the processor.
- ○ Compilerisusedtoshowerrorstotheprogrammer.

- Themainpurposeofcompileristochangethecodewritteninonelanguage without changing the meaning of the program.
- WhenyouexecuteaprogramwhichiswritteninHLLprogramminglanguage then it executes into two parts.
- Inthefirstpart,thesourceprogramcompiled and translatedintotheobject program (low level language).
- Inthesecondpart,objectprogramtranslatedintothetargetprogramthrough the assembler.
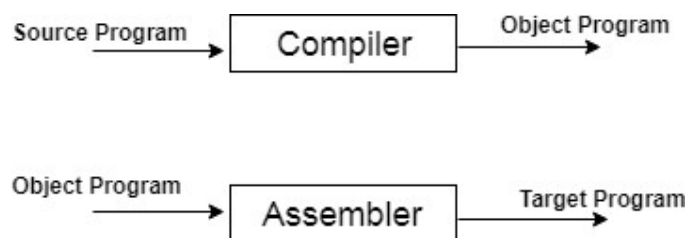
FUNCTIONSOFCOMPILER:

i. Detects errors during translation and gives suggestions on how to correct them.

ii. Linksthesubroutines(orafunction,procedure,andsubprogram,is code that may be called and executed anywhere in a program) usedin program instructions.

iii. Converts a source code written using high level programming languageinto machine code.

iv. Generatesexecutable(.exe)objectcodefile.

## COMPARECOMPILERANDINTERPRETER.

Interpreter

Interpreterisatranslatorwhichisusedtoconvertprogramsinhigh-levellanguage tolow-levellanguage.Interpretertranslateslinebylineandreportstheerroronce it encountered during the translation process.

Itdirectlyexecutes theoperationsspecifiedinthesourceprogramwhentheinput is given by the user.

Itgivesbettererrordiagnosticsthanacompiler.
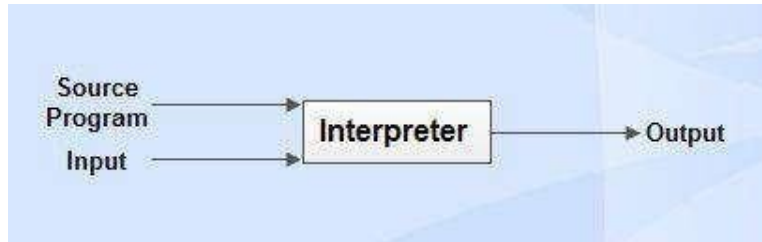
| Compiler	Interpreter
Acompilertranslatestheentire source code in a single run.	An interpreter translates the entiresourcecodelinebyline.
Itconsumeslesstimei.e.,itis faster than an interpreter.	Itconsumesmuchmoretimethan the compiler i.e., it is slower than the compiler.
Itismoreefficient.	Itisless efficient.
CPUutilizationis more.	CPUutilizationislessascompared to the compiler.
Bothsyntacticandsemantic errors can be checked simultaneously.	Onlysyntacticerrorsarechecked.
Thecompilerislarger.	Interpretersareoftensmallerthan compilers.
Itisnotflexible.	Itisflexible.
Thelocalizationoferrorsis difficult.	Thelocalizationoferroriseasier than the compiler.

Compiler	Interpreter
Apresenceofanerrorcancause the whole program to be re-organized.	Apresenceofanerrorcausesonly a part of the program to be re-organized.
Thecompilerisusedbythe language such as C, C++.	Aninterpreterisusedby languages such as Java.

**SEVENPHASESOFCOMPILER,BRIEFDESCRIPTIONOFEACHPHASE**

PHASESOFCOMPILER:

Compiler operates in various phases each phase transforms the source programfromonerepresentationtoanother.Everyphasetakesinputsfromits previous stage and feeds its output to the next phase of the compiler.
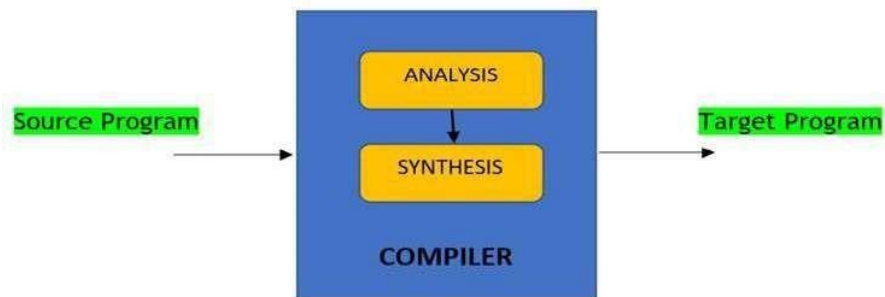
Thestructureofcompilerconsistsoftwoparts:

1. *Analysisphase*

- Analysis part breaks the source program into constituent pieces and imposesagrammaticalstructureonthemwhichfurtherusesthis structure to create an intermediate representation of the sourceprogram.

- Itisalsotermedasfrontendofcompiler.

- Informationaboutthesourceprogramiscollectedandstoredinadata
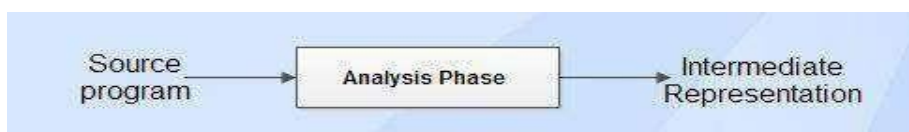
- structurecalledsymboltable.

Fig.2.0.DiagramofAnalysis phase

2. *Synthesisphase*

• Synthesisparttakestheintermediaterepresentationasinputand transforms it tothe target program.
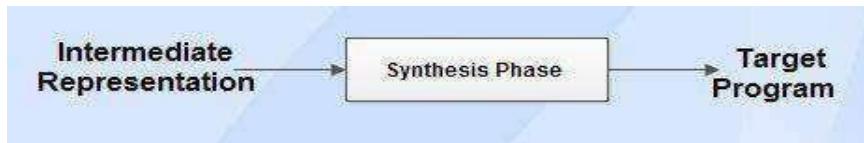
• Itisalsotermedasbackendofcompiler.

Fig.3.0.DiagramofSynthesisphase

Thedesignofcompilercanbedecomposedintoseveralphases,eachofwhich converts one form of source program into another

Thedifferentphasesofcompilerareasfollows:

1. Lexicalanalysis
2. Syntaxanalysis
3. Semanticanalysis
4. Intermediatecodegeneration
5. Codeoptimization
6. Codegeneration

Alloftheaforementionedphasesinvolvethefollowingtasks:

• Symboltablemanagement.

• Error handling.
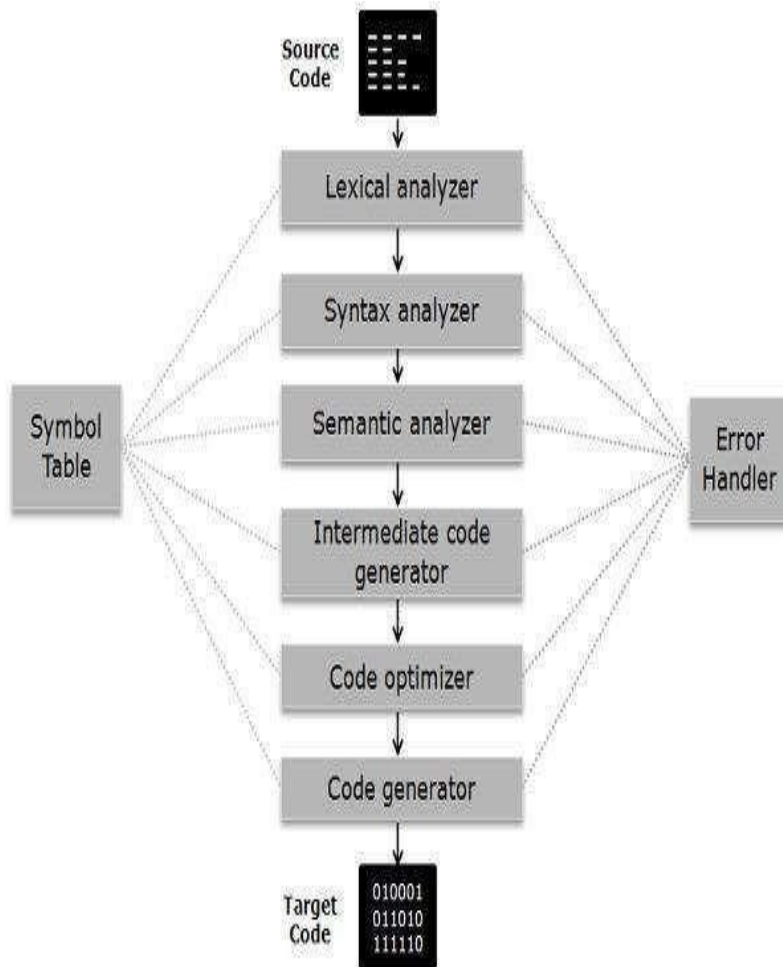
Fig.4.0.Blockdiagramofphasesofcompiler

1. LEXICALANALYSIS

• Lexicalanalysisisthefirstphaseofcompilerwhichisalsotermedasscanning.

  • Source program is scanned to read the stream of characters and those characters are grouped to form a sequence called lexemes which produces token as output.

- **Token:** Token is a sequence of characters that represent lexical unit, which matches with the pattern, such as keywords, operators, identifiers etc.
- **Lexeme:** Lexeme is instance of a token i.e., group of characters forming a token.

,

- **Pattern:** Pattern describes the rule that the lexemes of a token takes. It is the structure that must be matched by strings.
- Once a token is generated the corresponding entry is made in the symbol table.

Input: stream of characters Output: Token

Token Template: <token-name,attribute-value>

(eg.)c=a+b*5;

Lexemes and tokens

Lexemes	Tokens
c	Identifier
=	assignment symbol
a	Identifier
+	+(addition symbol)
b	Identifier
*	*(multiplication symbol)
5	5 (number)

Hence,<id,1><=><id,2><+><id,3><*><5>

2. SYNTAX ANALYSIS

- Syntax analysis is the second phase of compiler which is also called as parsing.
- Parser converts the tokens produced by lexical analyzer into a tree like representation called parse tree.

- A parse tree describes the syntactic structure of the input.

• Syntax tree is a compressed representation of the parse tree in which the operatorsappearasinteriornodesandtheoperandsoftheoperatorarethe children of the node for that operator.

Input: Tokens

Output: Syntaxtree

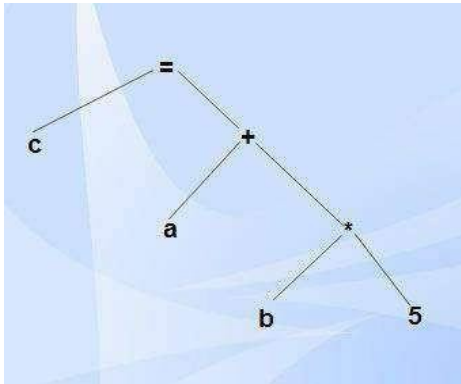*Fig.5.0.DiagramofSyntax tree*

3.SEMANTICANALYSIS

• Semanticanalysisisthethirdphaseofcompiler.

• Itchecksforthesemanticconsistency.

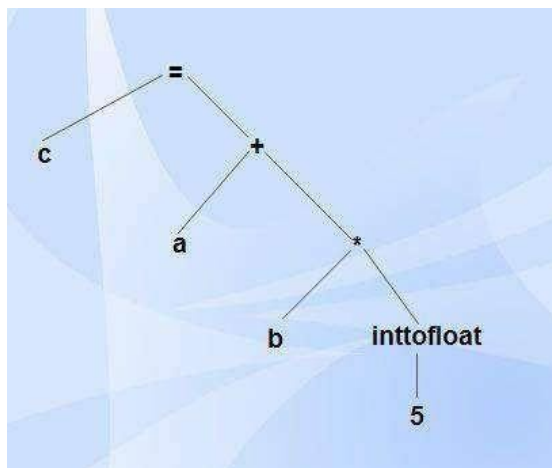• Typeinformationisgatheredand storedinsymboltableorinsyntaxtree.

• Performstypechecking.

Fig.6.0.DiagramofSemanticTree

4. INTERMEDIATE CODE GENERATION

- Intermediate code generation produces intermediate representations for the source program which are of the following

    forms: o Postfix notation

    o ree address code

    o Syntax tree

Most commonly used form is the three-address code.

to float(5) t2 = id3*tl

t3 = id2+t2 id1

= t3

Properties of intermediate code

- It should be easy to produce.
- It should be easy to translate into target program.

5. CODE OPTIMIZATION

- Code optimization phase gets the intermediate code as input and produces optimized intermediate code as output.
- It results in faster running machine code.
- It can be done by reducing the number of lines of code for a program.
- This phase reduces the redundant code and attempts to improve the intermediate code so that faster-running machine code will result.
- During the code optimization, the result of the program is not affected.
- To improve the code generation, the optimization involves
    - o Deduction and removal of dead code (unreachable code).
    - o Calculation of constants in expressions and terms.
    - o Collapsing of repeated expression into temporary string.
    - o Loop unrolling.

o Movingcodeoutsidetheloop.

o Removalofunwantedtemporaryvariables.

$$t1 = id3* 5.0$$

$$id1 = id2 + t1$$

6. CODEGENERATION

• Codegenerationisthefinalphaseofacompiler.

• Itgetsinputfromcodeoptimizationphaseandproducesthetargetcodeorobject code as result.

• Intermediateinstructionsaretranslatedintoasequenceofmachineinstructions that perform the same task.

• Thecodegenerationinvolves

o Allocationofregisterandmemory.

o Generationofcorrectreferences.

o Generationofcorrectdatatypes.

o Generationofmissingcode.

LDF	R2, id3
MULF	R2,#5.0
LDF	R1, id2
ADDF	R1,R2
STF	id1,R1

7(a)SYMBOLTABLEMANAGEMENT

• Symboltableisusedtostorealltheinformationaboutidentifiers used in theprogram.

• Itisadatastructurecontainingarecordforeachidentifier,withfieldsforthe attributes of the identifier.

• Itallowsfindingtherecordforeachidentifierquicklyandtostoreorretrieve data from that record.

• Wheneveranidentifierisdetectedinanyofthephases,itisstoredinthesymbol

table.

Example

inta,b;floatc;charz;

Symbolname	Type	Address
A	Int	1000
B	Int	1002
C	Float	1004
Z	char	1008

Example

doubletest(doublex);doublesample(intcount)

{

  doublesum=0.0;

      for(inti=1;i<=count;i++)

       sum+=   test((double)   i);

       return sum;

  }

Symbolname	Type	Scope
Test	function,double	extern
X	double	function parameter
Sample	function,double	global
Count	int	function parameter
Sum	double	blocklocal
I	int	for-loop statement

7(b)ERRORHANDLING

• Eachphasecanencountererrors.Afterdetectinganerror,aphasemusthandle the error so that compilation canproceed.

• Inlexicalanalysis,errorsoccurinseparationoftokens.

• Insyntaxanalysis,errorsoccurduringconstructionofsyntaxtree.

• Insemanticanalysis,errorsmayoccuratthefollowingcases:

(i) Whenthecompilerdetectsconstructsthathaverightsyntacticstructure but nomeaning

(ii) Duringtypeconversion.

• Incodeoptimization,errorsoccurwhentheresultisaffectedbytheoptimization. In code generation, it shows error when code is missingetc.

ErrorEncounteredinDifferentPhases

Eachphasecanencountererrors.Afterdetectinganerror,aphasemustsome howdeal with the error, so that compilation can proceed.
Aprogrammayhavethefollowingkindsoferrorsatvarious stages:

LexicalErrors

Itincludesincorrectormisspellednameofsomeidentifieri.e.,identifierstyped incorrectly.

SyntacticalErrors

It includes missing semicolon or unbalanced parenthesis. Syntactic errors arehandled by syntax analyzer (parser).
When an error is detected, it must be handled by parser to enable the parsing of the rest of the input. In general, errors may be expected at various stages of compilation but most of the errors are syntactic errors and hence the parser shouldbe able to detect and report those errors in the program.

Thegoalsoferrorhandlerinparserare:

• Reportthepresenceoferrorsclearlyandaccurately.
• Recoverfromeacherrorquicklyenoughtodetectsubsequenterrors.
• Addminimaloverheadtotheprocessingofcorrectingprograms.

There are four common error-recovery strategies that can be implemented in the parser to deal with errors in the code.

- o Panicmode.
- o Statementlevel.
- o Errorproductions.
- o Globalcorrection.

SemanticalErrors

These errors are a result of incompatible value assignment. The semantic errors that the semantic analyzer is expected to recognize are:

- Typemismatch.
- Undeclaredvariable.
- Reservedidentifiermisuse.
- Multipledeclarationofvariableinascope.
- Accessinganoutof scopevariable.
- Actualandformalparametermismatch.

Logicalerrors
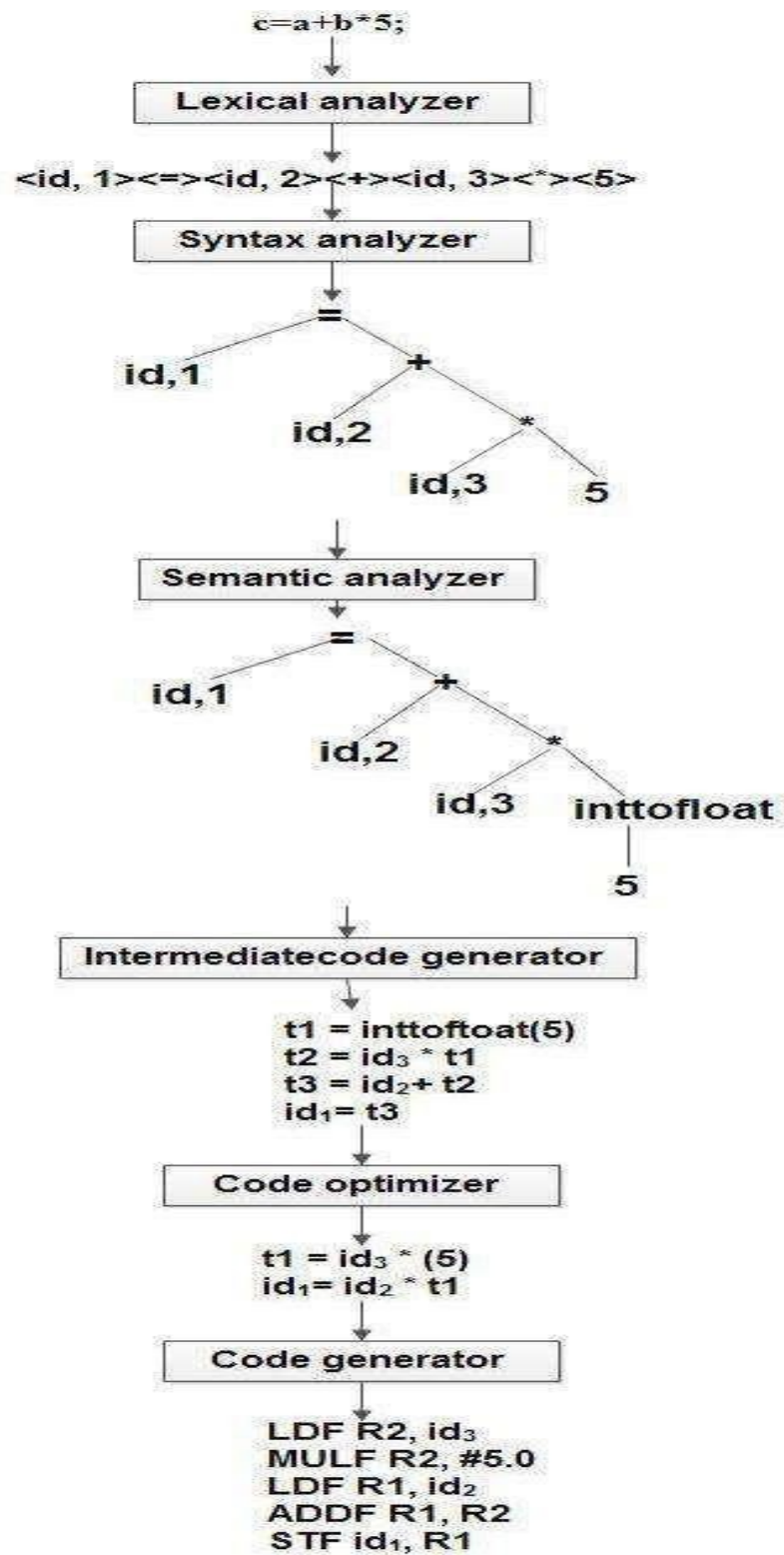
These errors occur due to not reachable code-infiniteloop.

Fig.7.0FlowchartofPhasesofCompiler