Q.1(a) Attempt any THREE of the following : [12]

Q.1(a) (i) Describe real time operating system in brief. [4]

(A) Real time systems are used in environment where a large number of events, mostly external to the computer system, must be accepted and processes in a short time or within certain deadlines. Such applications include real-time simulations, flight control, industrial control, military applications etc.

A primary objective of real-time systems is to provide quick event response time and thus meet the scheduling deadlines. User convenience and resource utilization are of secondary concern to realtime system designers.

In Real time systems, processor is allocated to the highest priority process among those that are ready to execute. Higher priority processes preempt execution of the lower priority processes. This form is called as 'priority –based preemptive scheduling'.

The primary functions of the real time operating system are to:

1. Manage the processor and other system resources to meet the requirements of an application.
2. Synchronize with and respond to the system events.
3. Move the data efficiently among processes and to perform coordination among these processes.

Types of real time system:

1. **Hard real time :** Hard real time means strict about adherence to each task deadline. When an event occurs, it should be serviced within the predictable time at all times in a given hard real time system.
   Example : video transmission, each picture frame and audio must be transferred at fixed rate.
2. **Soft real time :** Soft real time means that only the precedence and sequence for the task operations are defined, interrupt latencies and context switching latencies are small. There can be few deviations between expected latencies of the tasks and observed time constraints and a few deadline misses are accepted.
   Example: Mobile phone, digital cameras and orchestra playing robots.

Q.1(a) (ii) What is process management? State four functions to be performed by OS for process management. [4]

(A) **Process Management :** The operating system manages many kinds of activities ranging from user programs to system programs like printer spooler, name servers, file server etc. Each of these activities is encapsulated in a process. A process includes the complete execution context (code, data, PC, registers, OS resources in use etc.). The basic unit of software that the operating system deals with in scheduling the work done by the processor is either a process or a thread, depending on the operating system.

A processes software that performs some action and can be controlled by a user, by other applications or by the operating system. A process needs various system resources including CPU time, memory, files and I/O devices to complete the job execution. These resources can be given to the process when it is created or allocated to it while it is running.

The five major activities of an operating system in regard to process management are:

- Creation and deletion of user and system processes.
- Suspension and resumption of processes.
- A mechanism for process synchronization.
- A mechanism for process communication.
- A mechanism for deadlock handling.

Q.1(a) (iii) What is file? List and explain attributes of files. [4]

(A) **Definition of file:** A file is named collection of related information that is recorded on secondary storage. It is a sequence of bits, bytes, lines or records representing data. Data can be stored on secondary storage only when it is place inside the file. File represents programs and data. Data files may be numeric, alphabetic, alphanumeric or binary. Files are mapped by the operating system onto physical devices.

Attributes of file

Name. It is a string of characters which is in human readable form.

Identifier. This unique tag, usually a number, identifies the file within the file system; it is the non-human-readable name for the file.

Type. This is the information used by the system to support different types of the files.

Location. This information is a pointer to a device and to the location of the file on that device.

Size. The current size of the file (in bytes, words, or blocks) and possibly the maximum allowed size are included in this attribute.

Protection. Access-control information determines who can do reading, writing, executing, and so on.

Time, date, and user identification. This information may be kept for creation, last modification, and last use.

Attribute	Meaning
Protection	Who can access the file and in what way
Password	Password needed to access the file
Creator	ID of the person who created the file
Owner	Current owner
Read-only flag	0 for read/write; 1 for read only
Hidden flag	0 for normal; 1 for do not display in listings
System flag	0 for normal files; 1 for system file
Archive flag	0 for has been backed up; 1 for needs to be backed up
ASCII/binary flag	0 for ASCII file; 1 for binary file
Random access flag	0 for sequential access only; 1 for random access
Temporary flag	0 for normal; 1 for delete file on process exit
Lock flags	0 for unlocked; nonzero for locked
Record length	Number of bytes in a record
Key position	Offset of the key within each record
Key length	Number of bytes in the key field
Creation time	Date and time the file was created
Time of last access	Date and time the file was last accessed
Time of last change	Date and time the file has last changed
Current size	Number of bytes in the file
Maximum size	Number of bytes the file may grow to

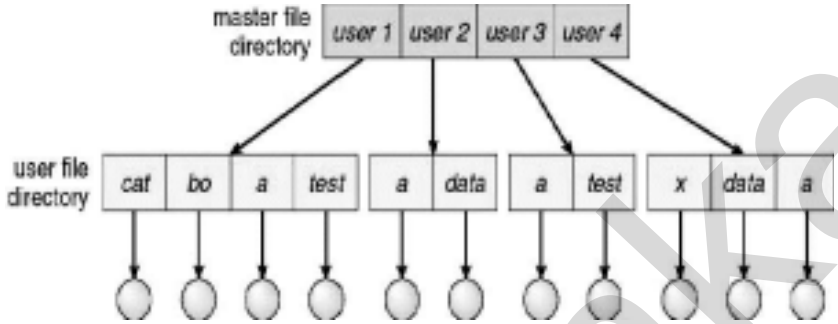
Q.1(a) (iv) Explain two level directory structure with the help of diagram. [4]

(A) The standard solution to limitations of single-level directory is to create a separate directory for each user.

In the two-directory structure, each user has his own user file directory (UFD). The UFDs have similar structures, but each lists only the files of a single user.

When a user job starts or a user logs in, the system's master file directory (MFD) is searched.

The MFD is indexed by user name or account number, and each entry points to the UFD for that user.



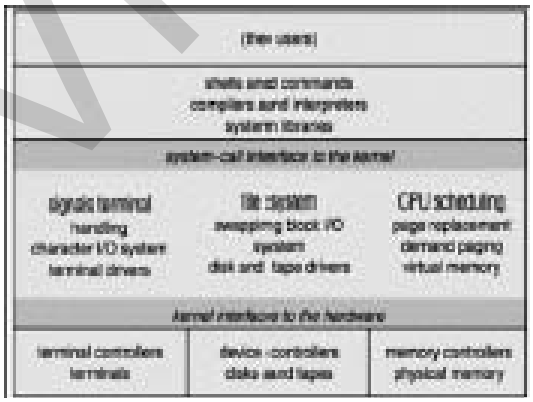
Q.1(b) Attempt any ONE of the following : [6]

Q.1(b) (i) Describe following operating system structures. [6]

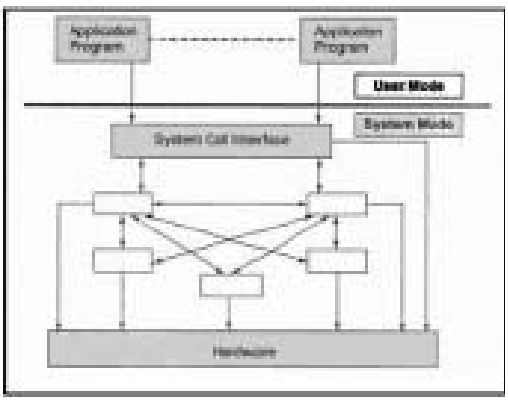
(1) Monolithic (2) Microkarnel

(A) **Monolithic Systems**: The structure is that there is no structure. The operating system is written as a collection of procedures, each of which can call any of the other ones whenever it needs to. When this technique is used, each procedure in the system has a well-defined interface in terms of parameters and results, and each one is free to call any other one, if the latter provides some useful computation that the former needs.

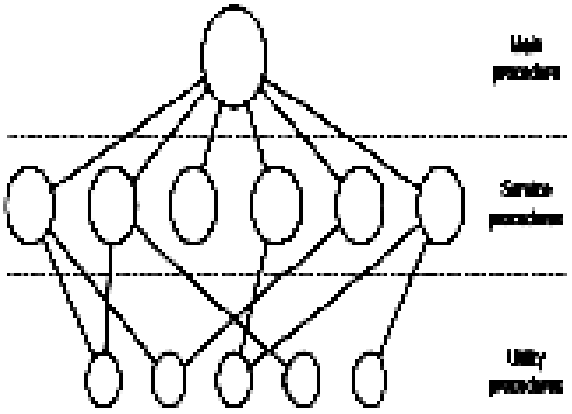
For constructing the actual object program of the operating system when this approach is used, one compiles all the individual procedures, or files containing the procedures, and then binds them all together into a single object file with the linker. In terms of information hiding, there is essentially none- every procedure is visible to every other one i.e. opposed to a structure containing modules or packages, in which much of the information is local to module, and only officially designated entry points can be called from outside the module.



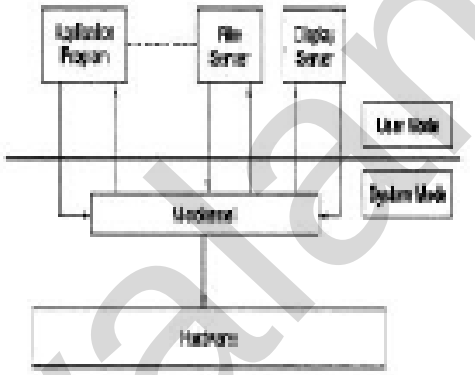
OR



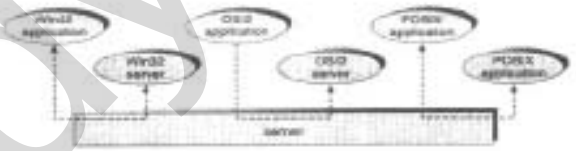
OR



Microkarnel : A microkernel (also known as -kernel) is the near-minimum amount of software that can provide the mechanisms needed to implement an operating system (OS). These mechanisms include low-level address space management, thread management, and inter-process communication (IPC).If the hardware provides multiple rings or CPU modes, the microkernel is the only software executing at the most privileged level (generally referred to as supervisor or kernel mode).Moves as much from the kernel into "user" space. Communication takes place between user modules using message passing.



OR



Q.1(b) (ii) Explain any six services provided operating system. Draw diagram showing services. **[6]**

(A) (1) User interface
(2) Program execution
(3) I/O operations
(4) File-system manipulation
(5) Communications
(6) Error detection
(7) Accounting
(8) Resource allocation
(9) Protection and security

(1) User interface : Almost all operating systems have a user interface (UI). The interface can take several forms. One is a DTrace command-line interface (CLI), which uses text commands and a method for entering them (say, a program to all entering and editing of

commands). Another is a batch interface, in which commands and directive to control those commands are entered into files, and those files are executed. Most commonly, a graphical user interface (GUI) is used.

(2) Program execution : The operating system loads the contents (or sections) of a file into memory and begins its execution. A user-level program could not be trusted to properly allocate CPU is time.

(3) I/O operations : Disks, tapes, serial lines, and other devices must be communicated with at a very low level. The user need only specify the device and the operation to perform on it, while the system converts that request into device or controller-specific commands. User-level programs cannot be trusted to access only devices they should have access to and to access them only when they are otherwise unused.

(4) File-system manipulation : There are many details in file creation, deletion, allocation, and naming that users should not have to per-form. Blocks of disk space are used by files and must be tracked. Deleting a file requires removing the name file information and freeing the allocated blocks. protections must also be checked to assured proper file access. User programs could neither ensure adherence to protection methods nor be trusted to allocate only free blocks and deallocate blocks on file deletion.

(5) Communications : Message passing between systems requires message to be turned intopackets of information, sent to the net-work controller, transmitted across a communications medium, and reassembled by the destination system. Packed ordering and data correction must take place. Again, user programs might not coordinate access to the network device, or they might receive packets destined for other processes.

(6) Error detection : Error detection occurs at both the hardware and software levels. At the hardware level, all data transfers must be inspected to ensure that data have not been corrupted in transit. All data on media must be checked to be sure they have not changed since they were written to the media. At the software level, media must be checked for data consistency; for instance, whether the number of allocated and unallocated blocks of storage match the total number on the device. There, errors are frequently process-independent (for instance, the corruption of data on a disk), so there must be a global program (the operating system) that handles all types of errors. Also, by having errors processed by the operating system, processes need not contain code to catch and correct all the error possible on a system.

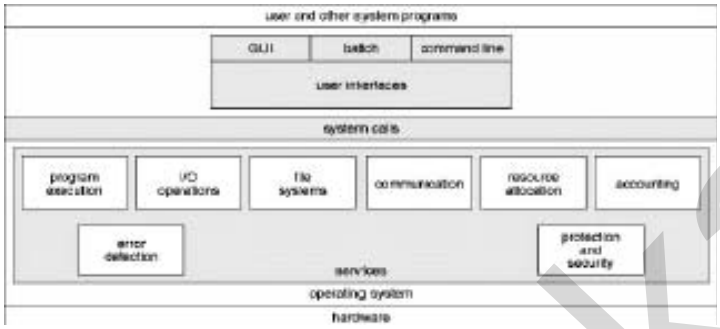
7. **Accounting :** We may want to keep track at which users use how must and what kind of computer resources. What was the login time for a particular user; is he working on the system right now, what is the process - 1 D for the user, all such in formations we can manage using accounting service provided by many multiuser systems. This record keeping may be for the purpose of paying for the system and its operation, or simply for accounting usage statistics.

8. **Resource allocation :** When there are multiple users or multiple jobs running at the same time. Resources must be allocated to each of them. Many different types of resources are managed by the operating system. Some (Such as CPU cycles, main memory, and file storage) may have special allocation code, whereas other (such as I/O devices) may have much more general request and release code.

9. **Protection and security :** The owners of information stored in multiuser or networked computer system may want to control use of the information. When several separate processes execute concurrently, it should not be possible for one process to interfere

with the others or with the operating system itself, and protection involves ensuring that all access to system resource is controlled. Security of the system from outsiders is also important. Such security starts with requiring each user to authenticate himself or herself to the system, usually by means of a password, to gain access to system resources. It extends to defending external I/O devices, including modems and network adapters, from invalid access attempts and to recording all such connections for detection of break-ins. If a system is to be protected and secure, precautions must be instituted throughout it. A chain is only as strong as its weakest link.

Diagram for services :



Q.2 Attempt any FOUR of the following : [16]

Q.2(a) With neat diagram describe use of Process Control Block (PCB). [4]

(A) **PCB** is a record or a data structure that is maintained for each and every process. Every process has one PCB that is associated with it. A PCB is created when a process is created and it is removed from memory when process is terminated.

A PCB contains several types of information depending upon the process to which PCB belongs.

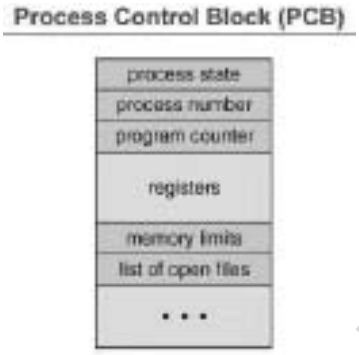
The information stored in PCB of any process may vary from process to process.

In general, a PCB may contain information regarding:

1. **Process Number**: Each process is identified by its process number, called process identification number (PID). Every process has a unique process-id through which it is identified. The Process-id is provided by the OS. The process id of two process could not be same because psid is always unique.
2. **Priority:** Each process is assigned a certain level of priority that corresponds to the relative importance of the event that it services Process priority is the preference of the one process over other process for execution. Priority may be given by the user/system manager or it may be given internally by OS. This field stores the priority of a particular process.
3. **Process State:** This information is about the current state of the process. I.e. whether process is in new, ready, running, waiting or terminated state.
4. **Program Counter:** This contains the address of the next instruction to be executed for this process.
5. **CPU Registers:** CPU registers vary in number and type, depending upon the computer architectures. These include index registers, stack pointers and general purpose registers etc.

   When an interrupt occurred, information about the current status of the old process is saved in registers along with the program counters. This information is necessary to allow the process to be continued correctly after the completion of an interrupted process.
6. **CPU Scheduling Information:** This information includes a process priority, pointers to scheduling queues and any other scheduling parameters.

7. **Memory Management Information:** This information may include such information as the value of base and limit registers, the page table or the segment table depending upon the memory system used by operating system.
8. **Accounting**: This includes actual CPU time used in executing a process in order to charge individual user for processor time.
9. **I/O Status:** It includes outstanding I/O request, allocated devices information, pending operation and so on.
10. **File Management:** It includes information about all open files, access rights etc.



Q.2(b) Define the following terms : [4]

(i) Preemptive scheduling (ii) Nonpreemptive scheduling

(A) (i) Preemptive scheduling

- Even if CPU is allocated to one process, CPU can be preempted to other process if other process is having higher priority or some other fulfilling criteria.
- It is suitable for RTS.
- Only the processes having higher priority are scheduled.
- It doesn't treat all processes as equal.
- Circumstances for preemptive
  - process switch from running to ready state
  - process switch from waiting to ready state

(ii) Nonpreemptive scheduling

- Once the CPU has been allocated to a process the process keeps the CPU until it releases CPU either by terminating or by switching to waiting state.
- It is not suitable for RTS.
- Processes having any priority can get scheduled.
- It treats all process as equal.
- Circumstance for Non preemptive
  - Process switches from running to waiting state
  - Process terminates

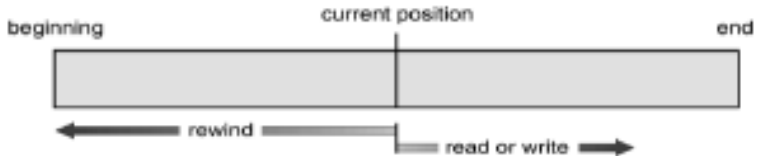
Q.2(c) Describe working of sequential and direct access methods. [4]

(A) Sequential access methods : The simplest access method is sequential access. Information in the file is processed in order, one record after the other.

This mode of access is by far the beginning current position most common; for example, editors and compilers usually access files in this fashion.

Reads and writes make up the bulk of the operations on a file.

- A read operation read next reads the next portion of the file and automatically advances a file pointer, which tracks the I/O location.
- Similarly, the write operation write next appends to the end of the file and advances to the end of the newly written material (the new end of file).



To read a piece of data that is stored at the end of the file, one has to read all of the data that comes before it-you cannot jump directly to the desired data. This is similar to the way cassette tape players work. If one wants to listen to the last song on a cassette tape, he has to either fast-forward over all of the songs that come before it or listen to them. There is no way to jump directly to a specific song.

Direct Access Method: A file is made up of fixed-length logical records that allow programs to read and write records rapidly in no particular order. Thus, we may read block 14, then read block 53, and then write block 7. There are no restrictions on the order of reading or writing for a direct-access file.

The direct-access method is based on a disk model of a file, since disks allow random access to any file block. Direct-access files are of great use for immediate access to large amounts of information. Databases are often of this type. For the direct-access method, the file operations must be modified to include the block number as a parameter.

The block number provided by the user to the OS is normally a relative block number.

- A relative block number is an index relative to the beginning of the file.
- Thus, the first relative block of the file is 0, the next is 1, and so on, even though the actual absolute disk address of the block may be 14703 for the first block and 3192 for the second.

The use of relative block numbers allows the OS to decide where the file should be placed (called the allocation problem) and helps to prevent the user from accessing portions of the file system that may not be part of her file.

When you work with a direct access file (which is also known as a random access file), you can jump directly to any piece of data in the file without reading the data that comes before it. This is similar to the way a CD player or an MP3 player works. You can jump directly to any song that you want to listen to. Sequential access files are easy to work with, and you can use them to gain an understanding of basic file operations.

Q.2(d) What is clustered system? Explain it. [4]

(A) **Clustered system :** Cluster is a group of interconnected, whole computers working together as a unified computing source that can create the illusion of being one machine. Each computer in a cluster is typically referred to as a node. Clustering (means gather together) allows two or more system to share storage closely linked via a local area network. Asymmetric Cluster (at least two servers: One is on a standby mode while the other is monitoring the other one. If one stops other will work). Symmetric Cluster (all work at the same level: They work together and monitor each other).

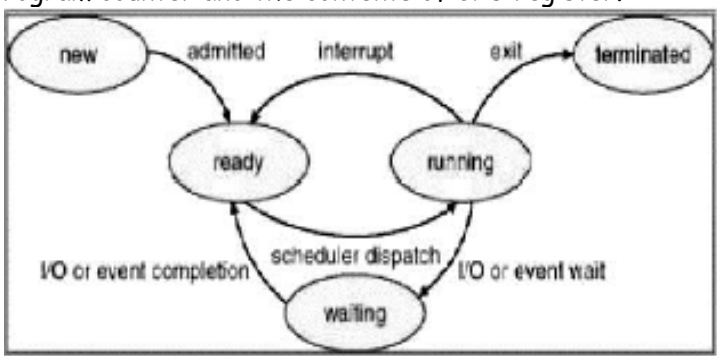
Cluster : is collection of computer in which any member of the cluster is capable of supporting the processing function of any other member. A cluster has n+k configuration where n processing nodes are actively processing the application and k processing nodes are in a standby state, serving as a spares. In the event of a failure of an active node, the application that was running on the failed node is moved to one of the standby nodes.

Other common cluster configuration include Simplex (one active node, no spare), n+1 active node (n active nodes, 1 spare)

Clustered system can be implemented using LAN. This system is a subsystem of a telecommunication switching system, running certain centralized application function. A LAN is interconnecting hub that provides connectivity with each other in the switching system.

Q.2(e) Draw and explain process state diagram. **[4]**

(A) Process is a program in execution. A process does not mean only program but it could contain some part called as text section. It may contain the current activity represented by the value of the program counter and the contents of CPU register.



Process States : A process is typically in one of the three states

(1) Running : has the CPU

(2) Blocked : waiting for I/O or another thread

(3) Ready to run : on the ready list, waiting for the CPU during the lifespan of a process, its execution status may be in one of four states : (associated with each state is usually a queue on which the process resides)

New : The process being created is available in the new state. It is the new state because the system is not permitted it to enter the ready state due to limited memory available in the ready queue. If some memory becomes available, then the process from the new state will go to ready state.

Running State : The process which is currently running and has control of the CPU is known as the process in running state. In single user system, there is only one process which is in the running state. In multiuser system, there are multiple processes which are in the running state.

Blocked State : The process is currently waiting on external event such as an I/O operation is said to be in blocked state. After the completion of I/O operation, the process from blocked state enters in the ready state and from the ready state when the process turn come it will again go to running state.

Terminated/Halted State : The process whose operation is completed, it will go the terminated state from the running state. the memory occupied by the process is released.

Q.2(f) Compare UNIX and LINUX with respect to following points. **[4]**
User interface, number of shells, providers, processing speed.

(A)

CRITERIA	LINUX	UNIX
User interface	Linux typically provides two GUIs, KDE and Gnome. But there are millions of alternatives such as LXDE, Xfce, Unity, Mate, twm, ect.	Initially Unix was a command based OS, but later a GUI was created called Common Desktop Environment. Most distributions now ship with Gnome.
Number of shells	Sh, bash, csh and tsh, ksh	B, C, K, Bash, tcsh, zsh
Processing speed	Low: As it is GUI based processing time is more as compare to UNIX.	High: As it is command based direct interpretation of commands is done so it takes less time as compare to LINUX.

Q.3 Attempt any FOUR of the following : [16]

Q.3(a) List any four operating system services and describe in one/two sentences. [4]

(A) User interface

(1) Program execution
(2) I/O operations
(3) File-system manipulation
(4) Communications
(5) Error detection
(6) Accounting
(7) Resource allocation
(8) Protection and security

Description of services of operating system

(1) User interface: Almost all operating systems have a user interface (UI).Almost all operating systems have a user interface (UI). It varies between Command-Line (CLI), Graphics User Interface (GUI).

(2) Program execution: The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)

(3) I/O operations: Since user programs cannot execute I/O operations directly, the operating system must provide some means to perform I/O. Each program requires an input and produces output.

(4) File-system manipulation: There are many details in file creation, deletion, allocation, and naming that users should not have to per-form. Blocks of disk space are used by files and must be tracked. Deleting a file requires removing the name file information and freeing the allocated blocks. Protections must also be checked to assure proper file access.

(5) Communications: Message passing between systems requires messages to be turned into packets of information, sent to the net-work controller, transmitted across a communications medium, and reassembled by the destination system. Packet ordering and data correction must take place. Again, user programs might not coordinate access to the network device, or they might receive packets destined for other processes.

(6) Error detection: Error detection occurs at both the hardware and software levels. At the hardware level, all data transfers must be inspected to ensure that data have not been corrupted in transit.

(7) Accounting: We may want to keep track at which users use how much and what kind of computer resources. What was the login time for a particular user; is he working on the system right now, what is the process ID for the user, all such in formations we can manage using accounting service provided by many multiuser systems.

(8) Resource allocation: When there are multiple users or multiple jobs running at the same time. Resources must be allocated to each of them. Many different types of resources are managed by the operating system.

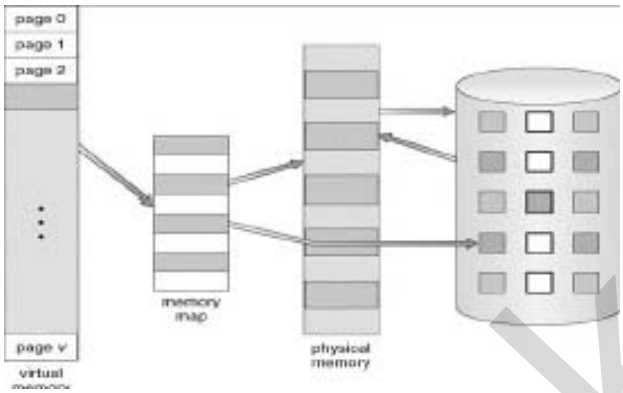
(9) Protection and security: The owners of information stored in multiuser or networked computer system may want to control use of the information. When several separate processes execute concurrently, it should not be possible for one process to interfere with the others or with the operating system itself, and protection involves ensuring that all access to system resources is controlled. Security of the system from outsiders is important.

Q.3(b) Describe concept of virtual memory with suitable example. [4]

(A) Virtual memory is the separation of user logical memory from physical memory.
This separation allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available.

Virtual memory makes the task of programming much easier, because the programmer no longer needs to worry about the amount of physical memory available, or about what code can be placed in overlays, but can concentrate instead on the problem to be programmed.
It is the process of increasing the apparent size of a computer's RAM by using a section of the hard disk storage as an extension of RAM.

As computers have RAM of capacity 64 or 128 MB to be used by the CPU resources which is not sufficient to run all applications that are used by most users in their expected way and all at once.

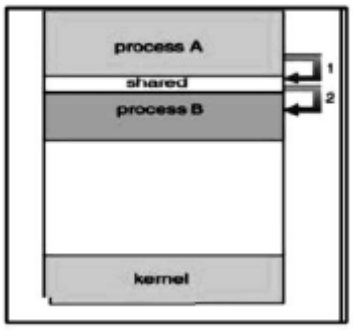


Example :
For example, an e-mail program, a web browser and a word processor is loaded into RAM simultaneously; the 64 MB space is not enough to store all these programs.
Without a virtual memory, a message "You cannot load any more applications. Please close an application to load a new one." would be displayed. By using a virtual memory, a computer can look for empty areas of RAM which is not being used currently and copies them on to the hard disk device. Thus RAM is freed to load new applications.
Actually it is done automatically, the user do not even know that it is happening, and the user feels like RAM has unlimited space even though the RAM capacity is 32 MB. It is a process of increasing computer's RAM by using a section of the hard disk storage as an extension of RAM.

Q.3(c) Explain interprocess communication. **[4]**

(A) **Inter-process communication** : Cooperating processes require an Inter-process communication (IPC) mechanism that will allow them to exchange data and information.

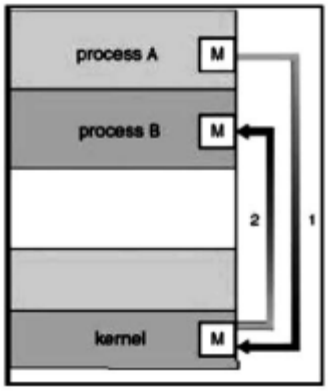
There are two models of IPC
(1) Shared memory



In this a region of the memory residing in an address space of a process creating a shared memory segment can be accessed by all processes who want to communicate with other processes. All the processes using the shared memory segment should attach to the address space of the shared memory. All the processes can exchange information by reading and/or writing data in shared memory segment. The form of data and location are determined by these processes who want to communicate with each other. These processes are not under the control of the operating system. The processes are also responsible for ensuring that

they are not writing to the same location simultaneously. After establishing shared memory segment, all accesses to the shared memory segment are treated as routine memory access and without assistance of kernel.

(2) Message Passing



In this model, communication takes place by exchanging messages between cooperating processes. It allows processes to communicate and synchronize their action without sharing the same address space. It is particularly useful in a distributed environment when communication process may reside on a different computer connected by a network Communication requires sending and receiving messages through the kernel. The process that want to communicate with each other must have a communication link between them. Between each pair of process exactly one communication link.

Q.3(d) Write steps for Banker's algorithm to avoid deadlock. [4]

(A) **Banker's Algorithm :** This algorithm calculates resources allocated, required and available before allocating resources to any process to avoid deadlock. It contains two matrices on a dynamic basis. Matrix A contains resources allocated to different processes at a given time. Matrix B maintains the resources which are still required by different processes at the same time.

Algorithm F : Free resources

Step 1 : When a process requests for a resource, the OS allocates it on a trial basis.

Step 2 : After trial allocation, the OS updates all the matrices and vectors. This updation can be done by the OS in a separate work area in the memory.

Step 3 : It compares F vector with each row of matrix B on a vector to vector basis.

Step 4 : If F is smaller than each of the row in Matrix B i.e. even if all free resources are allocated o any process in Matrix B and not a single process can completes its task then OS concludes that the system is in unstable state.

Step 5 : If F is greater than any row for a process in Matrix B the OS allocates all required resources for that process on a trial basis. It assumes that after completion of process, it will release all the recourses allocated to it. These resources can be added to the free vector.

Step 6 : After execution of a process, it removes the row indicting executed process from both matrices.

Step 7 : This algorithm will repeat the procedure step 3 for each process from the matrices and finds that all processes can complete execution without entering unsafe state. For each request for any resource by a process OS goes through all these trials of imaginary allocation and updation. After this if the system remains in the safe state, and then changes can be made in actual matrices.

Q.3(e) State necessary condition for deadlock. **[4]**

(A) **(1) Mutual Exclusion :** The resources involved are non-shareable.
At least one resource (thread) must be held in a non-shareable mode, that is, only one process at a time claims exclusive control of the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.

(2) Hold and Wait : Requesting process hold already, resources while waiting for requested resources. There must exist a process that is holding a resource already allocated to it while waiting for additional resource that are currently being held by other processes.

(3) No-Preemptive : Resource already allocated to a process cannot be preempted.
Resources cannot be removed from the processes are used to completion or released voluntarily by the process holding it.

(4) Circular Wait : The process in the system form a circular list or chain where each process in the list is waiting for a resource held by the next process in the list.

Q.3(f) List different types of files. Explain basic operations on file. **[4]**

(A)

File type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	Obj,o	compiled, machine language, not linked
source code	c,cc,java,pas, asm,a	Source code in various languagaes
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor Formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf,jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, r m, mp3, avi	binary file containing audio or A/V information

Common file types

File Operations

Basic file operations are

(1) Creating a file. To steps are necessary to create a file.
- Space in the file system must be found for the file.
- An entry for the new file must be made in the directory.

(2) Writing a file. To write a file, we make a system call specifying both the name of the file and the information to be written to the file. The system must keep a write pointer to the location in the file where the next write is to take place. The write pointer must be updated whenever a write occurs.

(3) Reading a file. To read from a file, we use a system call that specifies the name of the file and where (in memory) the next block of the file should be put. The system needs to keep a read pointer to the location in the file where the next read is to take place. Because a process is usually either reading from or writing to a file, the current operation location can be kept as a per-process current-file-position point. Both the read and write operations use this same pointer, saving space and reducing system complexity.

(4) Repositioning within a file. The directory is searched for the appropriate entry, and the current-file-position pointer is repositioned to a given value. Repositioning within a file need not involve any actual I/O. This file operation is also known as a file seek.

(5) Deleting a file. To delete a file, we search the directory for the named file. Having found the associated directory entry, we release all file space, so that it can be reused by the other files, and erase the directory entry.

The five operations described comprise only the minimal set of required file operations. More commonly, we shall also want to edit the file and modify its contents. A special case of editing a file is appending new information at the end of the file. Copies of the file can also be created, and since files are named object, renaming an existing file may also be needed. If the file is a binary object format, we may also want to execute it. Also of use are facilities to lock sections of an open file for multiprocess access, to share sections, and even to map sections into memory or virtual memory systems.

This last function allows a part of the virtual address to be logically associated with section of a file. Reads and writes to that memory region are then treated as reads and writes to the file. To that memory region are then treated as reads and writes to the file, greatly simplifying file use.

(6) Truncating a file. The user may want to erase the contents of a file but keep its attributes. Rather than forcing the user to delete the file and then recreate it, this function allows all attributes to remain unchanged (except for file length) but lets the file be reset to length zero and its file space released.

Q.4 Attempt any THREE of the following : [12]

Q.4(a) What is system call? List types of system call with one example of system call. [4]

(A) System Calls : System calls are programming interface to the services provided by the operating system.

Types of system calls:

(i) Process or Job control
(ii) File Management
(iii) Device Management
(iv) Information Maintenance

(1) System calls related to process control : End, Abort Load, Execute Create process, Terminate process Ready process, Dispatch process Suspend, Resume Get Process attribute, set attribute Wait for time Wait event, signal event.

(2) System calls Related to File management : Create file, delete file Open file, Close file Create directory Read, Write, Reposition Get file attribute, set file attribute Crete a link Change the working directory.

(3) System calls Related to Device Management : Request a device, Release a s device Read, Write, Reposition Get device attribute, set device attribute.

(4) System calls Related to Information Maintenance : Get Time or Date, Set Time or Date Get System data, Set system data Get process, file or device attributes Set process, file or Device attributes.

Q.4(b) Describe any four secondary storage management activities. [4]

(A) Secondary Storage Management : System have several levels of storage, including primary storage, secondary storage and cache storage. Instructions and data must be placed in primary storage or cache to be referenced by a running program. Because main memory is too small to accommodate all data and programs, and its data are lost when power is lost, the computer system must provide secondary storage to back up main memory.

Secondary storage consists of tapes, disks, and other media designed to hold information that will eventually be accessed in primary storage (primary, secondary, cache) is ordinarily divided into bytes or words consisting of a fixed number of bytes.

The four major activities of an operating system in regard to secondary storage management are :

(1) Managing the free space available on the secondary-storage device.
(2) Allocation of storage space when new files have to be written.
(3) Scheduling the requests for memory access.
(4) Deallocation of storage space when required.

Q.4(c) What is thread? Explain users and Kernel threads. [4]

(A) A thread, sometimes called a lightweight process, is a basic unit of CPU utilization. A traditional (or heavyweight) process has a single thread of control. If a process has multiple threads of control, it can do more than one task at a time. This is because there are situations in which it is desirable to have multiple threads of control in the same address space, running as though they were separate processes.

User-Level Threads :

- A user-level thread is a thread within a process which the OS does not know about.
- In a user-level thread approach the cost of a context switch between threads less since the operating system itself does not need to be involved-no extra system calls are required.
- A user-level thread is represented by a program counter, registers, stack, and small thread control block (TCB).
- Programmers typically use a thread library to simplify management of threads within a process.
- Creating a new thread, switching between threads, and synchronizing threads are done via function calls into the library. This provides an interface for creating and stopping threads, as well as control over how they are scheduled.

Kernel Threads :

- In system that use kernel-level threads, the operating system itself is aware of each individual thread.
- Kernel threads are supported and managed directly by the operating system.
- A context switch between kernel threads belonging to the same process requires only the registers, program counter, and stack to be changed; the overall memory management information does not need to be switched since both of the threads share the same address space. Thus context switching between two kernel threads is slightly faster than switching between two processes.
- Kernel threads can be expensive because system calls are required to switch between threads. Also, since the operating system is responsible for scheduling the threads, the application does not have any control over how its threads area managed.

Q.4(d) State and describe types of schedules. Describe how each of them schedule the job. [4]

(A) Schedulers are of three types :

- Long Term Scheduler
- Short Term Scheduler
- Medium Term Scheduler

Long Term Scheduler : It is also called job scheduler. Long term scheduler determines which programs are admitted to the system for processing. Job scheduler selects processes form the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling. The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the

system. On some system, the long term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When process changes the state from new to ready, then there is use of long term scheduler.

Short Term Scheduler : It is also called CPU scheduler. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects process among the processes that are ready to execute most frequently and makes the fine grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

Medium Term Scheduler : Medium term scheduling is part of the swapping. It removes the processes from the memory. If reduces the degree of multiprogramming. The medium term scheduler is in-change of handling the swapped out-processes. Running process may become suspended if it makes an I/O request. Suspended processes cannot make any progress towards completion. In this condition, to remove the process from memory and make space for other process, the suspended process is moved to the secondary storage. This process is called swapping, and the process is said to be swapped out or roller out. Swapping may be necessary to improve the process mix.

5. Attempt any TWO of the following : [16]

(a) Solve the following problem using SJF and Round Robin (RR) scheduling algorithm. Find average waiting time for each algorithm.

Process	Burst time
P_1	10
P_2	3
P_3	7
P_4	5

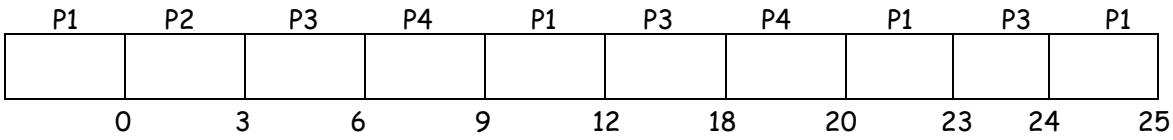
(A) SJF

Gantt chart

Waiting time
P1 15 mesc
P2 0 mesc
P3 8 mesc
P4 3 msec

Average waiting time = waiting time of all processes/number of processes
= waiting time of (P1 + P2 + P3 + P4)/4
= 15 + 0 + 8 + 3/4 = 26/4 = 6.5 mesc

Round Robin:
Gantt chart



Waiting time
P1 = 0 + (12 – 3) + (20 – 15) + (24 – 23) = 0 + 9 + 5 + 15msec
P2 = 3msec

P3 = 6 + (15 – 9) + (23 – 18) = 6 + 6 + 5 = 17msec
P4 = 9 + (18 – 12) = 9 + 6 = 15 msec
Average waiting time = waiting time of all processes/number of processes
= waiting time of (P1 + P2 + P3 + P4)/4
= 15 + 3 + 17 + 15/4 = 50/4 = 12.5 msec

(b) Explain how priority scheduling algorithm works with suitable example, also list advantages and disadvantages.

(A) Priority scheduling algorithm : In priority scheduling algorithm, Number (integer) indicting priority is associated with each process. The CPU is allocated to a process with the highest priority. A priority algorithm will preempt the CPU if the priority of the newly arrived process is higher than the priority of the currently running process.

A major problem with priority scheduling is indefinite blocking or starvation. A solution to the problem of indefinite blockage of the low-priority process is aging. Aging is a technique of gradually increasing the priority of processes that wait in the system for a long period of time.

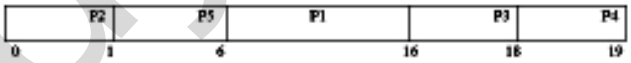
Advantage :
Priority Scheduling
- Simplicity
- Reasonable support for priority
- Suitable for applications with varying time and resource requirements.

Disadvantages of Priority Scheduling
- Indefinite blocking or starvation.
- A priority scheduling can leave some low priority processes waiting indefinitely for CPU.

Example :

PROCESS	BURST TIME	PRIORITY
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

Gantt chart

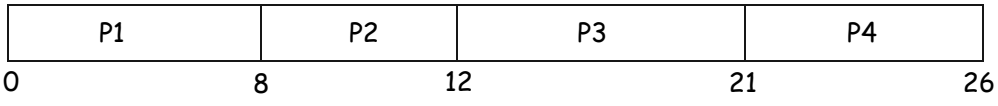


Waiting time for each process : P1 = 6, P2 = 0, P3 = 16, P4 = 18, P5 = 1
Average waiting time : = (6 + 0 + 16 + 18 + 1)/5 = 41/5 = 8.2 milliseconds

(c) Calculate average waiting time for FCFS and SJF for following table.

Process	Arrival Time	Burst Time
P1	0	8
P2	1	4
P3	2	9
P4	3	5

(A) FCFS
Gantt chart

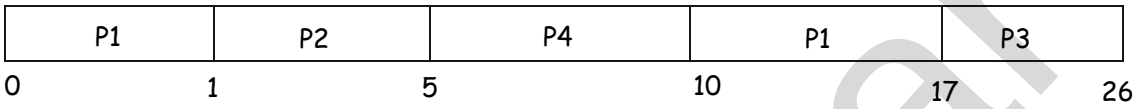


Waiting time

P1 = 0
P2 = (8 – 1) = 7
P3 = (12 – 2) = 10
P4 = (21 – 3) = 18

Average waiting time = waiting time of all processes/number of processes
= waiting time of (p1 + p2 + p3 + p4)/4
= 0 + 7 + 10 + 18/4
= 8.75 milli seconds (ms)

SJF



Waiting time

P1 = 0 + (10 – 1) = 9
P2 = 0
P3 = (17 – 2) = 15
P4 = (5 – 3) = 2

Average waiting time = waiting time of all processes/number of processes
= waiting time of (p1 + p2 + p3 + p4)/4
= 9 + 0 + 15 + 2/4
= 6.5 milli seconds (ms)

Q.6 Attempt any FOUR of the following : [16]

Q.6(a) What is multiprocessor system? Give two advantages of it. [4]

(A) Multiprocessor Systems: Multiprocessor systems with more than on CPU in close communication.
Tightly coupled system - processors share memory and a clock; communication usually takes place through the shared memory.

Advantages of multiprocessor system:

- Less time duration required for the large process.
- Increase throughput.

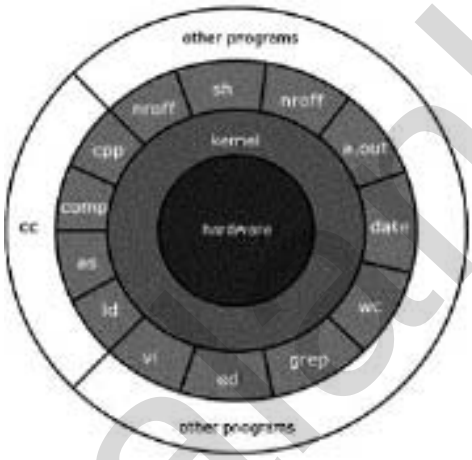
Q.6(b) Draw and explain structure of unix operating system. [4]

(A) The kernel of UNIX is the hub of the operating system: it allocates time and memory to programs nd handles the file store and communications in response to system calls. As an illustration of the way that the shell and the kernel work together, suppose a user types rm my file (which has the effect of removing the file my file). The shell searches the file store for the file containing the program rm, and then requests the kernel, through system calls, to execute the program rm on my file. When the process rm my file has finished running, the shell then returns the UNIX prompt % to the user, indicating that it is waiting for further commands.

Amongst the functions performed by the kernel are:

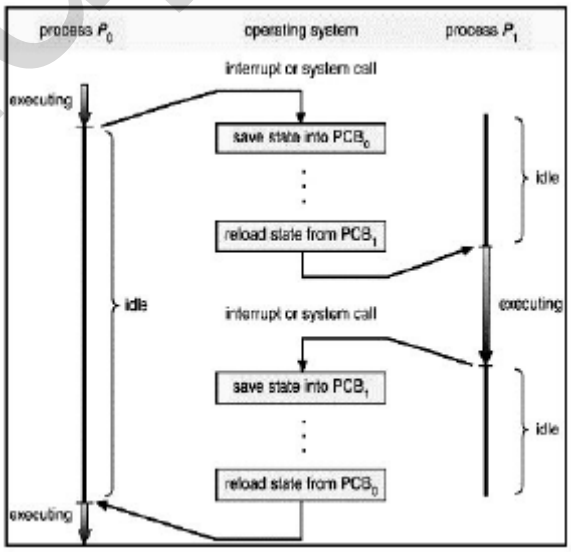
- Managing the machine's memory and allocating it to each process.
- Scheduling the work done by the CPU so that the work of each user is carried out as efficiently as is possible.
- Organizing the transfer of data from one part of the machine to another.

- Accepting instructions from the shell and carrying them out.
- Enforcing the access permissions that are in force on the file system the shell: The shell acts as an interface between the user and the kernel. When a user logs in, the login program checks the username and password, and then starts another program called the shell. The shell is a command line interpreter (CLI). It interprets the commands the user types in and arranges for them to be carried out. The commands are themselves programs: when they terminate, the shell gives the user another prompt (%on our systems). The user can customize his/her own shell, and users can use different shells on the same machine. The shell keeps a list of the commands you have typed in. If you need to repeat a command, use the cursor keys to scroll up and down the list or type history for a list of previous commands. You can use any one of these shells if they are available on your system. And you can switch between the different shells once you have found out if they are available.
- Bourne shell (sh)
- C shell (csh)
- TC shell (tcsh)
- Korn shell (ksh)



Q.6(c) Explain the concept of context switching. [4]

(A) Switching the CPU to another process requires saving the state of current process and loading the saved state for new process. This process is known as a context switch. The context switch is represented in PCB. Saves context of old process in its PCB and loads context of new process into the memory which is schedule to run next.



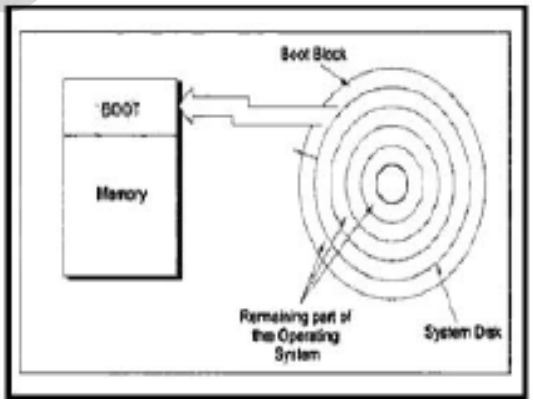
Q.6(d) Differentiate between paging and segmentation. **[4]**

(A)

	Paging	Segmentation
(i)	Paging divides the computer's primary memory into fixed-size units called page frames, and the program's address space into pages of the same size.	Segmentation is the only memory management technique that does not provide the user's program with a "linear and contiguous address space."
(ii)	The hardware memory management unit maps pages to frames.	Segments are areas of memory that usually correspond to a logical grouping of information such as a code procedure or a data array.
(iii)	The physical memory can be allocated on a page basis while the address space appears contiguous.	Segments require hardware support in the form of a segment table which usually contains the physical address of the segment in memory, its size, and other data such as access protection bits and status.
(iv)	Pages are used for swapping or managing memory.	Small pieces called segments are used for memory management.
(v)	Page is indicated by its number and offset.	Segment is indicted by segment number and its offset
(vi)	Page table is formed.	Segment table is formed.
(vii)	Do not support user's view of memory.	Supports user's view of memory.

Q.6(e) Explain booting system of UNIX **[4]**

(A) The loading of the operating system is achieved by a special program called BOOT. Generally portion is normally called "BOOT Block" as shown in figure. The ROM normally contains a minimum program. When one tums the computer "ON", the control is transferred to this program automatically by the hardware itself. This program in ROM loads the BOOT program in pre-determined memory locations. The beauty is to keep BOOT program as small as possible, so that the hardware can manage to load it easily and in a very few instructions. This BOOT program in turn contains to read the rest of the Operating System into the memory. This is depicted in figures. The mechanism gives an impression of pulling oneself up. Therefore, the nomenclature bootstrapping or its short form booting.



❑ ❑ ❑ ❑ ❑