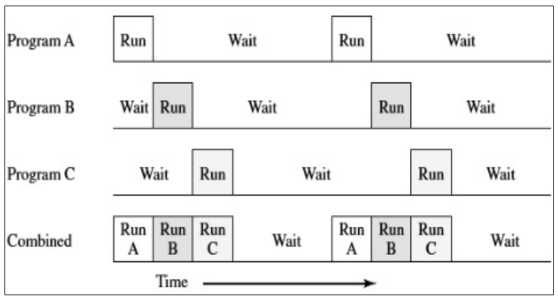
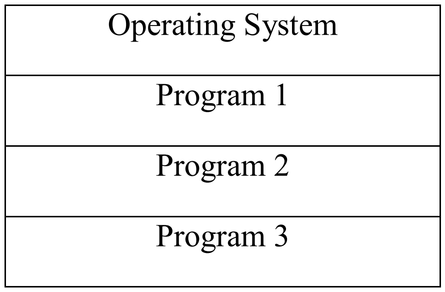
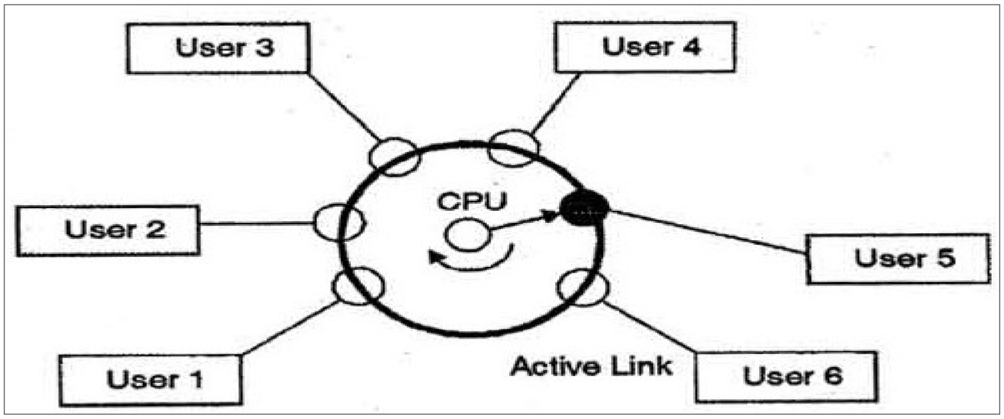
WINTER– 17 EXAMINATION

Subject Name: Operating System **Model Answer** **Subject Code: 17512**

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1.	a)	**Attempt any THREE of the following:**	**12 Marks**
	i)	**Describe multiprogramming and multitasking.**	**4M**
	Ans:	**Multiprogramming:** In multiprogramming, more than one program lies in the memory. The scheduler selects the jobs to be placed in ready queue from a number of programs. The ready queue is placed in memory and the existence of more than one program in main memory is known as multiprogramming. Since there is only one processor, there multiple programs cannot be executed at a time. Instead the operating system executes part of one program, then the part of another and so on. Example of multiprogramming: user can open word, excel, access and other applications in a system.  Program A Run Wait Run Wait Program B Wait Run Wait Run Wait Program C Wait Run Wait Run Wait Combined Run A Run B Run C Wait Run A Run B Run C Wait Time OR  Fig- Multiprogramming with three programs	**(Description of multiprogramming:2 marks, multitasking :2 marks)**

		Multitasking: A multitasking operating system is any type of system that is capable of running more than one task at a time. It also maintains the synchronization between I/O devices and processes so that user can use different application in background and current application in foreground. In multitasking the resources are made continuously working. The CPU switches from one task to another for reading and processing. Thus idle time of peripherals gets reduced. In multitasking Operating System the code as well as data of several processes is stored into main memory. For example, when you are printing a document of 100 pages, you can do other jobs like typing a new document. So, more than one task is performed.	
	ii)	**Explain time sharing operating system and state its advantages and disadvantages.**	**4M**
	Ans:	The main idea of time sharing systems is to allow a **large number of users** to interact with a **single computer system** concurrently. In time sharing system, the CPU executes multiple jobs by switching among them. The switches occur so frequently that the users can interact with each program while it is running. A **time sharing system** allows many users to share the computer resources simultaneously. Since each action or command in a time shared system tends to be short, only a little CPU time is needed for each user. As the system switches rapidly from one user to the next, each user is given the impression that the entire computer system is dedicated to his use, even though it is being shared among many users. The time sharing systems were developed to provide an interactive use of the computer system. Most time sharing systems use time-slicing (Round Robin) scheduling. A program executing longer than the system defined time slice is interrupted by the operating system and place at the end of the queue of waiting programs for execution. **Example:**  User 3 User 4 CPU User 2 User 5 User 1 Active Link User 6	**(Explanation :2 marks , one advantage: 1 mark, one disadvantage :1mark)**

In above figure the user 5 is active but user 1, user 2, user 3, and user 4 are in waiting state whereas user 6 is in ready status. As soon as the time slice of user 5 is completed, the control moves on to the next ready user i.e. user 6. In this state user 2, user 3, user 4, and user 5 are in waiting state and user 1 is in ready state. The process continues in the same way and so on.

Advantages of Time Sharing System

- Each user can get CPU time.
- Efficient CPU utilization.
- Time sharing systems were developed to provide interactive use of a computer system at a reasonable cost.
- A time shared operating system uses CPU scheduling and multi programming to provide each user with a small portion of a time-Shared Computer.

Disadvantages of Time Sharing System:

- The time-shared systems are more complex than the multi-programming systems.
- Context switching occurs frequently**. i.e.** Multiple processes are managed simultaneously which requires an adequate management of main memory so that the processes can be swapped in or swapped out within a short time.

iii) **What is system call? Enlist any four system calls related with process management.** **4M**

Ans: System call is an interface between a running program and operating system. It allows user to access services provided by operating system. This system calls are procedures written using C, C++ and assembly language instructions. Each operating system has its own name for each system call. Each system call is associated with a number that identifies itself.

(Description of system call:2 marks, any four system calls:½ mark each)

System calls related with process management:

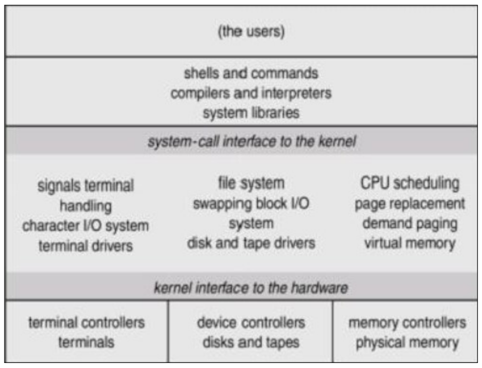
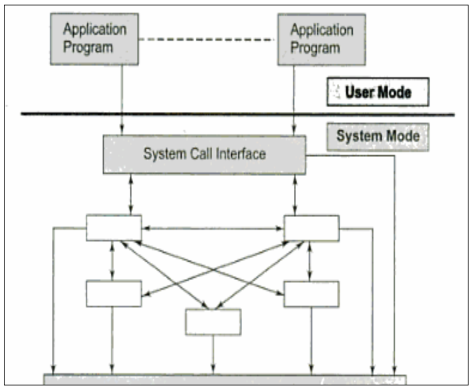
- end, abort
- load, execute
- create process, terminate process
- get process attributes, set process attributes
- wait for time
- wait event, signal event
- Allocate and free memory.

iv) **List and explain any four attributes of file.** **4M**

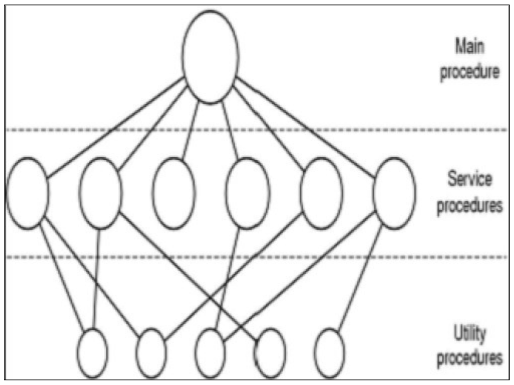
Ans: **File attributes:**

(Any four attributes: 1mark each)

- **Name:** The symbolic file name is the only information kept in human readable form.
- **Identifier:** File system gives a unique tag or number that identifies file within file system and which is used to refer files internally.
- **Type:** This information is needed for those systems that support different types.
- **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 that who can do reading, writing, executing and so on.
- **Time, Date and User Identification:** This information may be kept for creation, Last

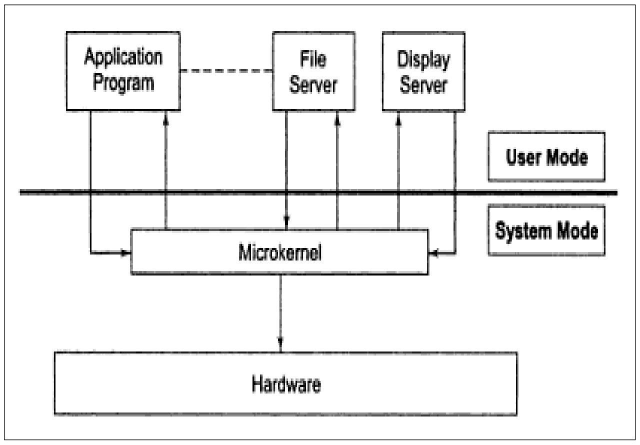
		modification and last use. These data can be useful for protection, security and usage monitoring.	
	b)	**Attempt any ONE of the following:**	**6 Marks**
	i)	**Explain following operating system structure in details:** **1) Monolithic** **2) Microkernel**	**6M**
	Ans:	**Monolithic System:** System is divided into multiple modules. Each module is designed for performing specific task such as file management, I/O management or memory management and so on. Any module can call any other module without any major restrictions. Operating system distinguishes between system mode and user mode while executing an application program. An application program runs in the user mode. A user makes a request for a service using application programs. Application programs request for a system call to system call interface. The operating system locates a system call and executes it in the system mode. Once execution of system call is over, the execution of the application programs resumes in the user mode. **{**Note: Any one diagram from the following**}**  (the users) shells and commands compilers and interpreters system libraries system-call interface to the kernel signals terminal handling character I/O system terminal drivers file system swapping block I/O system disk and tape drivers CPU scheduling page replacement demand paging virtual memory kernel interface to the hardware terminal controllers terminals device controllers disks and tapes memory controllers physical memory OR  Application Program Application Program User Mode System Mode System Call Interface	***(*Monolithic structure Diagram: 1 mark, Explanation: 2 marks ,Microkernel structure: Diagram: 1 mark, Explanation: 2 marks)**

OR

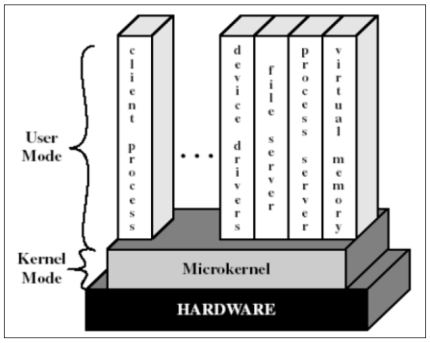


Microkernel:
In this system, kernel provides only the most essential operating system functions like process management, communication primitives and low level memory management. System programs and user level programs implemented outside the kernel, provides the remaining operating system services. These programs are known as servers. Due to separation of functionality of kernel, size of the kernel is reduced. This reduced kernel is called as microkernel. The application programs and various servers communicate with each other using messages that passed through microkernel. The microkernel validates the messages and passes them between the various modules of the operating system and permits access to the hardware.

{Note: Any one diagram from the following**}**



OR



ii) **Explain any six file operations performed by OS.** **6M**

Ans:

(Any six operations: 1mark each)

- **Creating a file:** Two steps are necessary to create a file. First space in the file system must be found for the file. Second an entry for the new file must be made in the directory. The directory entry records the name of the file and the location in the file system.

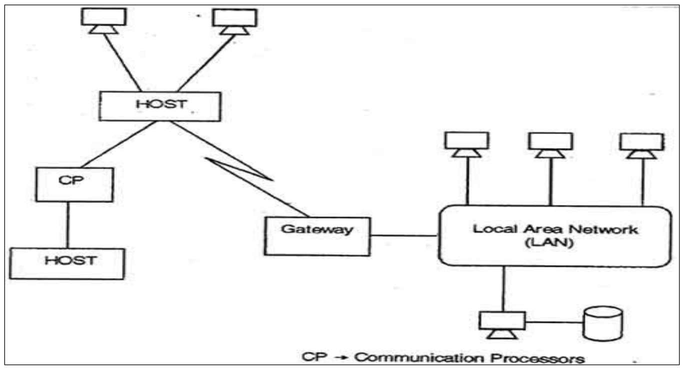
- **Writing a file:** To write a file, we make a system call specifying both the name of the file and the information to be written to the file. Given the name of the file, the system searches the directory to find the location of file then the write pointer must be updated whenever a write occurs

- **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. System needs to keep a read pointer to location in the file where the next read is to take place. Once the read has taken place, the read pointer is updated.

- **Repositioning within a file:** The directory is searched for the appropriate entry, and the current file position is set to a given value. Repositioning within a file does not need to involve any actual I/O. This file operation is also known as a file seeks.
- **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 and erase the directory entry.

- **Truncating a file:** Instead of deleting a file and then recreate it, this function allows all attributes to remain unchanged but for the file to be reset to length zero. User wants to erase the contents of the file.

Other common operations include appending new information to the end of an existing file, and renaming an existing file.

2.		**Attempt any FOUR of the following:**	**16 Marks**
	a)	**Explain distributed system in detail.**	**4M**
	Ans:	{**Note: Diagram is optional**} • A **distributed system** consists of a collection of autonomous computers, connected through a network and distribution middleware, which enables computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facility. • In such system the processors do not share memory or a clock; instead each processor has its own local memory. In such systems, if one machine or site fails the remaining sites can continue operation. So these types of systems are the reliable systems. The processors communicate with one another through various communications lines, such as a high speed buses or telephone lines. These systems are usually referred to as **Loosely Coupled Systems or Distributed Systems.**  HOST CP HOST Gateway Local Area Network (LAN) CP → Communication Processors **Fig. Distributed Systems** The structure shown in figure contains a set of individual computer systems and workstations connected via communication systems. By this structure we cannot say it is a distributed system because it is the software, not the hardware, that determines whether a system is distributed or not. The users of a true distributed system should not know, on which machine their programs are running and where their files are stored.	**(Explanation: 4 marks)**

b) **Differentiate between short term and long term scheduler.** **4M**

Ans: **{**Note: Any other relevant difference shall be considered**}**

(Any four relevant Points: 1 mark Each)

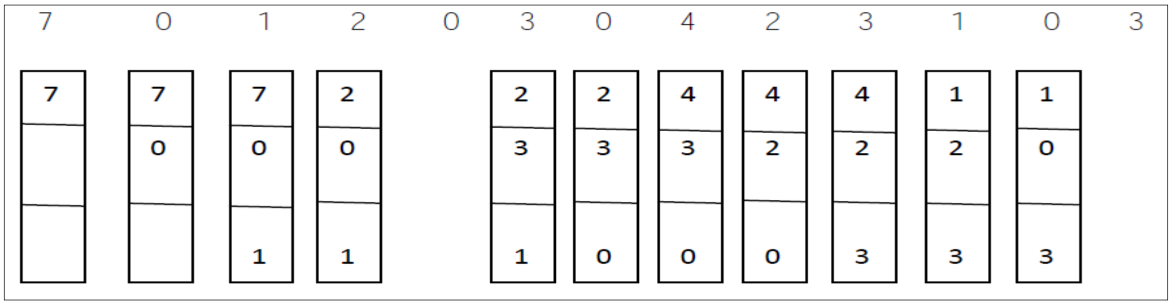
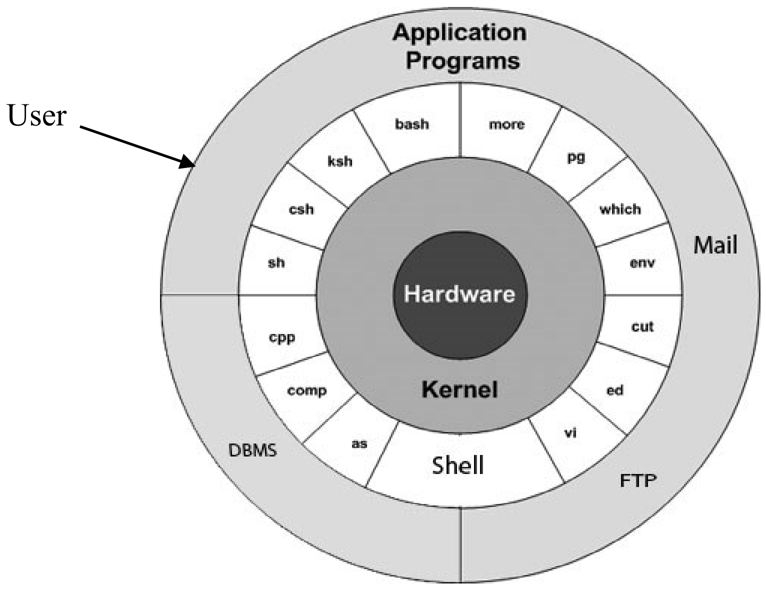
Sr. No	Short term scheduler	Long term scheduler
1	It is a CPU scheduler	It is a job scheduler
2	It selects processes from ready queue which are ready to execute and allocates CPU to one of them.	It selects processes from job pool and loads them into memory for execution.
3	Access ready queue and CPU.	Access job pool and ready queue
4	It executes frequently. It executes when CPU is available for allocation.	It executes much less frequently. It executes when memory has space to accommodate new process.
5	Speed is fast	Speed is less than short term scheduler
6	It provides lesser control over degree of multiprogramming	It controls the degree of multiprogramming

c) **State and explain criteria used for CPU scheduling.** **4M**

Ans:

(Any four criteria: 1 mark each)

- **CPU Utilization:** We want to keep to CPU as busy as possible; CPU Utilization may range from 0 to 100 percent. In a real system it should range from 40 percent to 90 percent.
- **Throughput:** If the CPU is busy executing processes, then work is being done. One measure of work is the number of processes that are completed per time unit called Throughput. For long processes, this rate may be one process per hour, for short transactions, throughput might be 10 processes per second.
- **Turnaround Time:** From the point of view of a particular process, the important criterion is how long it takes to execute that process. The interval from the time of submission of a process to the time of completion is the turnaround time. Turnaround Time is the sum; of the periods spent waiting to get into memory writing in the ready queue, executing on the CPU and doing I/O.
- **Waiting Time:** The CPU scheduling algorithm does not affect the amount of time during which a process executes of does I/O, if affects only the amount of time that a process spends waiting in the ready queue. Waiting time is the sum of the periods spent waiting in the ready queue.
- **Response Time:** A process may produce some output early and continue computing new results while previous results are being output to the user. Thus another measure is the time from submission of a request until the first response is produced. This measure called **response time** is the amount of time it takes to start responding, but not the time that it takes to output that response. The Turnaround Time is generally limited by the speed of the output device.

	d)	**Explain FIFO (First in First out) page replacement algorithm for reference string 7012030423103.**	**4M**
	Ans:	First-In-First-Out (FIFO) Algorithm: A FIFO replacement associates with each page the time when that page was bought into memory. When the page must be replaced, the oldest page is chosen. It maintains a FIFO queue to hold all pages in memory. We replace the page at the head of the queue. When a page is brought into the memory, we insert it at the tail of the queue. Reference string: Consider three frames are available.  7 0 1 2 0 3 0 4 2 3 1 0 3 7 7 0 7 0 1 2 0 1 2 3 1 2 3 0 4 3 0 4 2 0 4 2 3 1 2 3 1 0 3 **Page fault 11** **Page hit 2**	**(Explanation: 4 marks)**
	e)	**Explain structure of unix operating system with the help of diagram.**	**4M**
	Ans:	**{**Note: Any other relevant diagram shall be considered**}** **Architecture of UNIX:**  User Application Programs bash more pg which env cut ed vi Shell as comp cpp sh csh ksh Hardware Kernel Mail FTP DBMS	**(Diagram: 2 marks, Explanation: 2 marks)**

Hardware: The hardware is Centre of structure that provides the Operating System with basic services. The hardware consists of all peripherals like memory (RAM, HDD, FDD etc) processor, mouse, and other input devices, terminals, printers etc.

The Kernel: The kernel is the heart of the system - a collection of programs mostly written in 'C' which communicate with the hardware directly. Kernel is an interface between hardware of the system and shell. It is loaded into the memory when the system is booted. User programs that need to communicate with the hardware use the services of the kernel, which performs the job on the user's behalf. It manages the system's memory, schedules processes, decides their priorities and performs other tasks.

Shell: The shell is an interface between the user and the kernel that isolates the user from knowledge of kernel functions. The shell accepts the commands keyed by the users and checks for their syntax and gives out error messages if something goes wrong. It is a command interpreter of user requests.

Application programs: The various compilers for languages like c, c++, pascal, fortran and other application programs written by programmers which are used by users for their operations falls in this layers. Only those persons who maintain on "account" with the computer system can use the UNIX system.

User can directly access application programs through which they can interact with the system.

f) **Compare UNIX and LINUX operating system w.r.t.** **4M**

i) **User interface**
ii) **Name of provider**
iii) **Processing speed**
iv) **Security**

Ans: **(Each Difference: 1 Mark)**

Parameter	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, etc..	Initially Unix was a command based OS, but later a GUI was created called Common Desktop Environment. Most distributions now ship with Gnome.
Name of Provider	Redhat, Ubantu, Fedora	Osx, Solaris, All LINUX
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
Security	Linux has had about 60-100 viruses listed till date. None of them actively is spreading nowadays.	A rough estimate of UNIX viruses is between 85 -120 viruses reported till date.

3.		**Attempt any <u>FOUR</u> of the following :**	**16 Marks**
	a)	**List any four services provided by OS and explain any two of them.**	**4M**
	Ans:	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** **OS services provided to the user:-** **1. User Interface**: - All operating systems have a user interface that allows users to Communicate with the system. Three types of user interfaces are available:- **a.** Command line interface (CLI): - It uses text commands and a method for entering them. For example working on DOS prompt. **b.** Batch interface: - Commands and directives to control that commands, are entered into a file and the file is executed. For example, Combining set of C programming statements into a file to perform a specific task and executing that file in TURBO C. **c.** Graphical user interface (GUI): - This interface is a window system with a pointing device to direct I/O, select from menus and make selections and keyboard to enter the text. For example, Windows system provides icons for selecting an application. Double clicking on that icon will open that application. Some system provides to or all three of these variations. **2. Program execution**: - The operating system provides an environment where the user can conveniently run programs. To run a program, the program is loaded into the main memory and then <u>CPU</u> is assigned to that process for its execution. Operating system performs this function for the convenience of the user. It also performs other important tasks like allocation and de-allocation of memory, CPU scheduling etc. It also provides service to end process execution either normally or abnormally by indicating error. **3. I/O operations**: - When a program is running, it may require input/output resources such as a file or devices such as printer. For specific devices, special functions may be required such as recording to a CD drive. For efficiency and protection users usually cannot control I/O devices directly. So the operating system provides a service to do I/O.	**(List of any four services: 1 mark ; Explanation of any two services: 1 ½ marks each)**

4. **File system manipulation: -** Programs may need to read and write data from and to the files and directories. Operating system manages the secondary storage. User gives a command for reading or writing to a file. Operating system makes it easier for user programs to accomplish their task such as opening a file, saving a file and deleting a file from the storage disk. It also provides services for file permission management to allow or deny access to files or directories based on file ownership.

5. **Communication: -** In the system, one process may need to exchange information with another process. Such communication may occur between processes that are executing on different computer systems tied together by a computer network. Communication can be implemented via shared memory or through message passing, in which packets of information are moved between processes by the operating system.

6. **Error detection: -**The operating system needs to be constantly aware of possible errors.
   Errors can occur in:

a) CPU and memory hardware such as a memory error or power failure

b) I/O devices such as parity error on tape, a connection failure on a network or lack of paper in the printer.

c) The user program such as an arithmetic overflow, an attempt to access an illegal memory location or a too-great use of CPU time.
For each type of error, the operating system should take the appropriate action to ensure correct and consistent computing. Debugging facilities can greatly enhance the user's and programmer's abilities to use the system efficiently.

OS services provided to the system:-

a. Resource allocation: - When there are multiple users or multiple processes running at the same time, resources must be allocated to each of them. Operating system manages resource allocation to the processes. These resources are CPU, main memory, file storage and I/O devices. For maximizing use of CPU, operating system does CPU scheduling. Operating system contains routines to allocate printers, modems, USB storage drives and other peripheral devices.

b. Accounting: - Operating system keeps track of usages of various computer resources allocated to users. This accounting is used for reconfiguration of system to improve computing services.

a. **Protection & security**:-Owners of information stored in a multiuser or networked computer system want to control use of that information. When several separate processes execute concurrently, one process should not interfere with the other processes or operating system itself. Protection provides controlled access to system resources. Security is provided by user authentication such as password for accessing information.

	b)	**State and explain different process state.**	**4M**
	Ans:	Different process states are as follows: **a) New** **b) Ready** **c) Running** **d) Waiting** **e) Terminated** **New:** When a process enters into the system, it is in new state. In this state a process is created. In new state the process is in job pool. **Ready State:** When the process is loaded into the main memory, it is ready for execution. In this state the process is waiting for processor allocation. **Running State:** When CPU is available, system selects one process from main memory and executes all the instruction from that process. So when a process is in execution, it is in running state. In single user system, only one process can be in the running state. In multiuser system, there can be multiple processes which are in the running state. **Waiting State:** When a process is in execution, it may request for I/O resources. If the resource is not available, process goes into the waiting state. When the resource is available, the process goes back to ready state. **Terminated State:** When the process completes its execution, it goes into the terminated state. In this state the memory occupied by the process is released.	**(Stating:1 mark,Explanation:3 marks)**
	c)	**Describe the terms:** **i) Preemptive scheduling** **ii) Non preemptive scheduling.**	**4M**
	Ans:	**Preemptive Scheduling** 1. 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. 2. It is suitable for RTS. 3. Only the processes having higher priority are scheduled. 4. It doesn't treat all processes as equal. 5. Algorithm design is complex. 6. Circumstances for preemptive • process switch from running to ready state • process switch from waiting to ready State **Example:** Round Robin, Priority algorithms, SJF (Preemptive)	**(Description of Preemptive: 2 marks, Description of non-Preemptive: 2 marks)**

Non Preemptive Scheduling

1. 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.
2. It is not suitable for RTS.
3. Processes having any priority can get scheduled.
4. It treats all process as equal.
5. Algorithm design is simple.
6. Circumstances for Non preemptive

- Process switches from running to waiting state
- Process terminates

Example: FCFS algorithm,SJF (Non preemptive)

d) **Explain Round Robin algorithm with suitable example.** **4M**

Ans: Round –Robin Scheduling:

(Explanation of Round Robin algorithm:2 marks; Example:2 marks)

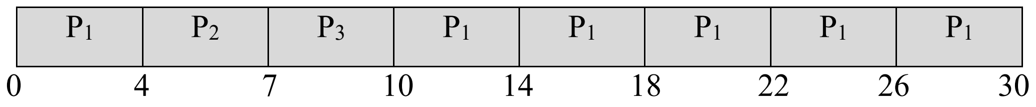
The Round–Robin (RR) scheduling algorithm is designed especially for time sharing systems. It is similar to FCFS scheduling, but preemption is added to enable the system to switch between processes. A small unit of time, called as time quantum or time slice, is defined. A time quantum is generally from 10 to 100 milliseconds in length. The ready queue is treated as a circular queue. The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum. To implement RR scheduling, we keep the ready queue as a FIFO queue of processes. New processes are added to the tail of the ready queue. The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatches the process. One of two things will then happen. The process may have a CPU burst of less than 1 time quantum. In this case, the process itself will release the CPU voluntarily. The scheduler will then process to the next process in the ready queue. Otherwise, if the CPU burst of the currently running process is longer than 1 time quantum, the timer will go off and will cause an interrupt to the operating system. A context switch will be executed, and the process will be put at the tail of the ready queue. The CPU scheduler will then select the next process in the ready queue. The average waiting time under the RR policy is often long. Consider the following set of processes that arrive at time 0, with the length of the CPU burst given in milliseconds:

Example:

Process	Burst Time
P_1	24
P_2	3
P_3	3

If we use a time quantum of 4 milliseconds, then process P_1 gets the first 4 milliseconds. Since it requires another 20 milliseconds, it is preempted after the first time quantum. And the CPU is given to the next process in the queue, process P_2. process P_2 does not need 4 milliseconds, so it quits before its time quantum expires. The CPU is then given to the next process, process P_3. Once each process has received 1 time quantum, the CPU is returns to process P_1 for an additional time quantum.

The resulting RR schedule is as follows:



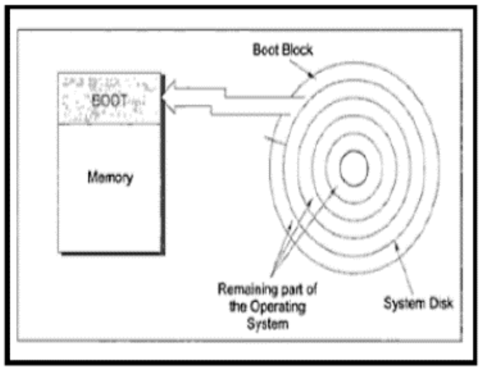
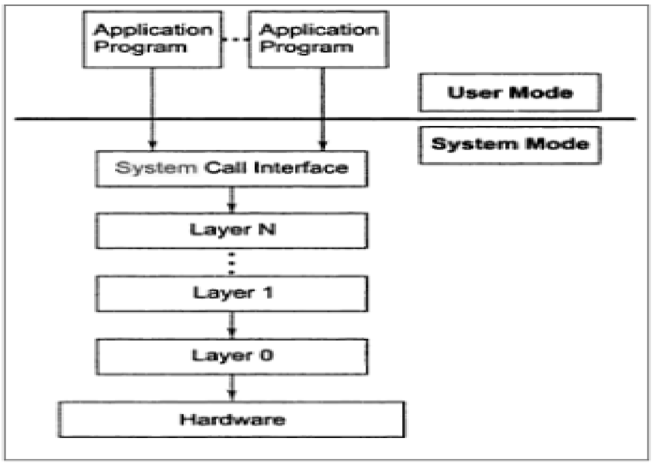
e) **Compare paging and segmentation memory management techniques.** **4M**

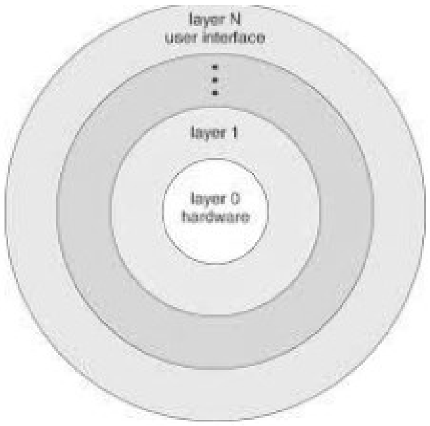
Ans:

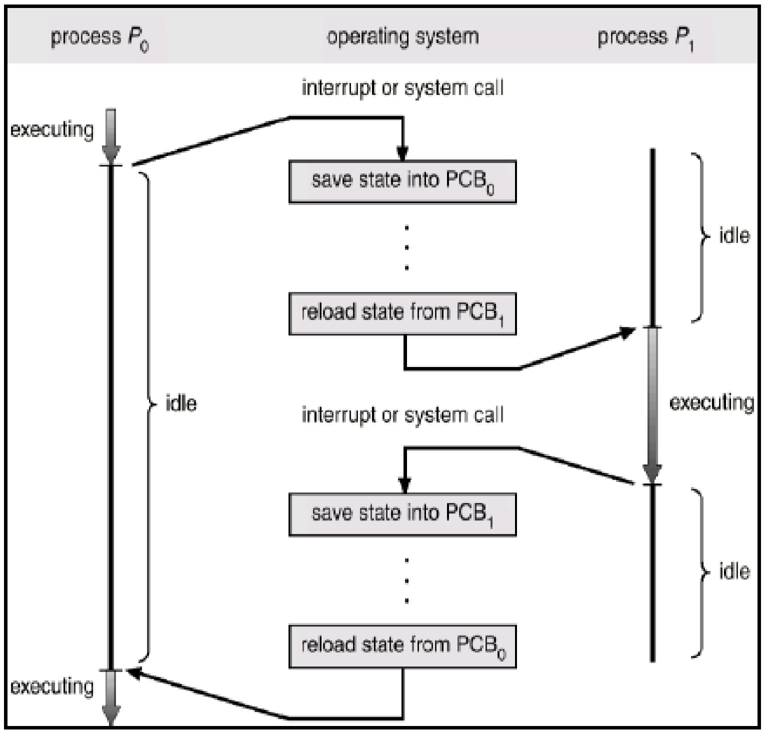
{Note: Any other valid point shall be considered**}**

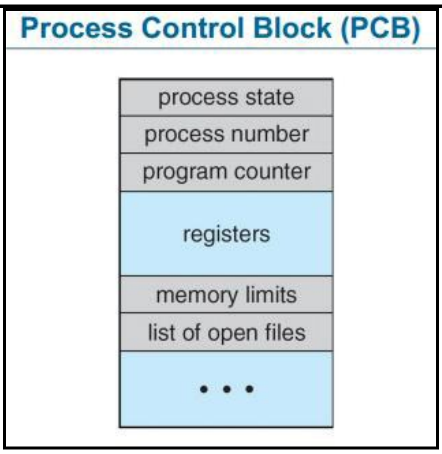
(Any 4 points: 1 mark each)

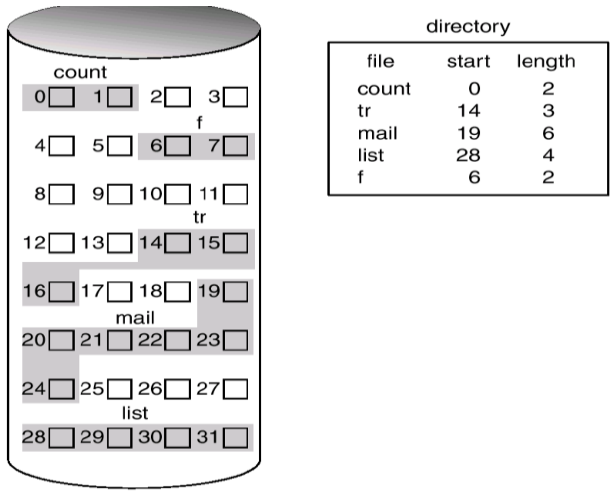
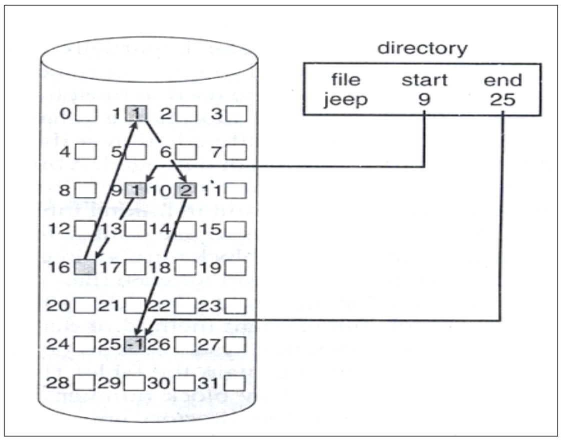
Sr. No	**Paging**	**Segmentation**
1.	It divides the physical memory into frames and program's address space into same size pages.	It divides the computer's physical memory and program's address space into segments.
2	Page is always of fixed block size.	Segment is of variable size.
3.	The size of the page is specified by the hardware.	The size of the segment is specified by the user.
4.	It may lead to internal fragmentation as the page is of fixed block size.	It may lead to external fragmentation as the memory is filled with the variable sized blocks.
5.	Page table is used to map pages with frames from memory.	Segment table is used to map segments with physical memory.
6.	Page table contains page number and frame number.	Segment table contains segment number, length of segment and base address of segment from memory.

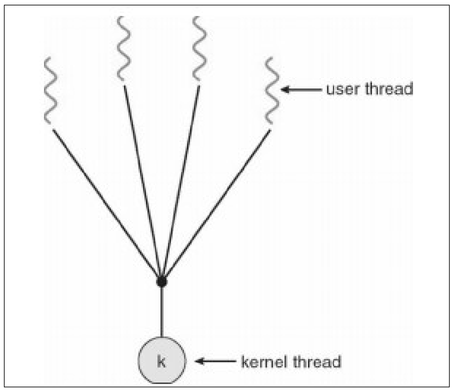
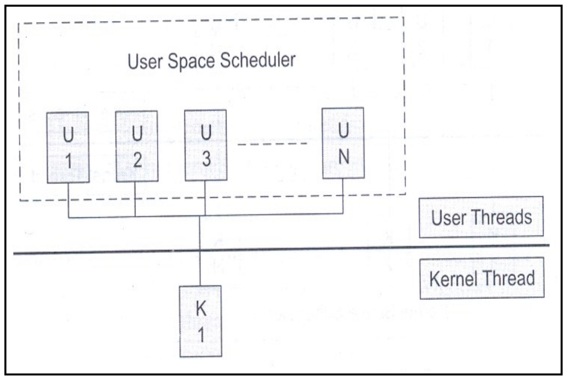
4.	**a)**	**Attempt any THREE of the following:**	**12 Marks**
	i)	**Explain booting procedure in details with the help of diagram.**	**4M**
	Ans:	The loading of the operating system is done with the help of a special program called BOOT. This program is stored in one or two sectors on the disk with a pre-determined address. This portion is called as BOOT Block. The ROM contains minimum program called as bootstrap program. When the computer is turn ON, the control is transferred to this program automatically by the hardware itself. This program in ROM locates the BOOT program and loads it into predetermined memory locations. This BOOT program loads the operating System into the memory.  Boot Block BOOT Memory Remaining part of the Operating System System Disk	**(Explanation: 2 marks, Diagram: 2 marks)**
	ii)	**Explain layered approach operating system.**	**4M**
	Ans:	 Application Program Application Program User Mode System Mode System Call Interface Layer N Layer 1 Layer 0 Hardware	**(Explanation:2 marks , diagram:2 marks)**

		OR  The modules of the operating system are divided into several layers stacked one above the other, thus forming a hierarchical structure. The lowest layer (Layer 0) interacts with the underlying hardware and the topmost layer (Layer N) provides an interface to the application programs/ users. Only adjacent layers can communicate with each other. A layer N can request for services only from a layer immediately below it (layer N-1). A layer N can provide services only to the layer immediately above it (layer N + 1). A Layer only needs to know what services are offered by the layer below it. In this structure any request that requires access to hardware has to go through all layers. Bypassing of layers is not allowed.	
	iii)	**Explain context switch with help of diagram.**	**4M**
	Ans:	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. System saves context of old process in its PCB and loads context of new process into the memory which is schedule to run next.	**(Description: 2 marks, Diagram: 2 marks)**

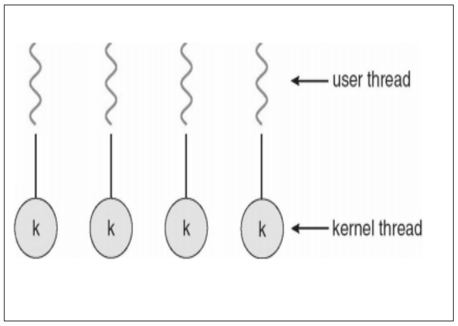
			
	iv)	**Describe how process is terminated.**	**4M**
	Ans:	A process terminates when it finishes executing its final statement and asks the operating system to delete it. The process may return a status value to its parent process. All the resources of the process including physical and virtual memory, open files and I/O buffers, are deallocated by the operating system. For win32- Terminate Process( ) is used to terminate a process. For Unix- exit ( ) is used to terminate a process A parent may terminate the execution of one of its children for a variety of reason, such as these: • The child has exceeded its usage of some of the resources that it has been allocated. • The task assigned to the child is no longer required. • The parent is exiting, and the operating system does not allow a child to continue if tis parent terminates.	**(Explanation of Process Termination: 4 marks*)***
	b)	**Attempt any ONE of the following:**	**6 Marks**
	i)	**With neat diagram describe Process Control Block (PCB)**	**6M**
	Ans:	Process Control Block **(PCB)** is a record or a data structure that is maintained for each and every process. A PCB is created when a process is created and it is removed from memory when process is terminated. A PCB may contain 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.	**(Explanation: 4 marks, Diagram:2 marks)**

		 Process Control Block (PCB) process state process number program counter registers memory limits list of open files • • • In general, a PCB may contain information regarding: **Process State**: It indicates current state of a process. Process state can be new, ready, running, waiting and terminated. **Process number**: Each process is associated with a unique number. **Program Counter**: It indicates the address of the next instruction to be executed for the process. **CPU Registers**: The registers vary in number and type depending on the computer architecture. Register includes accumulators, index registers, stack pointers and general purpose registers plus any condition code information. **CPU-scheduling information**: This information includes a process priority, pointers to scheduling queues and any other scheduling parameters. **Memory Management Information**: It includes information such as value of base and limit registers, page tables, segment tables, depending on the memory system used by OS. **Accounting Information**: This information includes the amount of CPU used, time limits, account holders, job or process number and so on. **I/O status information**: It includes information about list of I/O devices allocated to the process such as list of open files and so on.	
	ii)	**Explain following memory allocation methods:** **1) Contiguous** **2) Linked**	**6M**
	Ans:	**1. Contiguous Allocation** The contiguous allocation method requires each file to occupy a set of contiguous address on the disk. Disk addresses define a linear ordering on the disk. When a file has to be stored on a disk, system search for contiguous set of blocks as required by the file size i.e. system waits till it finds required number of memory blocks in sequence. When space is available system stores the file in the disk and makes an entry in the directory. With this ordering, accessing block b+1 after block b normally requires no head movement. Contiguous allocation of a file is defined by the disk address and the length of the first block. If the file is n blocks long, and starts at location b, then it occupies blocks b, b+1, b+2, …, b+n-1. The directory entry for each file indicates the address of	**(Explanation of contiguous Allocation-2 marks, diagram of Contiguous Allocation-1 mark ;explanation of linked**

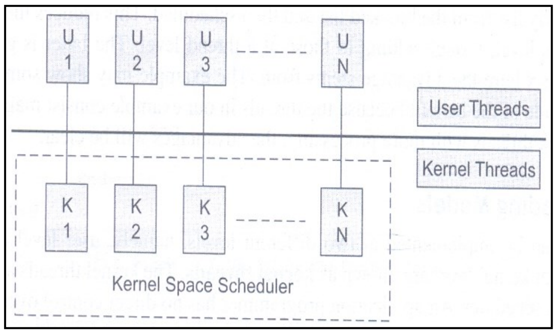
		the starting block and the length of the area allocated for this file.  count f tr mail list directory file start length count 0 2 tr 14 3 mail 19 6 list 28 4 f 6 2 **2. linked Allocation:-** In this method, each file occupies disk blocks scattered anywhere on the disk. It is a linked list of allocated blocks. When space has to be allocated to the file, any free block can be used from the disk and system makes an entry in directory. Directory entry for allocated file contains file name, a pointer to the first allocated block and last allocated block of the file. The file pointer is initialized to null value to indicate empty file. A write to a file, causes search of free block. After getting free block data is written to the file and that block is linked to the end of the file. To read the file, read blocks by following the pointers from block to block starting with block address specified in the directory entry. For example, a file of five blocks starting with block 9 and continue with block 16,then block 1,then block 10 an finally block 25.each allocated block contains a pointer to the next block. 	**Allocation: 2 marks, diagram of linked Allocation: 1 mark)**

5.		**Attempt any <u>TWO</u> of the following:**	**16 Marks**
	a)	**List and explain various type of multi-threading models with diagram.**	**8M**
	Ans:	Multithreading models are as follows: • Many to One • One to one • Many to Many **1) Many to One model** The many to one model maps many user-level threads to a single kernel thread.Thread management is done by the thread library in user space. It is efficient, but the entire process will block if a thread makes a blocking system call. Only one thread can access the kernel at a time, multiple threads are unable to run in parallel on multiprocessors. **Any one diagram:**  OR  User Space Scheduler U 1 U 2 U 3 U N User Threads Kernel Thread K 1 **2) One to One model** The one to one model maps each user thread to a single kernel thread. It provides more concurrency than the many to one model by allowing another thread to run when a thread makes a blocking system call. It also allows multiple threads to run in parallel on multiprocessors.	**(List: 2 marks, Description of all models :2 marks each)**

Any one diagram:



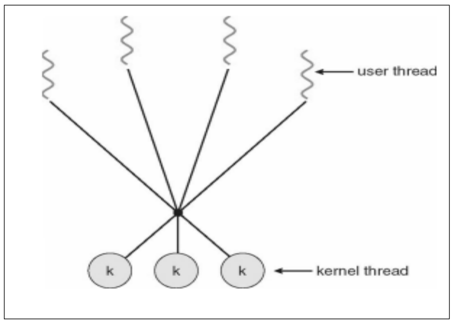
OR



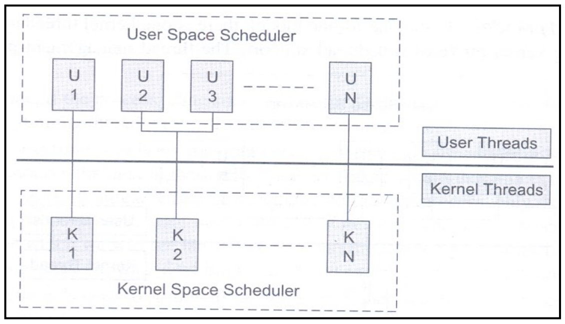
3) Many to Many model

The many to many model multiplexes many user level threads to a smaller or equal number of kernel threads. The number of kernel threads may be specific to either a particular application or a particular machine. It allows creating as many user level threads as required and the corresponding kernel level threads can run in parallel on a multiprocessor.

Any one diagram:



OR



	b)	**Enlist and describe in details conditions leading to Deadlocks.**	**8M**
	Ans:	Conditions leading to deadlock are: 1. Mutual Exclusion 2. Hold and wait 3. No-preemptive 4. Circular wait	**(List:2 marks, Description of four conditions: 1 ½ marks Each)**

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.

A disk drive can be shared by two processes simultaneously. This will not cause deadlock, but printers, tape drives, plotters etc. have to be allocated to a process in an exclusive manner until the process completely finishes its work with it, which normally happens when the process ends. This will cause deadlock.

2. Hold and Wait: Requesting process already holds resources while waiting for requested resources. Even if a process holds certain resources at any moment, it is possible for it to request for new resource. 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. It will not give the resource already held and request for new one. If it is true, deadlock will take place and if this is not true, a deadlock can never take place.

3. No-Preemption: Resources cannot be pre-empted i.e a resource can be released only voluntarily by the process holding it. Once the resources are allocated to the process, system cannot forcefully deallocate the resources from a process even though that process goes into the waiting state for additional resources.

4. Circular Wait: The processes 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.
The set of waiting processes P0, P1, P2, P3…..Pn waiting for resources which are already held by the next process. In this example, P0 is waiting for the resource already held by P1, P1 waiting for the resource already held by P2, P2 waiting for the resource already held by P3, Pn-1 waiting for the resource which is held by Pn and Pn waiting for the resource which is held by P0.

It is necessary to understand that all these four conditions have to be satisfied simultaneously for deadlock.

c) **The job are scheduled for execution as follows solve the problem using:** **8M**

(i) SIS
(ii) FCFS

also find average waiting time using Gantt chart.

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

Ans: **{**Note: If student has attempted to solve algorithm then give appropriate marks. **}**

(Gantt chart: 2 marks each, Average waiting time: 2 marks each)

{Note: the following solution is given considering SJF Algorithm**}**

i) SJF:

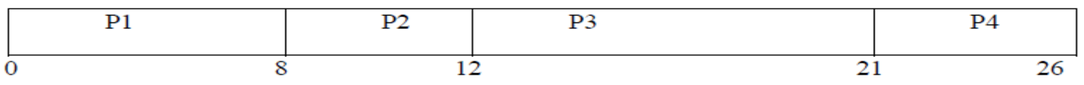
P1	P2	P4	P1	P3

0 1 5 10 17 26

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)

ii) 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)	
6.		**Attempt any FOUR of the following:**	**16 Marks**
	a)	**Describe real time system. State any one example of its application.**	**4M**
	Ans:	A real time system has well defined fixed time constraints. Processing should be done within the defined constraints. A primary objective of real-time systems is to provide quick event response time and thus meet the scheduling deadlines. 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. 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 – Flight Control System** All tasks in that system must execute on time. Example: Satellite application of real time OS- The satellite connected to the computer system sends the digital samples at the rate of 1000 samples per second. The computer system has an application program that stores these samples in a file. The sample sent by the satellite arrives every millisecond to the application. So computer must store or respond the sample in less than 1 millisecond. If the computer does not respond to the sample within this time, the sample will lost. Some of the examples of Real time systems are: A web server, A word processor, An audio/video media center, A microwave oven, A chess computer.	**(Relevant description: 2 marks, Any relevant example:2 marks)**
	b)	**Enlist system components. Describe any two in detail.**	**4M**
	Ans:	**List of System Components:** **1. Process management** **2. Main memory management** **3. File management** **4. I/O system management** **5. Secondary storage management** **1. 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. It's tempting to think of a process as an application, but that gives an incomplete picture of how processes relate to the operating	**(List:1 mark, Description of any two: 1 ½ marks each)**

system and hardware. The application you see (word processor or spreadsheet or game) is, indeed, a process, but that application may cause several other processes to begin, for tasks like communications with other devices or other computers. There are also numerous processes that run without giving you direct evidence that they ever exist. A process, then, is software that performs some action and can be controlled by a user, by other applications or by the operating system. It is processes, rather than applications, that the operating system controls and schedules for execution by the CPU. In a single-tasking system, the schedule is straightforward. The operating system allows the application to begin running, suspending the execution only long enough to deal with interrupts and user input.

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

1. Creation and deletion of user and system processes.
2. Suspension and resumption of processes.
3. A mechanism for process synchronization.
4. A mechanism for process communication.
5. A mechanism for deadlock handling.

2. Main-Memory Management

Services provided under Memory Management are directed to keeping track of memory and allocating/de allocating it to various processes. The OS keeps a list of free memory locations. Before a program is loaded in the memory from the disk, this MM consults the free list, allocates the memory to the process, depending upon the program size and updates the list of free memory. Primary-Memory or Main-Memory is a large array of words or bytes. Each word or byte has its own address. Main-memory provides storage that can be access directly by the CPU. That is to say for a program to be executed, it must in the main memory.

The major activities of an operating in regard to memory-management are:

1. Keeping track of which parts of memory are currently being used and by whom.
2. Deciding which processes (or parts thereof) and data to move into and out of memory.
3. Allocating and deallocating memory space as needed.

3. File Management

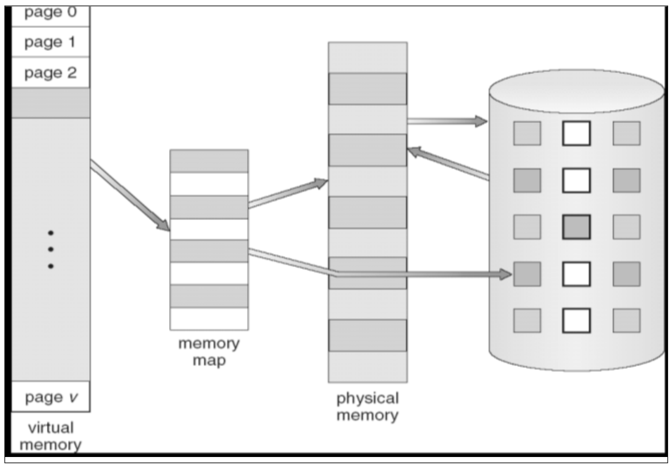
A file is a collected of related information defined by its creator. Computer can store files on the disk (secondary storage), which provide long term storage. Some examples of storage media are magnetic tape, magnetic disk and optical disk. Each of these media has its own properties like speed, capacity, and data transfer rate and access methods. A file system normally organized into directories to ease their use. These directories may contain files and other directions.

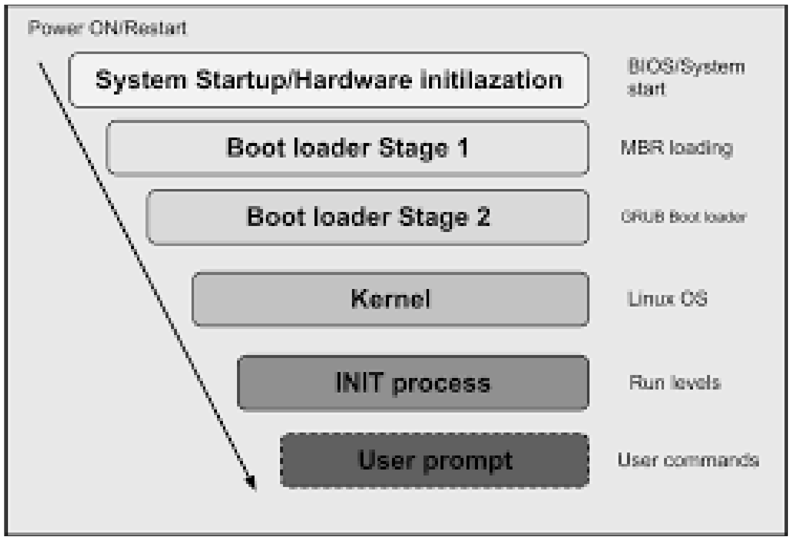
The five main major activities of an operating system in regard to file management are

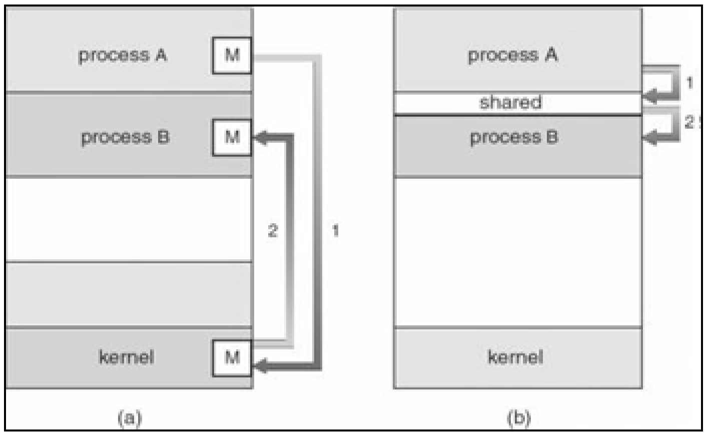
1. The creation and deletion of files.
2. The creation and deletion of directions.
3. The support of primitives for manipulating files and directions.
4. The mapping of files onto secondary storage.
5. The backup of files on stable storage media.

4. I/O device Management

		Input / Output device management provides an environment for the better interaction between system and the I / O devices. To interact with I/O devices in an effective manner, the operating system uses some special programs known as device driver. The device drivers take the data that operating system has defined as a file and then translate them into streams of bits or a series of laser pulses. The device driver is a specialized hardware dependent computer program that enables another program, typically an operating system to interact transparently with a hardware device, and usually provides the required interrupt handling necessary for the time dependent hardware interfacing. The I/O subsystem consists of several components: 1. A memory management component that includes buffering, caching, spooling 2. A general device driver interface 3. Drivers for specific hardware devices **5. Secondary-Storage Management** Systems 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. Each location in storage has an address; the set of all addresses available to a program is called an address space. **The three 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.	
	c)	**Describe the concept of virtual memory with suitable example.**	**4M**
	Ans:	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. On systems which support virtual memory, overlays have virtually disappeared. **Example:** For example, a 16M program can run on a 4M machine by carefully choosing which 4M to keep in memory at each instant, with pieces of the program being swapped between disk and memory as needed.	**(Description:2 marks, Example: 2 marks)**

		 **Fig: Virtual Memory**	
	d)	**Describe stepwise booting process of unix along with diagram.**	**4M**
	Ans:	Boot strapping is the process by which the computer system starts working. The process involves several steps. **1. Loading UNIX into Memory:** When system is powered ON, computer accesses a small ROM based start-up routine that performs elementary system verifications. (Assuring HDD and networks) the boot routine does more sophisticated hardware verifications and then loads kernel file UNIX from systems root directory to system RAM. **2. Executing the kernel:** Once loaded into the memory, the kernel starts working. It sets up the information table needed to control UNIX environment, checks system hardware, checks memory available, hardware devices and then creates the swapper process. **3. Swapper process:** Swapper process is identified as process 0. It monitors memory management overhead, when overhead, swapper removes entire process from memory till system performance becomes acceptable. **4. Init process:** init initialises system processes, places system in multi-user mode unless single user mode is specified, sets the computer name and environment variables such as PATH and HOME checks the file system with fsck command, deletes temporary files, initiates network services and starts the /etc/getty process. **5. Getty:** initiates individual terminal lines. It periodically checks for terminals that are switched ON. After terminal is ON, it prints the login prompt, prompting for users login name, once user enters a login name, getty spawns or starts the login process for that terminal. **6. Login:** The login process prompts the user for a password. It validates the login name and the password against the entry in the /etc/passwd file and the /etc/shadow file. The users shell specified in the Home directory. **7. Shell:** The shell prints the Unix prompt and executes user commands when user logs out, sh is taken over by login to allow the next user to log in.	**(Description:2 marks, Diagram:2 marks)**

		 Power ON/Restart System Startup/Hardware initilazation BIOS/System start Boot loader Stage 1 MBR loading Boot loader Stage 2 GRUB Boot loader Kernel Linux OS INIT process Run levels User prompt User commands	
	e)	**Draw and explain Inter-process communication model.**	**4M**
	Ans:	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: a. **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. b. **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 processes that want to communicate with each other must have a communication link between them. Between each pair of processes exactly one communication link.	**(Diagram:2 marks, Explanation: 2 marks)**

		 a) Message Passing b) Shared Memory	