MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

SUMMER – 19 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
	(i)	**Describe 2nd and 3rd generation of operating system.**	4
	Ans:	**Second generation: 1955-1965** Around 1955, transistors were introduced. The transistor was far superior to the vacuum tube, allowing computers to become smaller, faster, cheaper, more energy-efficient and more reliable than their first-generation predecessors. Second-generation computers relied on punched cards for input and printouts for output. Assembly language which allowed programmers to specify instructions in words, introduced as second generation Language Then IBM-7094-a faster and larger computer came into picture. In that, control cards were in use. In this system, cards were arranged as a stack to save CPU time. All these cards were then read one by one and copied onto a tape using a 'card to tape' utility program. The prepared tape was taken to the main computer and processed. • Technology used: Transistor • Memory: Magnetic core technology • Programming: Assembly level language • Example: IBM-1401, IBM-7094, IBM 1620, CDC 3600 **Advantages:** • Smaller in size as compared to first generations computers. • More reliable. • Less heat generated as compared to first generation machine.	Description of each generation-2M

- These computers were able to reduce computational times from milliseconds to microseconds.
- Better portability.
- Wider commercial use.
- Less prone to hardware failure.
- Other components are invented like printers, tape storage, memory, OS, stored program.
- less expensive than vacuum tube
- Magnetic disk and magnetic tapes used as secondary storage devices.
- High level languages such as FORTRAN, COBOL, ALGOL were used for programming.

Third generation: 1965-1980

Third generation came with introduction of Integrated Circuits(IC). Transistors were placed on silicon chips, called semiconductors, which drastically increased the speed and efficiency of computers. With ICs, the cost and the size of the computer reduced and the performance improved. The systems of the 1960's were also batch processing systems, but they were able to take better advantage of the computer's resources by running several jobs at once. So operating systems designers developed the concept of multiprogramming in which several jobs are in main memory at once; a processor is switched from job to job as needed to keep jobs advancing while keeping the peripheral devices in use.

- Technology used: Integrated Circuits
- Memory: Disk
- Programming: Job Control Language
- Example: IBM 360 mainframe, IBM-370, PDP-8, VAX 750

Advantages:

- Smaller in size as compared to previous.
- More reliable and easily portable.
- Lower heat generated the second-generation computers.
- Reduce computational times from microseconds to nanoseconds.
- Maintenance cost is low because hardware failure is rare.
- Widely used for various commercial applications all over the world.
- Less power requirement.
- Commercial production was easier and cheaper.
- language used are BASIC (Beginners all-purpose symbolic instruction code), PASCAL, RPG (Report Program Generator)
- Keyboard used as input and VDU used as output devices.
- Capable of multiprogramming.
- Increase Processing Speed

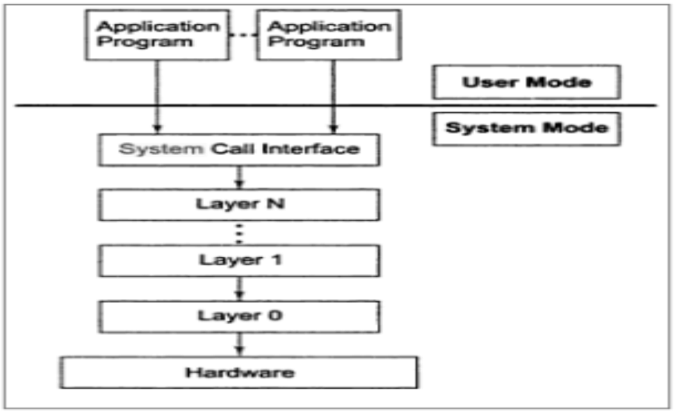
	(ii)	**Describe Layered Structure of Operating System**	**4**
	Ans:	The modules of the operating system are divided into several layers stacked one above the other, thus forming a hierarchical structure. The lowest layer (Layer 0) interacts with the underlying hardware and the topmost layer (Layer N) 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.	Description 2M Structure Diagram- 2M

Advantage:

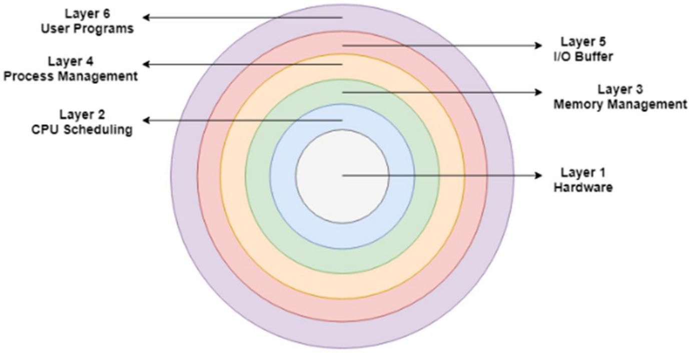
- This approach makes it easy to build, maintain and enhance the operating system.
- Locating an error is easy as system can start debugging from 0th layer and proceed further covering entire system if required.

Disadvantage:

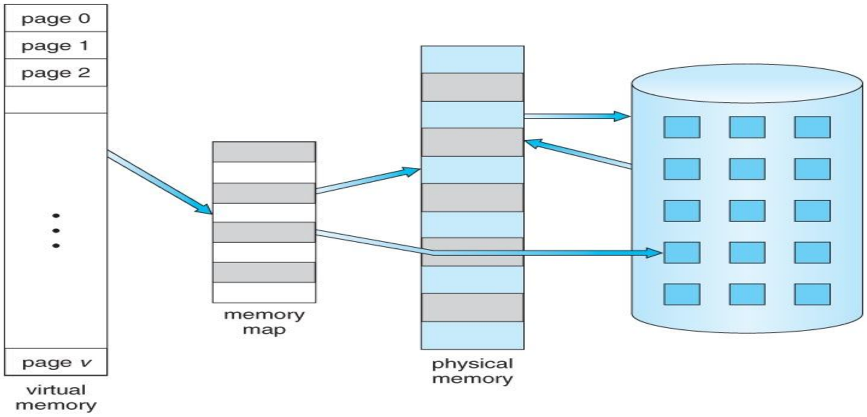
- Overall performance speed is slow as requests pass through multiple layers of software before they reach the hardware.



OR



LAYERED OPERATING SYSTEM

	(iii)	**Explain concept of Virtual Memory with Diagram**	**4**
	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 for execution of program. 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.  page 0 page 1 page 2 page v virtual memory memory map physical memory Example: Consider, 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.	**Explanation 2M Diagram 2M**
	(iv)	**Explain Real Time Operating System. Explain its types**	**4**
	Ans:	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. User convenience and resource utilization are of secondary concern to real-time 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:	Explanation 2M Types: 1M each

- Manage the processor and other system resources to meet the requirements of an application.
- Synchronize with and respond to the system events.
- 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.

1	**b)**	**Attempt any ONE of the following**	**6**
	(i)	**Difference between Segmentation and Paging (Any 6 points)**	**6**

Paging	**Segmentation**
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.
Page is always of fixed block size.	Segment is of variable size.
The size of the page is specified by the hardware.	The size of the segment is specified by the user.
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.
Page table is used to map pages with frames from memory.	Segment table is used to map segments with physical memory.
Page table contains page number and frame number.	Segment table contains segment number, length of segment and base address of segment from memory.
Invisible to Programmer	Visible to programmer
Paging consist of Static linking & dynamic loading	Segment consist of Dynamic Linking & Dynamic Loading
A page is of physical unit	A page is of logical unit

Any Six relevant points: 1 M each

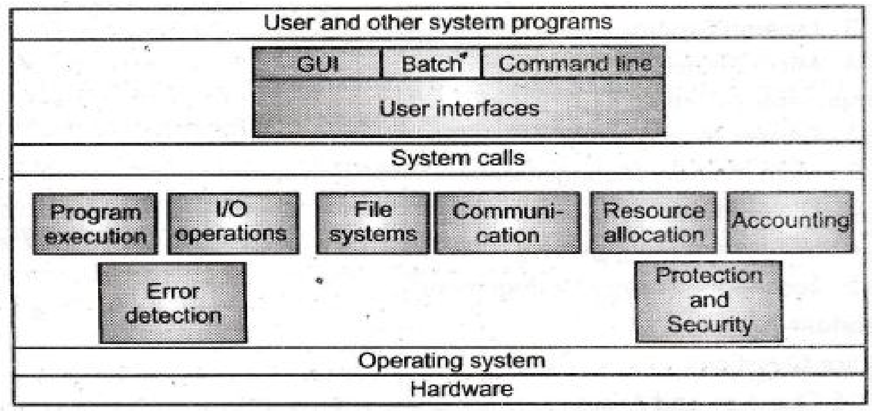
	(ii)	**Explain any six services of Operating System. Draw diagram of services of OS**	**6**
	Ans:	**1. User Interface:** All operating systems have a user interface that allows users to communicate with the system. Three types of user interfaces are available: a. Command line interface (CLI) b. Batch interface c. Graphical user interface (GUI) **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 CPU is assigned to that process for its execution. 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. **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.	Services: 4M Diagram: 2M

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.

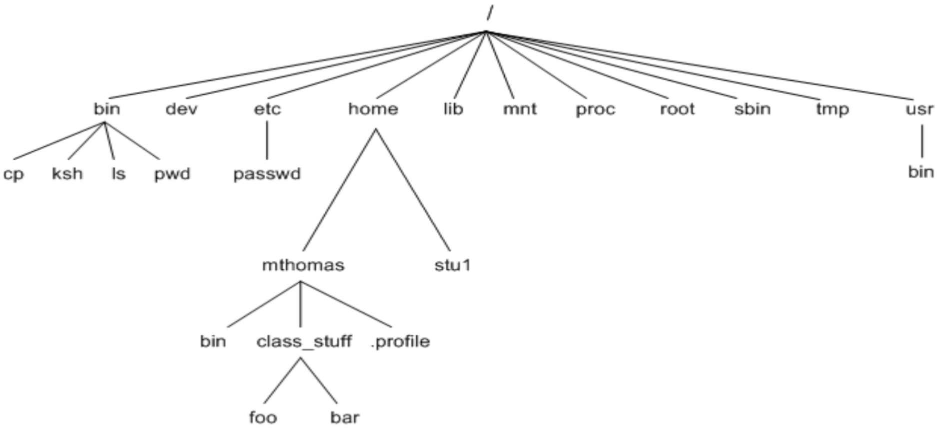
7. Resource allocation: When there are multiple users or multiple processes running at the same time, resources must be allocated to each of them. Operating system manages resource allocation to the processes. These resources are CPU, main memory, file storage and I/O devices. For maximizing use of CPU,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

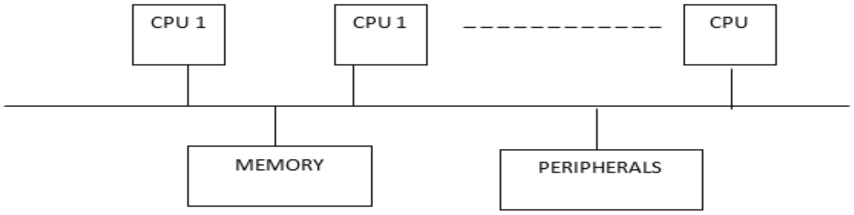
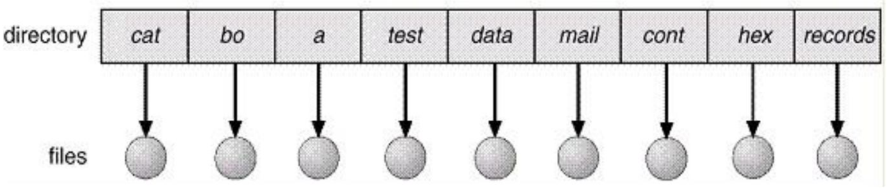
8. Accounting: Operating system keeps track of usages of various computer resources allocated to users. This accounting is used for reconfiguration of system to improve computing services.

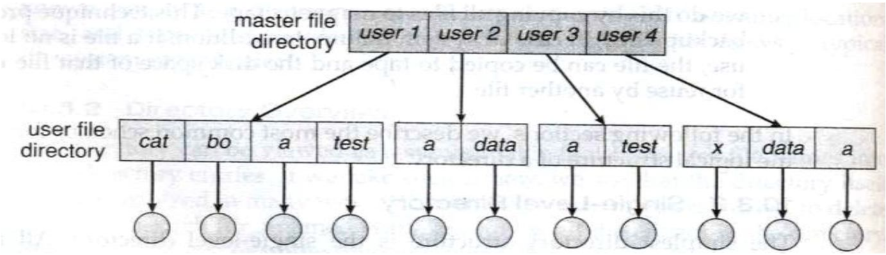
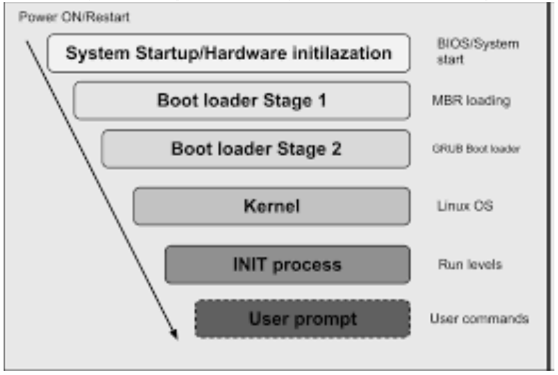
9. Protection & security: Owners of information stored in a multiuser or networked computer system want to control use of that information. When several separate processes execute concurrently, one process should not interfere with the other processes or operating system itself. Protection provides controlled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

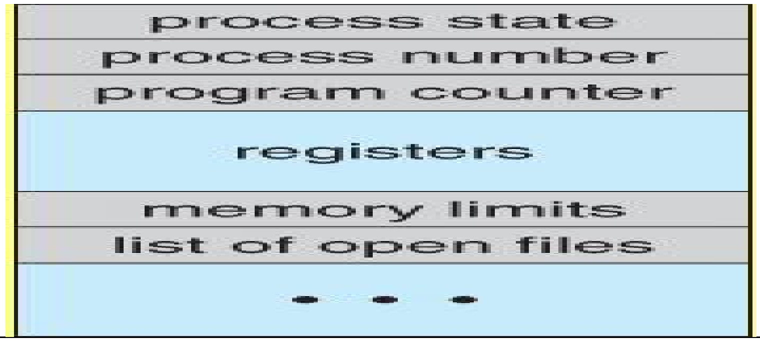


2		**Attempt any FOUR of the following**	**16**
	a)	**Explain file system of UNIX**	**4**

	Ans:	 / bin dev etc home lib mnt proc root sbin tmp usr cp ksh ls pwd passwd bin mthomas stu1 bin class_stuff .profile foo bar The Unix file system is a methodology for logically organizing and storing large quantities of data such that the system is easy to manage. A file can be informally defined as a collection of related data, which can be logically viewed as a stream of bytes (i.e. characters). A file is the smallest unit of storage in the Unix file system. The Unix file system has a hierarchical (or tree-like) structure with its highest level directory called root (denoted by /, pronounced slash). Immediately below the root level directory are several subdirectories, most of which contain system files. Below this can exist system files, application files, and/or user data files. Similar to the concept of the process parent-child relationship, all files on a UNIX system are related to one another. That is, files also have a parent-child existence. Thus, all files (except one) share a common parental link, the top-most file (i.e. /) being the exception. Below is a diagram of a "typical" Unix file system. The top-most directory is / (slash), with the directories directly beneath being system directories.	Explanation: 3M Diagram: 1M
	b)	**Describe Multiprocessor Operating System with its two advantages**	**4**
	Ans:	Multiprocessor systems are also known as parallel systems or tightly coupled systems. These systems have two or more processors in close communication and they share computer resources such as bus, clock, memory and peripheral devices. The whole task of multiprocessing is managed by the operating system, which allocates different tasks to be performed by the various processors in the system. Applications designed for the use in multiprocessing are said to be threaded, which means that they are broken into smaller routines that can be run independently. This allows the operating system to let these threads run on more than one processor simultaneously, which is multiprocessing that results in improved performance. Generally, the parallel processing is used in the fields like artificial intelligence and expert system, image processing, weather forecasting etc.	Description: 2M Two Advantages: 2 M

		 **Advantages:** • **Increased throughput:** Increase in number of processors requires less time for more work. Number of processes completing their task for a particular duration is more. • **Economy of scale:** Cost is less than multiple single processor systems. If several programs operate on the same set of data, then it is cheaper to store those data on one disk and to have all the processors share them. • **Increase reliability:** Functions can be distributed properly among several processors and then the failure of one processor will not halt the system.	
	(c)	**List different directory structure and explain any one in detail**	**4**
	Ans:	**List of directory structures:** • Single level directory structure • Two level directory structure **Single level directory structure:** It is the simplest form of directory structure, having one directory containing all the files, and each file must have a unique name. Software design is simple. The advantages of this scheme are its simplicity and the ability to locate files quickly. Since all files are in the same directory, they must have unique names. If there are two users who call their data file "test", then the unique-name rule is violated. Even with a single-user, as the number of files increases, it becomes difficult to remember the names of all the files in order to create files with unique name.  directory cat bo a test data mail cont hex records files **Two level directory structure:** In this structure, each user has its own user file directory (UFD). The UFD lists only files of a single user. System contains a master file directory (MFD) which is indexed by user name or account number. Each entry in MFD points to the UFD for that user. When a user refers to a particular file, only his own UFD is searched. Different users can have files with the same name, as long as all the file names within each UFD are unique. When we create a file for a user, operating system searches only that user's UFD same name file already present in the directory.	List: 1M Explain One Structure: 2M, Diagram: 1M

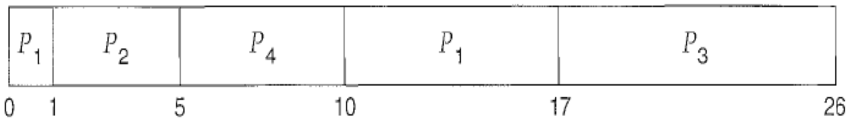
		For deleting a file again operating system checks the file name in the user's UFD only.  master file directory user 1 user 2 user 3 user 4 user file directory cat bo a test a data a test x data a	
	d)	**Explain Booting Procedure of UNIX**	**4**
	Ans:	Booting is the process by which the computer system starts working.  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 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 initializes 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.	Relevant Explanation: 4M

		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.	
	(e)	**Explain Process Control Block with suitable Diagram**	**4**
	Ans:	Each process is represented as a process control block (PCB) in the operating system. It contains information associated with specific process. **Process State**: It indicates current states of a process. Process state can be new, ready, running, waiting and terminated. **Process number**: Each process is identified by its process number, called process identification number (PID). **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. **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 operating system. **Accounting Information**: This information includes the amount of CPU used, time limits, account holders, job or process number and so on. It also includes information about listed I/O devices allocated to the process such as list of open files.  process state process number program counter registers memory limits list of open files • • •	Explanation: 2M Diagram: 2M
	(f)	**Explain Shortest Remaining Time Next (SRTN) scheduling algorithm with example**	**4**
	Ans:	**SRTN: Shortest Remaining Time Next** A Shortest remaining Time Next scheduling algorithm is also referred as preemptive SJF scheduling algorithm. When a new process arrives at ready queue while one process is still executing then SRTN algorithm is performed to decide which process will execute next. This algorithm compare CPU burst time of newly arrived process with remaining (left) CPU burst time of currently executing process. If CPU burst time of new process is less than remaining time of current process then SRTN algorithm preempts current process execution and starts executing new process.	Explanation: 2M Example: 2M

Example: Consider four processes with arrival time and burst time mentioned below in table.

Process	Arrival Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

Gantt chart:



- As P_1 is the only process in ready queue at time 0,P_1 will start execution first.
- At time 1, process P_2 arrives. The Burst time of P_2 i.e 4 ms is less than remaining burst time of process P_1 i.e.7 ms, so process P_1 is preempted and process P_2 starts executing.
- At time 2, process P_3 arrives. The Burst time of P_3 i.e. 9 ms is greater than remaining burst time of process P_2 i.e. 3 ms, so Process P_2 continues its execution.
- At time 3, process P_4 arrives. The Burst time of P_4 i.e. 5 ms is greater than remaining burst time of process P_2 i.e. 2 ms, so Process P_2 continues its execution.
- When P_2 process completes its execution, all remaining processes execute with shortest job first algorithm.

Waiting time of processes:

P_1:9 ms

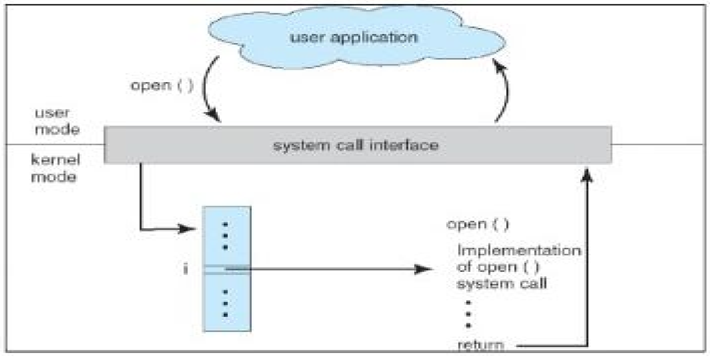
P_2:0 ms

P_3:15 ms

P_4: 2 ms

Average waiting time= (9+0+15+2)/4=26/4=6.5 ms

3		**Attempt any FOUR of the following**	**16**
	a)	**Explain execution of system call with diagram**	**4**
	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. 1. Each system call associated with a particular number. 2. System call interface maintains a table indexed according to these numbers. 3. The system call interface invokes intended system call in operating system kernel and returns status of the system call and any return values.	Explanation: 2M Diagram: 2M

		4. The caller needs to know nothing about how the system call is implemented. Just needs to obey API and understand what OS will do as a result call. 5. Most details of operating system interface hidden from programmers by API. It is managed by run-time support library.  user application open ( ) user mode system call interface kernel mode i open ( ) Implementation of open ( ) system call return Open ( ) system call for most file systems, a program initializes access to a file in a file system using the open system call. This allocates resources associated to the file (the file descriptor), and returns a handle that the process will use to refer to that file.	
	b)	**Explain different file attributes**	**4**
	Ans:	File attributes: • **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.	Explanation of any 4 file attributes: 1 mark each
	c)	**Explain any four benefits of using threads**	**4**
	Ans:	The benefits of using threads: • **Responsiveness:** Multithreading an interactive application may allow a program to continue running even if part of it is blocked or is performing a lengthy operation, thereby increasing responsiveness to the user. For instance, a multithreaded web browser could still allow user interaction in one thread while an image was being loaded in another thread. • **Resource sharing:** By default, threads share the memory and the resources of the process to which they belong. The benefit of sharing	Any four Benefits: 1M each

		code and data is that it allows an application to have several different threads of activity within the same address space. • **Economy:** Allocating memory and resources for process creation is costly. Because threads share resources of the process to which they belong, it is more economical to create and context-switch threads. Empirically gauging the difference in overhead can be difficult, but in general it is much more time consuming to create and manage processes than threads. In Solaris, for example, creating a process is about thirty times slower than is creating a thread, and context switching is about five times slower. • **Utilization of multiprocessor architectures:** The benefits of multithreading can be greatly increased in a multiprocessor architecture, where threads may be running in parallel on different processors. A single threaded process can only run on one CPU, no matter how many are available. Multithreading on a multi-CPU machine increases concurrency.	
	d)	**Write Steps of banker's algorithm to avoid deadlock**	**4**
	Ans:	**Steps of 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 updating 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 to any process in Matrix B and not a single process can complete 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 indicating 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.	Correct Steps: 4M

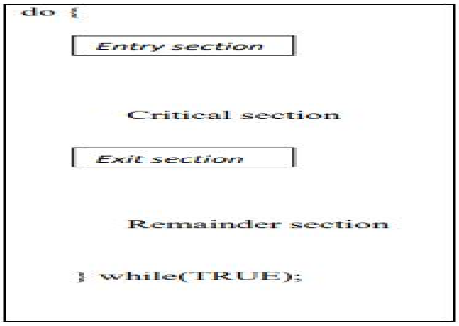
	e)	**Differentiate between pre-emptive and non-pre-emptive scheduling (any 4 points)**	**4**
	Ans:		Any four points: 1M each

Pre-emptive Scheduling	Non Pre-emptive 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.	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.
Throughput is less	Throughput is high.
Only the processes having higher priority are scheduled.	Processes having any priority can get scheduled.
It doesn't treat all processes as equal.	It treats all process as equal
Algorithm design is complex.	Algorithm design is simple
Circumstances for preemptive (i) Process switch from running to ready state (ii) Process switch from waiting to ready state	Circumstances for Non-preemptive Process switches from running to waiting state Process terminates
For e.g.: Round Robin, Priority Algorithms	For e.g.: FCFS Algorithm

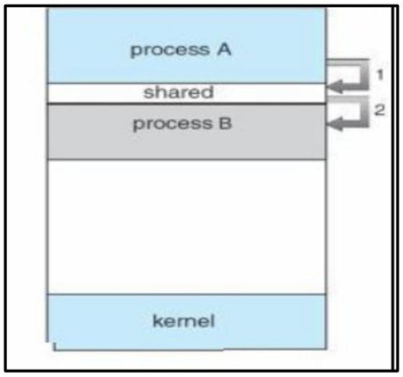
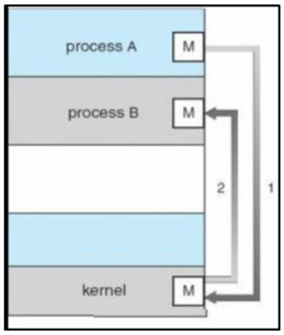
4	**a)**	**Attempt any THREE of the following**	**12**
	(i)	**Differential between Monolithic and Microkernel OS (Any four points)**	**4**
	Ans:		Any four points: 1M each

Monolithic OS	Microkernel OS
The entire O.S. is placed inside the kernel	Only bare minimum code is placed inside the kernel (only basic memory management and Inter Process Communication code)
It runs as a single large process	Here the kernel is broken down into processes called as servers
As all the services are placed inside the kernel, they have a single address space	As services (Servers provide services) are separated they have different address spaces
It is easy to implement/code	It is tough to implement/code
Performance is high (As kernel can invoke any function directly as everything is placed in the kernel)	Performance is low (As servers are separated, so to invoke services from other servers IPC(Inter Process Communication) is needed which requires kernel's permission and thus increases access time and lowers the performance)

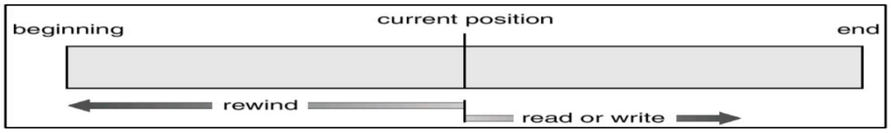
Less Secure (If one service fails, entire system crashes)	More Secure (Even if one service crashes, others can function properly because of separation)

	(ii)	**Explain critical section problem with example**	**4**
	Ans:	Each process contains two sections. One is critical section where a process may need to access common variable or objects and other is remaining section containing instructions for processing of sharable objects or local objects of the process. Each process must request for permission to enter inside its critical section. The section of code implementing this request is the entry section. In entry section if a process gets permission to enter into the critical section then it works with common data. At this time all other processes are in waiting state for the same data. The critical section is followed by an exit section. Once the process completes its task, it releases the common data in exit section. Then the remaining code placed in the remainder section is executed by the process.  do { Entry section Critical section Exit section Remainder section } while(TRUE); Two processes cannot execute their critical sections at the same time. The critical section problem is to design a protocol that the processes can use to cooperate i.e. allowing entry to only one process at a time inside the critical section. Before entering into the critical section each process must request for permission to entry inside critical section.	Explanation: 2M Example: 2M
	(iii)	**Explain different activities of I/O system management components of OS**	**4**
	Ans:	I/O System: Input / Output device management provides an environment for the better interaction between system and the I / O devices such as printers, scanners, tape drives etc. 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. A device driver is a specific type of computer software that is developed to allow interaction with hardware devices. Typically this continues an interface for communicating with the I/O device, through the specific computer bus or communication subsystem that the hardware is connected with. 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. **Activities:** • Providing interfaces to other system components.	Description of four activities of I/O system: 1 mark each

		• Managing devices • Transferring data • Detecting I/O completion	
	(iv)	**Explain user thread and kernel threads**	**4**
	Ans:	**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 systems 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 are managed.	Explanation of User Thread: 2 marks, Explanation of Kernel Thread: 2 marks
	b)	**Attempt user ONE of the Following**	**6**
	(i)	**Explain different methods of inter process communication with help of diagram**	**6**
	Ans:	There are two methods of IPC: **Shared memory:**	Two Methods with Description of each: 3 marks (1 mark

		 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. **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 mark Explanation)
	(ii)	**Explain different file access methods**	**6**
	Ans:	**Sequential Access Method:**	Two 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.

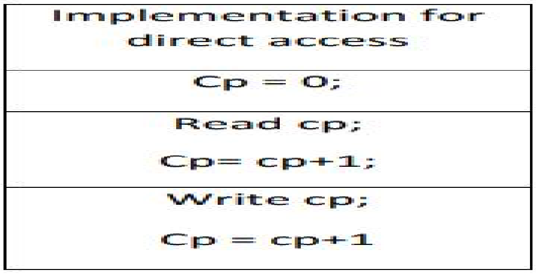
Direct Access Method:

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.

Description of each: 3 marks (1 mark Diagram, 2 mark Explanation)



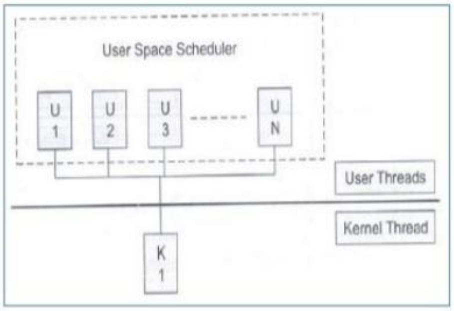
Implementation for direct access
Cp = 0;

Read cp;
Cp= cp+1;

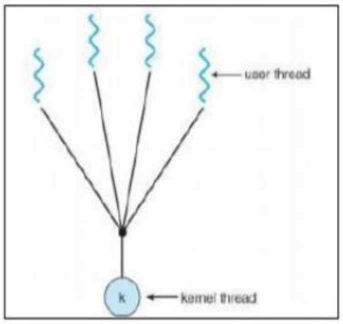
Write cp;
Cp = cp+1

5		**Attempt any TWO of the following**	**16**
	(a)	**Explain following multithreading models with advantages and disadvantages (i) Many to one (ii) Many to Many**	**8**
	Ans:	**Many to One Model:**	Explanation: 1M, Diagram: 1M Advantages: 1M Disadv: 1M = 4M For each Model

- This model maps many user level threads to one kernel level thread.
- If user level thread generates blocking system call then it blocks an entire process.
- At a time only one user level thread can access kernel level thread i.e multiple threads can't execute in parallel.
- Thread management is done by Thread libraries.
- Example: - Green threads – a thread library available for Solaris use many-to-one model.



OR



Advantages:-

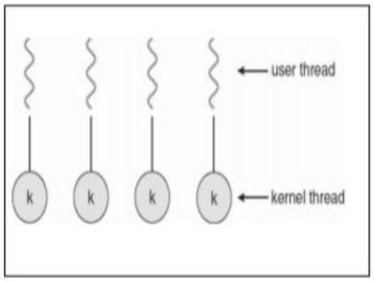
- It is an efficient model as threads are managed by thread library in user space.
- Portable: Because user level threads packages are implemented entirely with standard Unix and POSIX library calls, they are often quite portable.
- One kernel level thread controls multiple user level threads.
- Easy to do with few system dependencies.

Disadvantages:

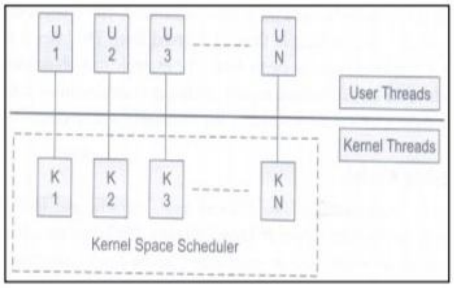
- One block call from kernel level thread blocks all user level threads.
- Cannot take advantage of multiprocessing.

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.
- Whenever user level thread is created, it compulsorily creates corresponding kernel level thread.
- This model is used in Linux & Windows version like 95,97,XP, NT.



OR



Advantages:

- It allows multiple threads to run in parallel on multiprocessors.
- More concurrency
- Less complication in processing

Disadvantages:

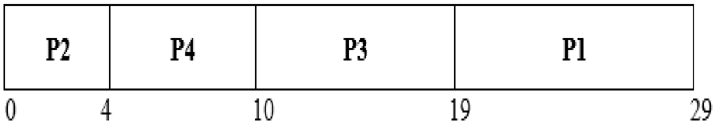
- Creating a user thread requires creating the corresponding kernel thread. Creating kernel thread may affect the performance of an application.
- It reduces performance of the system.
- Kernel thread is overhead.

b) **Calculate Average locating Time for SJF (Shortest Job First) and Round Robin (RR) algorithm for following table: (Time Slice 4 msec)** **8**

Process	Burst Time
P1	10
P2	04
P3	09
P4	06

Ans: **SJF:**

Gantt Chart:



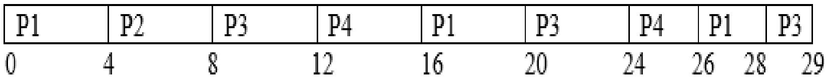
Waiting Time and Turn Around Time Table:

Process	Burst Time	Waiting Time	Turn Around Time
P1	10	19	29
P2	04	0	04
P3	09	10	19
P4	06	4	10

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

Average waiting time: (19 + 0 + 10 + 4)/4 = 8.25
Average turn around time: (29 + 4 + 19 + 10)/4 = 15.5

RR:
Gantt Chart:



Waiting Time and Turn Around Time Table:

Process	Burst Time	Waiting Time	Turn Around Time
P1	10	18	28
P2	04	04	08
P3	09	20	29
P4	06	20	26

Average waiting time: (18 + 04 + 20 + 20)/4 = 15.5
Average turn around time: (28 + 08 + 29 + 26)/4 = 22.75

	c)	**Explain first come first served (FCFS) algorithm. Give one example. State any one advantages and one disadvantage.**	**8**

Ans:

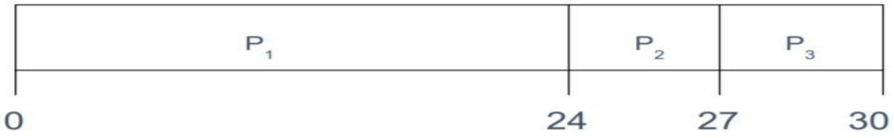
- First-Come - First-Served (FCFS) Scheduling FCFS scheduling is non preemptive algorithm.
- Once the CPU is allocated to a process, it keeps the CPU until it releases the CPU, either by terminating or by requesting I/O.
- In this algorithm, a process, that a request the CPU first, is allocated the CPU first. FCFS scheduling is implemented with a FIFO queue.
- When a process enters the ready queue, its PCB is linked to the tail of the queue.
- When the CPU is available, it is allocated to the process at the head of the queue. Once the CPU is allocated to a process, that process is removed from the queue.
- The process releases the CPU by its own.

4 Marks:- FCFS algorithm; 2 Marks:- Any relevant Example; 1 Mark:- Advantage; 1 Mark:- Disadvantage

Example:

Process	Burst Time
P1	24
P2	3
P3	3

Suppose that the processes arrive in the order: P1, P2, P3
Gantt Chart:



Process	Burst Time	Waiting Time	Turn Around Time
P1	24	0	24
P2	3	24	27
P3	3	27	30

Average waiting time: (0 + 24 + 27)/3 = 17

Average turn around time: (24 + 27 + 30)/3 = 27

Advantage:

It is simple to implement.

Disadvantage:

- This scheduling method is non preemptive, that is, the process will run until it finishes. Because of this non preemptive scheduling, short processes which are at the back of the queue have to wait for the long process at the front to finish.
- It is not suitable for real time systems.
- Average waiting time and average turnaround time is more comparatively.

6 **Attempt any FOUR of the following** **16**

a) Explain main memory Management components of OS with its activities **4**

Ans:

- 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.

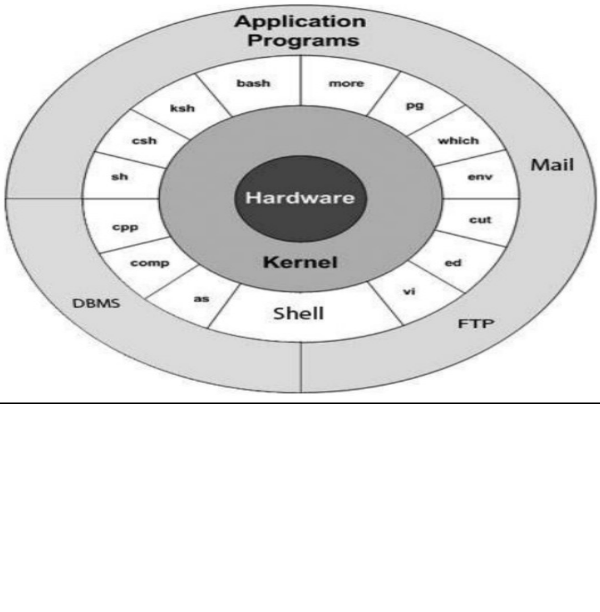
The major activities of an operating in regard to memory-management are:

- Keep track of which part of memory are currently being used and by whom.
- Decide which processes are loaded into memory when memory space becomes available.
- Allocate memory space as needed Deallocate memory space as needed

2 Marks:- Explanation; 2 Marks:- Activities

b) Explain structure of Unix OS **4**

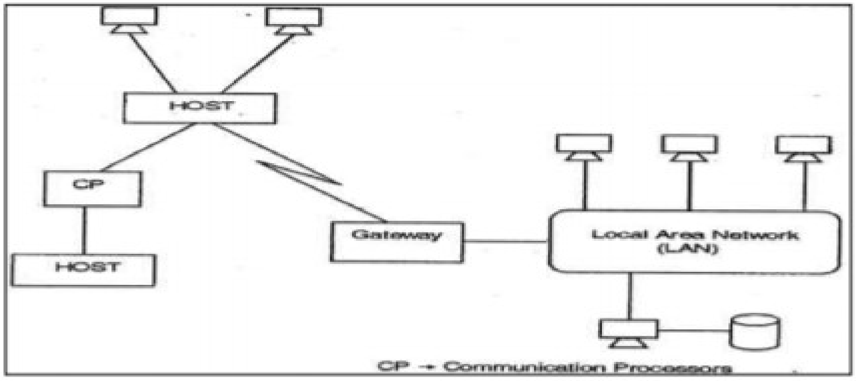
Ans:



2 Marks:- Explanation; 2 Marks:- Diagram

		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.	
	c)	**Explain distributed Operating System with advantages and disadvantages**	**4**
	Ans:	• 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 • The structure shown in figure contains a set of individual computer systems and workstations connected via communication systems.	2 Marks:- Explanation; 1 Mark:- Advantage; 1 Mark:- Disadvantage

- 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.



Advantages:

- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speedup the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

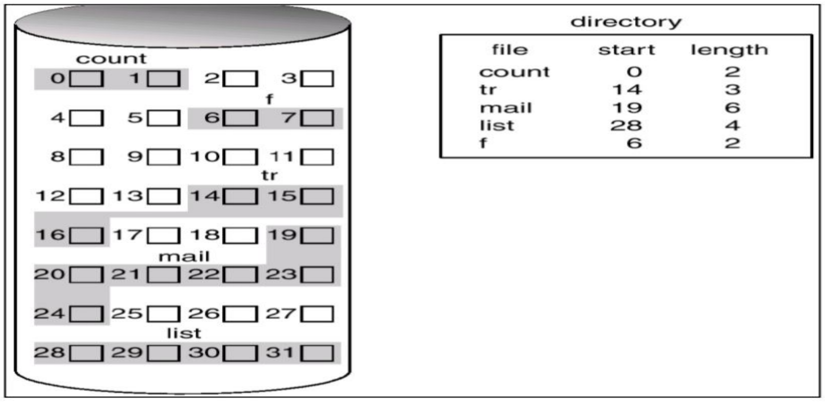
Disadvantages:

- Security problem due to sharing
- Some messages can be lost in the network system
- Bandwidth is another problem if there is large data then all network wires to be replaced which tends to become expensive
- Overloading is another problem in distributed operating systems
- If there is a database connected on local system and many users accessing that database through remote or distributed way, then performance become slow
- The databases in network operating is difficult to administrate then single user system

	d)	**List different file allocation methods. Explain any one in detail**	**4**
	Ans:	**File allocation methods are:** • Contiguous Allocation method • Linked Allocation method • Indexed Allocation method	1 Mark-Listing;(2 Marks-Explanation; 1 Mark-Diagram(any one method)))

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.
- 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 the starting block and the length of the area allocated for this file

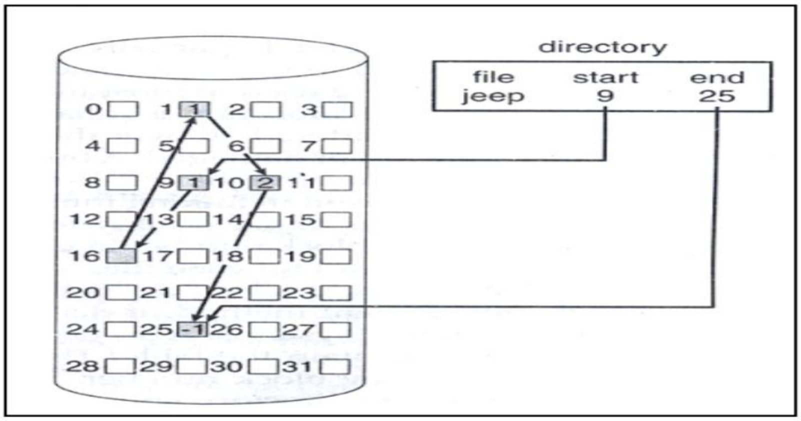


directory

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

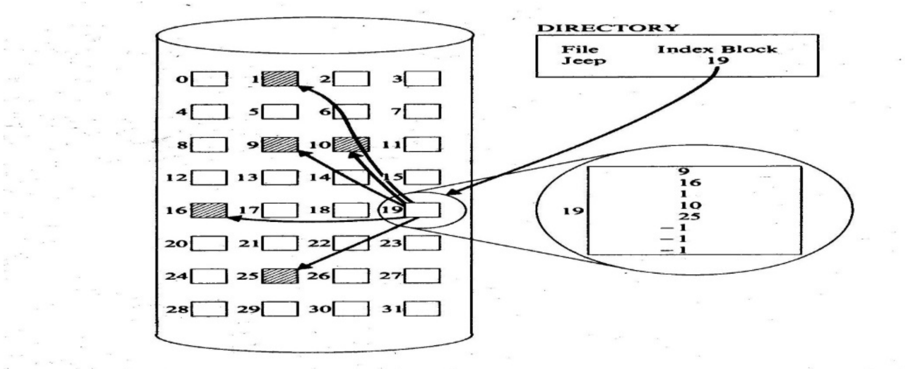
linked Allocation:

- 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 nil 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.



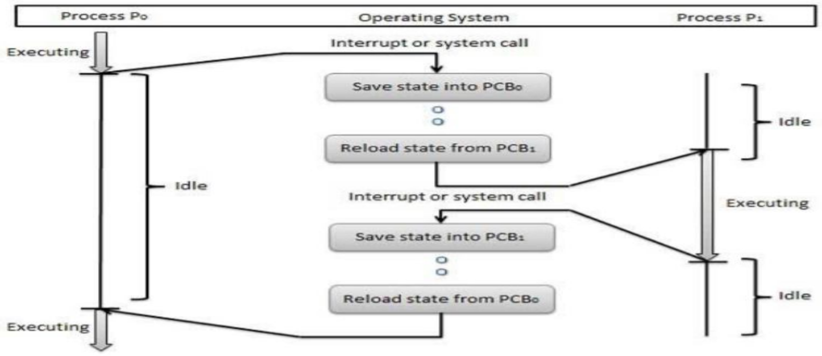
Indexed Allocation:

- In this method, each file has its own index block.
- This index block is an array of disk block addresses.
- When a file is created, an index block and other disk blocks according to the file size are allocated to that file.
- Pointer to each allocated block is stored in the index block of that file.
- Directory entry contains file name and address of index block.
- When any block is allocated to the file, its address is updated in the index block.
- Any free disk block can be allocated to the file. Each ith entry in the index block points to the ith block of the file. To find and read the ith block, we use the pointer in the ith index block entry.



Indexed allocation of disk space

	e)	**Explain context switch with help of diagram**	4
	Ans:	• A context switch is the mechanism to store and restore the state or context of a CPU in Process Control block so that a process execution can be resumed from the same point at a later time. • Using this technique, a context switcher enables multiple processes to share a single CPU. • Context switching is an essential part of a multitasking operating system features.	(2 Marks:- Diagram; 2 Marks:- Explanation)

		• When the scheduler switches the CPU from executing one process to execute another, the context switcher saves the content of all processor registers for the process being removed from the CPU, in its process descriptor. • The context of a process is represented in the process control block of a process. • Context switch time is pure overhead. • Context switching can significantly affect performance as modern computers have a lot of general and status registers to be saved. • Content switching times are highly dependent on hardware support  Process P0 Operating System Process P1 Executing Interrupt or system call Save state into PCB0 Idle Reload state from PCB1 Idle Interrupt or system call Executing Save state into PCB1 Reload state from PCB0 Idle Executing	