MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)
WINTER – 15 EXAMINATION

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

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 judgment 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.

Marks

1. (A) Attempt any THREE of the following: 12

a) List different types of operating systems. Explain advantages of multiprocessor system (any two)

(List any four types – ½ Mark each, any two relevant Advantages - 1 Mark each)

Ans:

Different types of operating systems:

- Mainframe systems
- Multiprocessor systems
- Clustered systems
- Distributed systems
- Real time systems

Advantages of multiprocessor system:

- Less time duration required for the large process.
- Increase throughput.
- Economy of scale.
- Increased reliability.

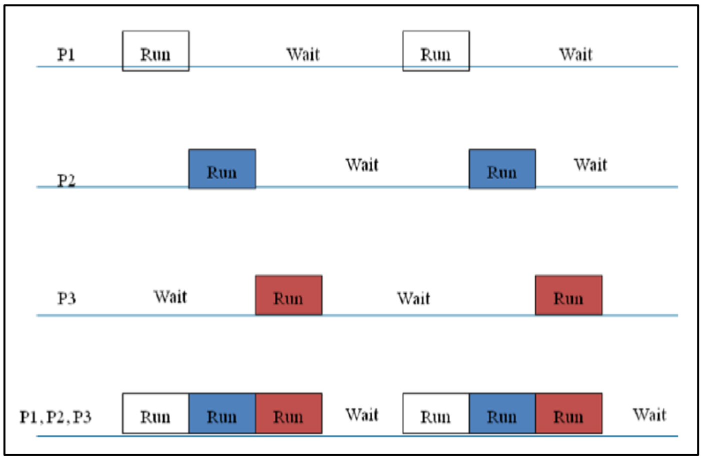
b) Define multiprogramming system with diagram.

(Explanation - 2 Marks, any suitable Diagram - 2 Marks)

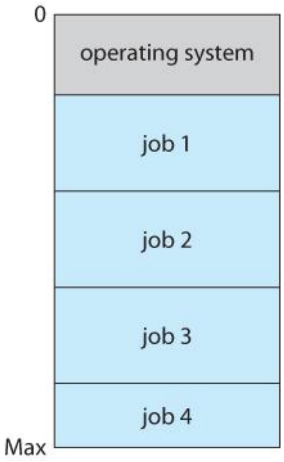
Ans:

In multiprogramming, more than one program lies in the memory i.e. in terms of Operating system, 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 once processor, there can be no simultaneous execution of different programs. Instead the operating system executes part of one program, then the part of another and so on. Multiprogramming is the simple form of parallel processing in which several programs run at the same time on a processor.



OR



Example of multiprogramming, we open word, excel, access and other applications together but while we type in word other applications such as excel and access are just present in main memory but they are not performing any task or work. Or we can say that are not being

c) Draw and explain monolithic structure of operating system.

(Explanation - 2 Marks, Any suitable Diagram - 2 Marks)

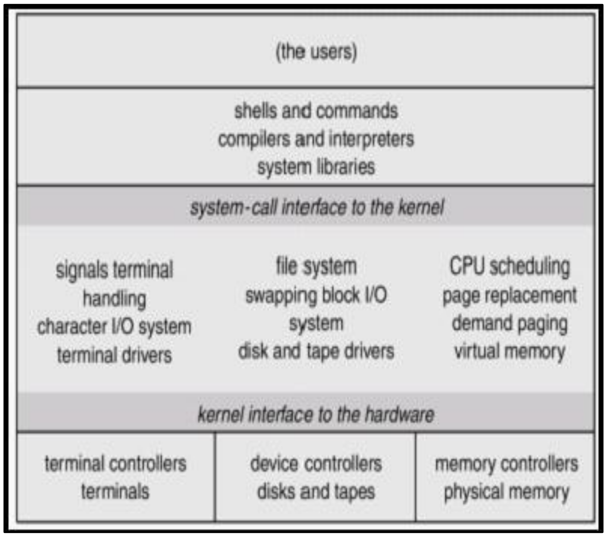
Ans:

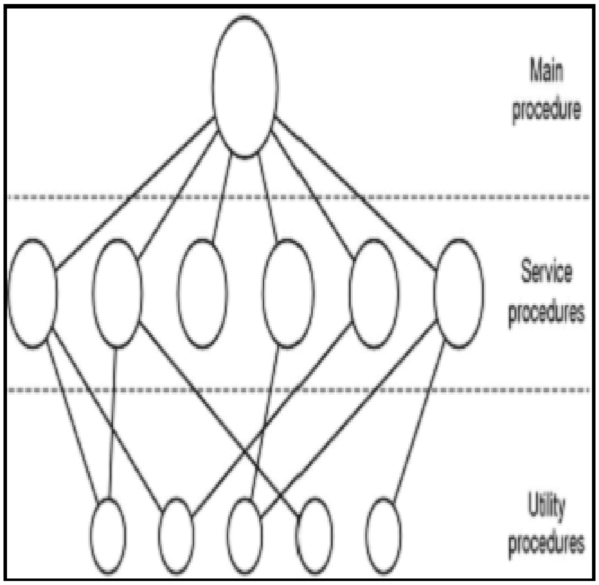
Monolithic Systems:

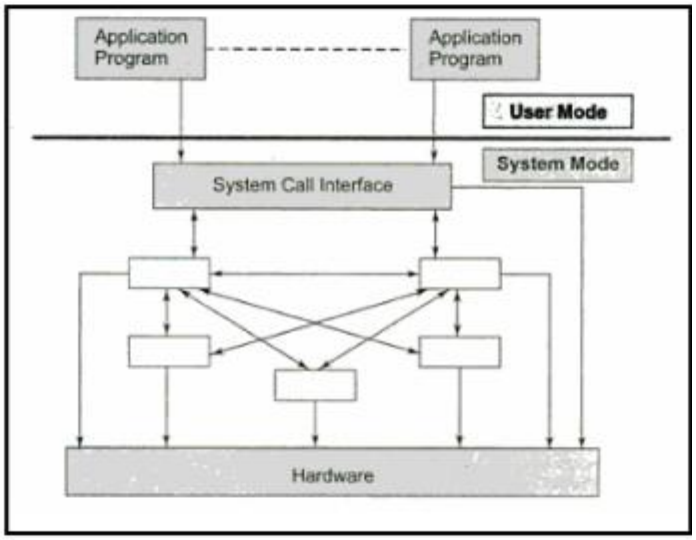
The structure is that there is no structure. The operating system is written as a collection of procedures, each of which can call any of the other ones whenever it needs to. When this technique is used, each procedure in the system has a well-defined interface in terms of parameters and results, and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

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

However, even in Monolithic systems, it is possible to have at least a little structure. The services like system calls provide by the operating system are requested by putting the parameters in well-defined places, such as in registers or on the stack, and then executing a special trap instruction known as a kernel call or supervisor call.







d) Explain two level directory structure with the help of diagram.

(Explanation - 2 Marks, any suitable Diagram - 2 Marks)

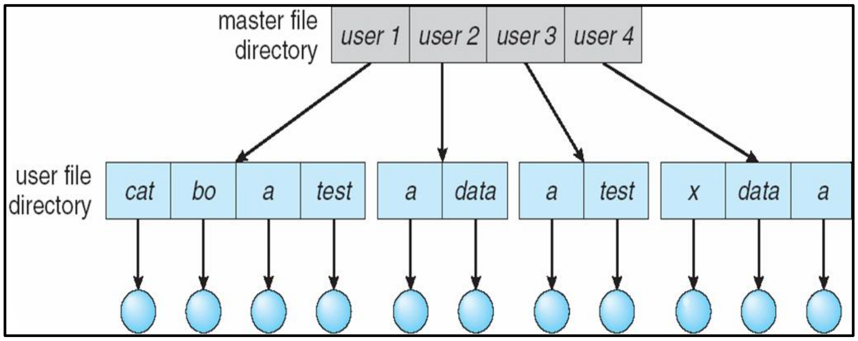
Ans:

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

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

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

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



(B) Attempt any ONE of the following: 6

(a) Explain any six services provided operating system. Draw diagram showing services.

(Explanation of six services – 3 Marks, any relevant Diagram - 3 Marks)

Ans:

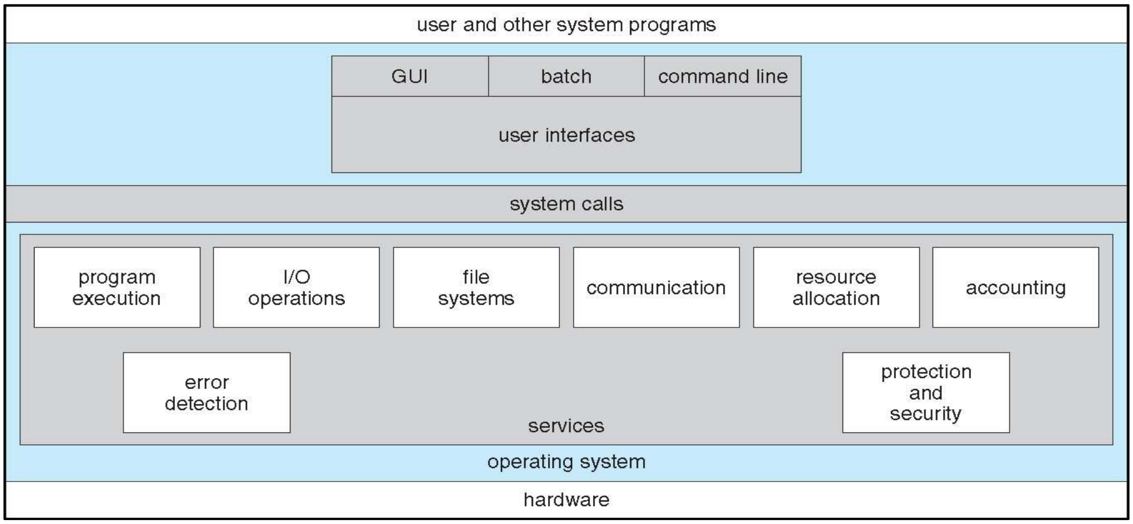
1. **User interface**
2. **Program execution**
3. **I/O operations**
4. **File-system manipulation**
5. **Communications**
6. **Error detection**
7. **Accounting**
8. **Resource allocation**
9. **protection and security**

1. **User interface:** Almost all operating systems have a user interface (UI).The interface can take several forms. One is a DTrace command-line interface(CLI),which uses text commands and a method for entering them (say, a program to allow entering and editing of commands).Another is a batch interface , in which commands and directives to control those commands are entered into files, and those files are executed . Most commonly, a graphical user interface (GUI) is used.
2. **Program execution:** The operating system loads the contents (or sections) of a file into memory and begins its execution. A user-level program could not be trusted to properly allocate CPU time.
3. **I/O operations:** Disks, tapes, serial lines, and other devices must be communicated with at a very low level. The user need only specify the device and the operation to perform on it, while the system converts that request into device- or controller-specific commands. User-level programs cannot be trusted to access only devices they should have access to and to access them only when they are otherwise unused.
4. **File-system manipulation:** There are many details in file creation, deletion, allocation, and naming that users should not have to per-form. Blocks of disk space are used by files and must be tracked. Deleting a file requires removing the name file information and freeing the allocated blocks. Protections must also be checked to assure proper file access. User programs could neither ensure adherence to protection methods nor be trusted to allocate only free blocks and deallocate blocks on file deletion.
5. **Communications:** Message passing between systems requires messages to be turned into packets of information, sent to the net-work controller, transmitted across a communications medium, and reassembled by the destination system. Packet ordering and data correction must

take place. Again, user programs might not coordinate access to the network device, or they might receive packets destined for other processes.

6. **Error detection:** Error detection occurs at both the hardware and software levels. At the hardware level, all data transfers must be inspected to ensure that data have not been corrupted in transit. All data on media must be checked to be sure they have not changed since they were written to the media. At the software level, media must be checked for data consistency; for instance, whether the number of allocated and unallocated blocks of storage match the total number on the device. There, errors are frequently process-independent (for instance, the corruption of data on a disk), so there must be a global program (the operating system) that handles all types of errors. Also, by having errors processed by the operating system, processes need not contain code to catch and correct all the errors possible on a system.
7. **Accounting:** We may want to keep track at which users use how much and what kind of computer resources. What was the login time for a particular user; is he working on the system right now, what is the process -1 D for the user, all such in formations we can manage using accounting service provided by many multiuser systems. This record keeping may be for the purpose of paying for the system & its operation, or simply for accounting usage statistics.
8. **Resource allocation:** When there are multiple users or multiple jobs running at the same time. Resources must be allocated to each of them. Many different types of resources are managed by the operating system. Some (Such as CPU cycles, main memory, and file storage) may have special allocation code, whereas others (such as I/O devices) may have much more general request and release code.
9. **Protection and security:** The owners of information stored in multiuser or networked computer system may want to control use of the information .When several separate processes execute concurrently, it should not be possible for one process to interfere with the others or with the operating system itself, and Protection involves ensuring that all access to system resources is controlled. Security of the system from outsiders is also important. Such security starts with requiring each user to authenticate himself or herself to the system, usually by means of a password, to gain access to system resources. It extends to defending external I/O devices, including modems and network adapters, from invalid access attempts and to recording all such connections for detection of break-ins .If a system is to be protected and secure , precautions must be instituted throughout it.A chain is only as strong as its weakest link.

Diagram for services:



(b) Draw and explain contiguous method for access.

(Explanation - 4 Marks, diagram - 2 Marks)

Ans:

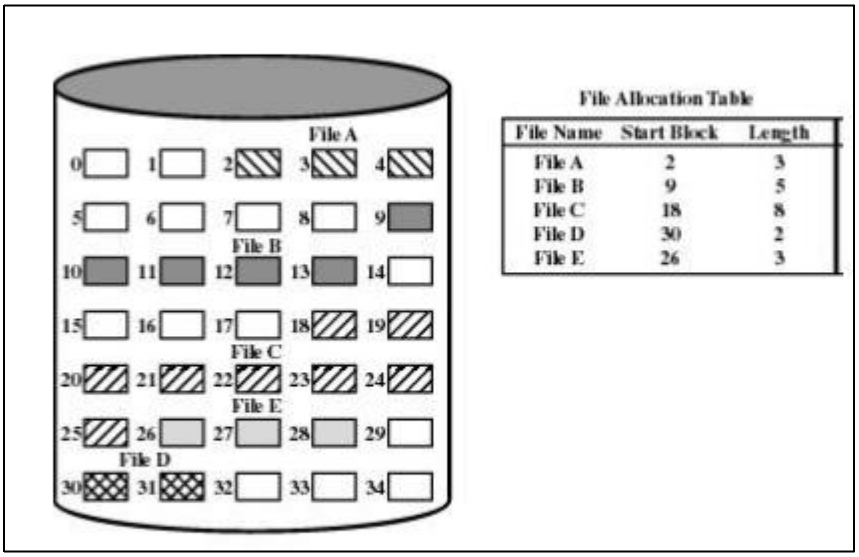
The contiguous allocation method requires each file to occupy a set of contiguous address on the disk. Disk addresses define a linear ordering on the disk. Notice that, with this ordering, accessing block b+1 after block b normally requires no head movement. When head movement is needed (from the last sector of one cylinder to the first sector of the next cylinder), it is only one track. Thus, the number of disk seeks required for accessing contiguous allocated files in minimal, as is seek time when a seek is finally needed. 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.

The difficulty with contiguous allocation is finding space for a new file. If the file to be created is n blocks long, then the OS must search for n free contiguous blocks. First-fit, best-fit, and worst-fit strategies (as discussed in Chapter 4 on multiple partition allocation) are the most common strategies used to select a free hole from the set of available holes. Simulations have shown that both first-fit and best-fit are better than worst-fit in terms of both time storage utilization.

Neither first-fit nor best-fit is clearly best in terms of storage utilization, but first-fit is generally faster.

These algorithms also suffer from external fragmentation. As files are allocated and deleted, the free disk space is broken into little pieces. External fragmentation exists when enough total disk space exists to satisfy a request, but this space not contiguous; storage is fragmented into a large number of small holes.

Another problem with contiguous allocation is determining how much disk space is needed for a file. When the file is created, the total amount of space it will need must be known and allocated. How does the creator (program or person) know the size of the file to be created. In some cases, this determination may be fairly simple (e.g. copying an existing file), but in general the size of an output file may be difficult to estimate.



File Allocation Table

File Name	Start Block	Length
File A	2	3
File B	9	5
File C	18	8
File D	30	2
File E	26	3

Contiguous Allocation Method

2. Attempt any FOUR of the following: 16

(a) What is clustered system? Explain it.

(Definition - 2 Marks, Explanation - 2 Marks)

Ans:

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

will work).Symmetric Cluster (all work at the same level: They work together and monitor each other).

Cluster – is collection of computer in which any member of the cluster is capable of supporting the processing function of any other member. A cluster has n+k configuration where n processing nodes are actively processing the application & k processing nodes are in a standby state, serving as a spares. In the event of a failure of an active node, the application that was running on the failed node is moved to one of the standby nodes.
Other common cluster configuration include Simplex (one active node, no spare), n+1 active node (n active nodes, 1 spare)

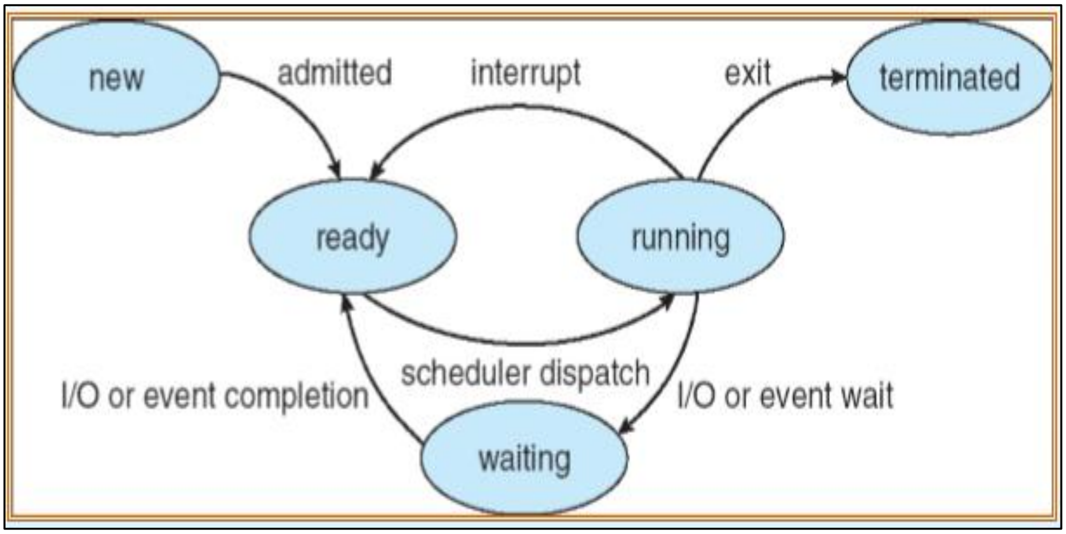
Clustered system can be implemented using LAN.This system is a subsystem of a telecommunication switching system, running certain centralized application function. A LAN is interconnecting hub that provides connectivity with each other in the switching system.

(b) Draw and explain process state diagram

(Process state diagram – 2 Marks, explanation - 2 Marks)

Ans:

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



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

1. Running: has the CPU

2. Blocked: waiting for I/O or another thread

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

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

Ready State: The process which is not waiting for any external event such as I/O operation and which is not running is said to be in ready state. It is not in the running state because some other process is already running. It is waiting for its turn to go to the running state.

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

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

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

(c) Define preemptive and non-preemtive scheduling with suitable example.

(Definition of preemptive and non-preemptive – 1 Mark each, example – 1 Mark each)

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. Throughput is less.
3. It is suitable for RTS.
4. Only the processes having higher priority are scheduled.
5. It doesn't treat all processes as equal.
6. Algorithm design is complex.
7. Circumstances for preemptive
    - process switch from running to ready state
    - process switch from waiting to ready State

Example:

Round Robin, Priority algorithms, SJF (Preemptive)

Non Preemptive Scheduling

1. Once the CPU has been allocated to a process the process keeps the CPU until releases CPU either by terminating or by switching to waiting state
2. Throughput is high.
3. It is not suitable for RTS.
4. Processes having any priority can get scheduled.
5. It treats all process as equal.
6. Algorithm design is simple.
7. Circumstances for Non preemptive
    - Process switches from running to waiting state
    - Process terminates

Example:

FCFS algorithm
SJF (Non preemptive)

(d) What is virtual memory? Explain paging and page fault

(Description of virtual memory - 2 Marks, Paging - 1 Mark, Page fault - 1 Mark)

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.

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.

Paging:

Paging refers to the transfer of memory pages from physical memory to disk and vice versa.

Virtual memory uses a technique called demand paging for its implementation.

Logical address space of a process can be noncontiguous; process is allocated physical memory whenever the latter is available

Divide physical memory into fixed-sized blocks called **frames** (size is power of 2, between 512 bytes and 8192 bytes)

Divide logical memory into blocks of same size called **pages**.

- Address generated by CPU divided into:
  - *Page number (p)* - index to page table
    - + *page table* contains base address of each page in physical memory (frame)
  - *Page offset (d)* - offset into page/frame

Page fault:

When the process executes and accessed the pages that are present into memory, execution proceeds normally. But if the process tries to access a page which is marked invalid, then it causes a page fault trap. This trap occurs because operating system has failed to bring the desired page into memory. The main functions of paging are performed when a program tries to access pages that do not currently reside in the RAM. This situation is known as page fault. The OS must take control and handle the page fault.

(e) Explain the structure of UNIX.

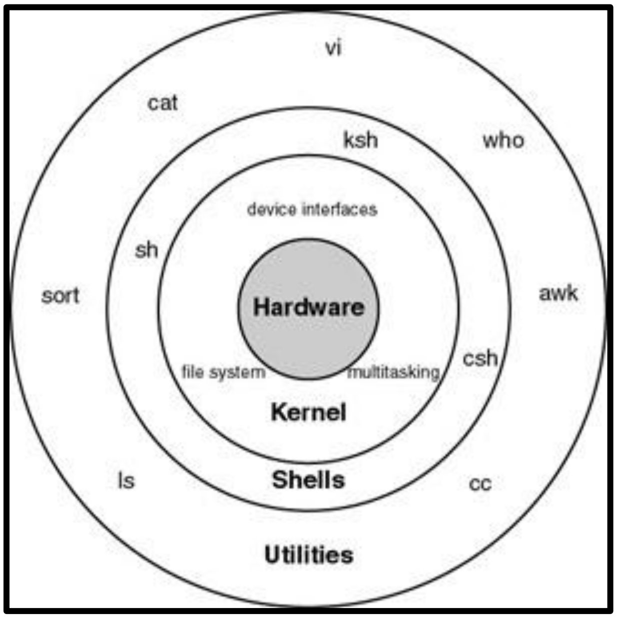
(Explanation - 3 Marks, Diagram - 1 Mark)

Ans:

STRUCTURE OF UNIX

- **Hardware** of the system
- The **kernel**, which schedules tasks and manages storage;
- The **shell**, which connects and interprets users' commands, calls programs from Memory, and executes them.
- The **tools and applications** that offer additional functionality to the OS

The kernel is the heart of the system, a collection of programs written in C that directly communicate with the hardware. There is only one kernel for any system. It's that part of UNIX system that is loaded into memory when the system is booted. It manages the System resources, allocates time between user and processes, decides process priorities, and performs all other tasks. The kernel, in traditional parlance, is often called the Operating system. The shell, on the other hand, is the "sleeping beauty" of UNIX. It is actually the interface between the user and the kernel. The shell is the agency which takes care of the features of redirection and has a programming capability of its own.The Tools and Applications consist of Application Software, Compilers, Database Package, Internet tools, UNIX commands, etc.


vi
cat
ksh
who
device interfaces
sh
sort
Hardware
awk
csh
file system
multitasking
Kernel
Shells
ls
cc
Utilities

(f) Compare UNIX and LINUX w.r.t. following points: User interface, number of shells, providers, processing speed.

(Comparison - 1 Mark to each point)

Ans:

CRITERIA	LINUX	UNIX
User interface	Linux typically provides two GUIs, KDE and Gnome. But there are millions of alternatives such as LXDE, Xfce, Unity, Mate, twm, ect.	Initially Unix was a command based OS, but later a GUI was created called Common Desktop Environment. Most distributions now ship with Gnome.
Number of shells	Sh, bash, csh and tsh, ksh	B, C, K, Bash, tcsh, zsh
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.

3. Attempt any FOUR of the following: 16

(a) Describe any four activities of process management and memory management.

(Any four activities of Process Management – ½ Mark each, any four activities of Memory Management – ½ Mark each)

Ans:

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

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

Memory Management:

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

(b) Explain interprocess communication.

(Any relevant Explanation - 4 Marks)

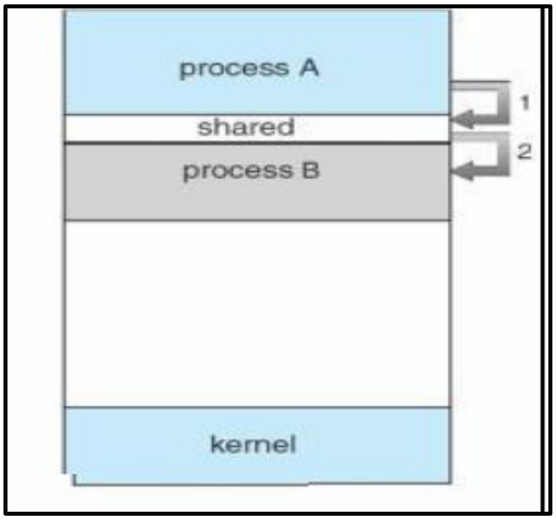
[Note: Explanation of interprocess communication with models or without models shall be considered.]**

Ans:

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

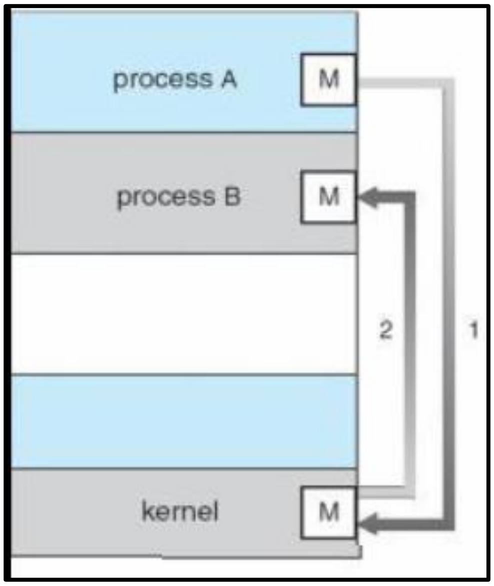
There are two models of IPC

1. **Shared memory**



In this a region of the memory residing in an address space of a process creating a shared memory segment can be accessed by all processes who want to communicate with other processes. All the processes using the shared memory segment should attach to the address space of the shared memory. All the processes can exchange information by reading and/or writing data in shared memory segment. The form of data and location are determined by these processes who want to communicate with each other. These processes are not under the control of the operating system. The processes are also responsible for ensuring that they are not writing to the same location simultaneously. After establishing shared memory segment, all accesses to the shared memory segment are treated as routine memory access and without assistance of kernel.

2. Message Passing



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

(c) Write steps for Banker's algorithm to avoid deadlock.

(Correct steps for algorithm - 4 Marks)

Ans:

Banker's Algorithm

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

Algorithm F: Free resources

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

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

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

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

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

Step 6: After execution of a process, it removes the row 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.

(d) State necessary condition for Deadlock.

(Four conditions - 1 Mark each)

Ans:

1. Mutual Exclusion: The resources involved are non-shareable.

At least one resource (thread) must be held in a non-shareable mode, that is, only one process at a time claims exclusive control of the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.

2. Hold and Wait: Requesting process hold already, resources while waiting for requested resources.

There must exist a process that is holding a resource already allocated to it while waiting for additional resource that are currently being held by other processes.

3. No-Preemptive: Resources already allocated to a process cannot be preempted.

Resources cannot be removed from the processes are used to completion or released voluntarily by the process holding it.

4. Circular Wait: The 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.

(e) List different types of files. Explain basic operations on file.

(Any 4 types – ½ Mark each, any 4 operations – ½ Mark each)

Ans:

Different types of files

file type	**usual extension**	**function**
executable	exe,com,bin Or none	ready-to-run machine-language program
object	Obj,o	compiled, machine language, not linked
source code	c,cc,java,pas ,asm,a	Source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt ,doc	textual data, documents

word processor	wp, tex, rtf, doc	various word-processor Formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf,jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg,mov,rm,mp3,avi	binary file containing audio or A\V information

Common file types

File Operations

Basic file operations are

1. **Creating a file**. Two steps are necessary to create a file.
1. Space in the file system must be found for the file.
2. An entry for the new file must be made in the directory.

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

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

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

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

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

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

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

4. (A) Attempt any THREE of the following: 12

(a) List any four system calls for device management and communication.

(Any four System calls Related to Device Management – ½ Mark each, any four System calls Related to Communication – ½ Mark each)

Ans:

System calls Related to Device Management:

- Request a device
- Release a device
- Read, Write, Reposition
- Get device attribute
- Set device attribute

System calls Related to Communication:

- create ,delete communication connection
- send, receive messages
- transfer status information
- attach or detach remote devices

(b) Describe any four secondary storage management activities.

(Any relevant four activities -1 Mark each)

Ans:

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.

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

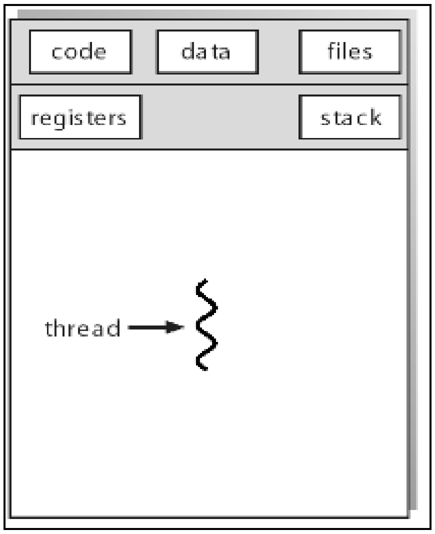
1. Managing the free space available on the secondary-storage device.
2. Allocation of storage space when new files have to be written.
3. Scheduling the requests for memory access.
4. Deallocation of storage space when required.

(c) What is thread? Explain users and Kernal threads.

(Description of thread - 2 Marks, Explanation of user thread - 1 Mark, Explanation of Kernel thread - 1 Mark)

Ans:

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



User-Level Threads: -

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

Kernel Threads: -

- In 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.

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

(State types of scheduler - 1 Mark, Description of three types - 1 Mark each)

[**Note: - Any relevant description about schedules shall be considered]

Ans:

Schedulers are of three types:-

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

Long Term Scheduler

It is also called job scheduler. Long term scheduler determines which programs are admitted to the system for processing. Job scheduler selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling. The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system. On some systems, the long term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When process changes the state from new to ready, then there is use of long term scheduler.

Short Term Scheduler

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

Medium Term Scheduler

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

(B) Attempt any ONE of the following: 06

(a) What is the process? Explain the different process states with diagram.

(Definition of process - 2 Marks, Explanation of process state - 2 Marks, Diagram of process state - 2 Marks)

Ans:

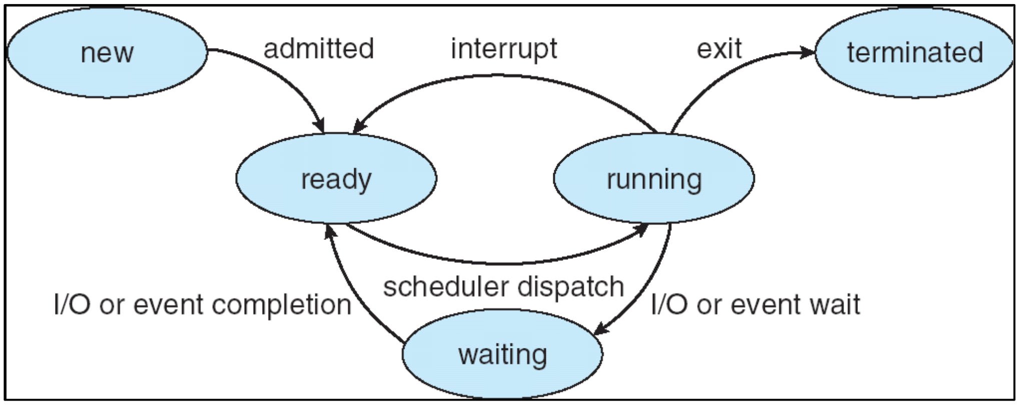
Process is a program in execution. A process does not mean only program but it could contain some part called as text section. It may contain the current activity represented by the value of the program counter & the contents of CPU register.

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

1. Running: has the CPU

2. Blocked: waiting for I/O or another thread

3. Ready to run: on the ready list, waiting for the CPU



During the lifespan of a process, its execution status may be in one of four states: (associated with each state is usually a queue on which the process resides)

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

Ready State: The process which is not waiting for any external event such as I/O operation and which is not running is said to be in ready state. It is not in the running state because some other process is already running. It is waiting for its turn to go to the running state.

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

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

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

(b) With suitable diagram, explain how linked allocation is performed.

(Diagram - 2 Marks, Explanation – 4 Marks)

Ans:

The problems in contiguous allocation can be traced directly to the requirement that the spaces be allocated contiguously and that the files that need these spaces are of different sizes. These requirements can be avoided by using linked allocation.

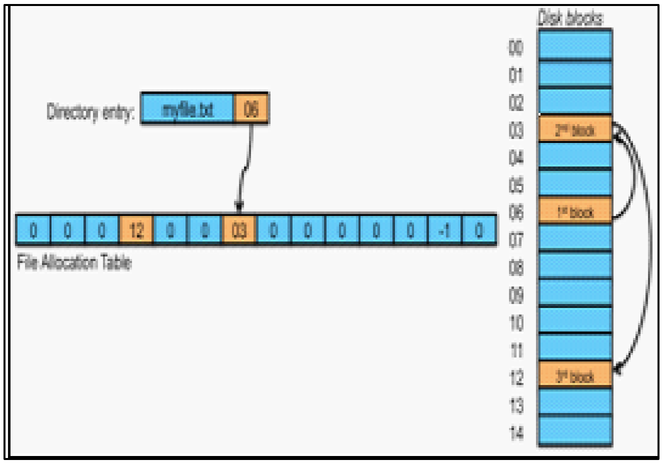
In linked allocation, each file is a linked list of disk blocks. The directory contains a pointer to the first and (optionally the last) block of the file. For example, a file of 5 blocks which starts at block 4, might continue at block 7, then block 16, block 10, and finally block 27. Each block contains a pointer to the next block and the last block contains a NIL pointer. The value -1 may be used for NIL to differentiate it from block 0.

With linked allocation, each directory entry has a pointer to the first disk block of the file. This pointer is initialized to nil (the end-of-list pointer value) to signify an empty file. A write to a file removes the first free block and writes to that block. This new block is then linked to the end of the file. To read a file, the pointers are just followed from block to block.

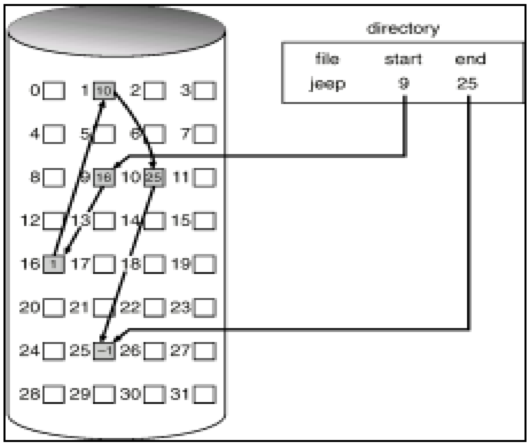
There is no external fragmentation with linked allocation. Any free block can be used to satisfy a request. Notice also that there is no need to declare the size of a file when that file is created. A file can continue to grow as long as there are free blocks.

Linked allocation, does have disadvantages, however. The major problem is that it is inefficient to support direct-access; it is effective only for sequential-access files. To find the ith block of a file, it must start at the beginning of that file and follow the pointers until the ith block is reached. Note that each access to a pointer requires a disk read.

Another severe problem is reliability. A bug in OS or disk hardware failure might result in pointers being lost and damaged. The effect of which could be picking up a wrong pointer and linking it to a free block or into another file.



OR



5. Attempt any TWO of the following: 16

(a) Describe many to one and one to many multithreading model with diagram. Explain advantages of each (any two).

(Two model: Description of each model with diagram – 2 Marks, any two advantages of each model– 1 Mark each)

[**Note- for one to many multithreading model, one to one model, many to many model or any other relevant multithreading description shall be considered]

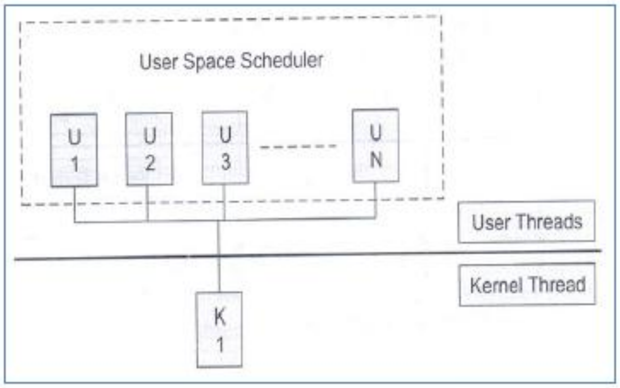
Ans:

1. Many to One model:

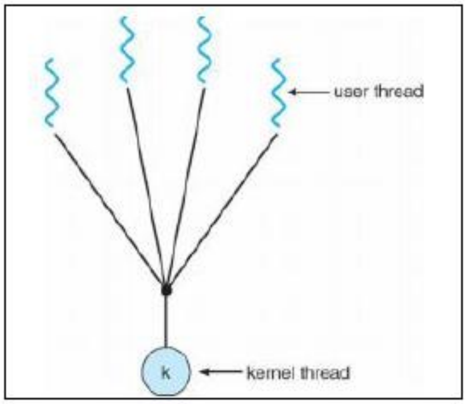
This model maps many user level threads to one kernel level thread. Thread management is done by thread library in user space.

Advantages:-

1. It is an efficient model as threads are managed by thread library in user space.
2. Portable: Because user level threads packages are implemented entirely with standard Unix and POSIX library calls, they are often quite portable



OR

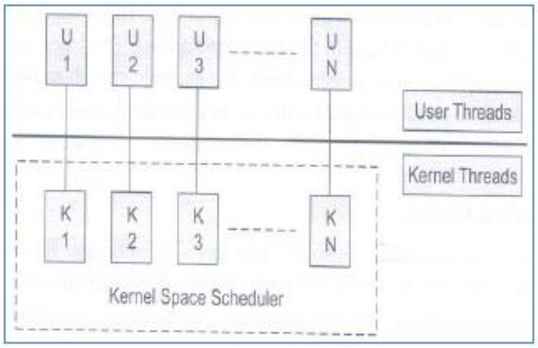


2. One to One model:

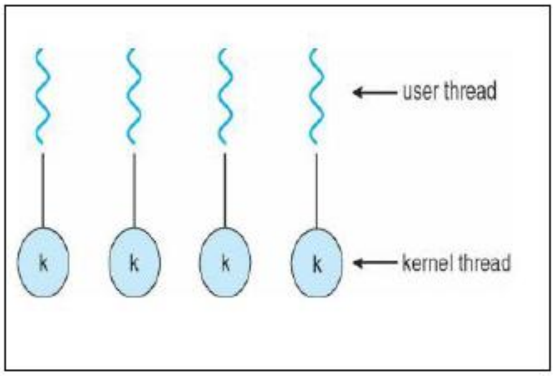
It maps each user thread to a kernel thread. Even a thread makes a blocking call;
Other thread can run with the kernel thread.

Advantages:-

1. It allows multiple threads to run in parallel on multiprocessors.
2. Multithreaded OS interface: when one user thread and its kernel thread block, the other user threads can continue to execute since their kernel threads are unaffected



OR



3. Many to Many model :

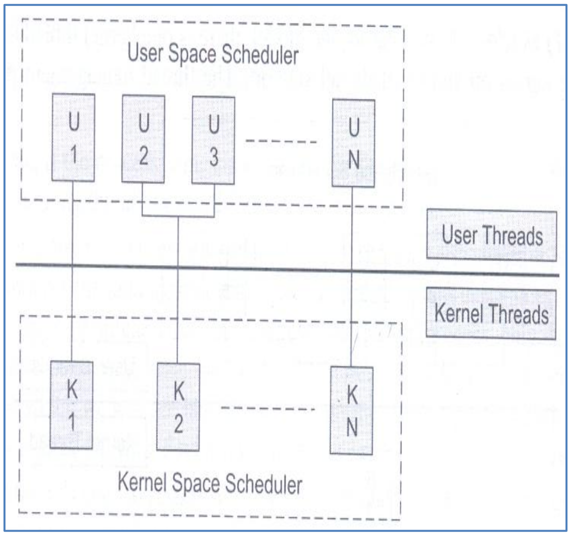
Allows many user level threads to be mapped to many kernel threads.

Allows the operating system to create a sufficient number of kernel threads

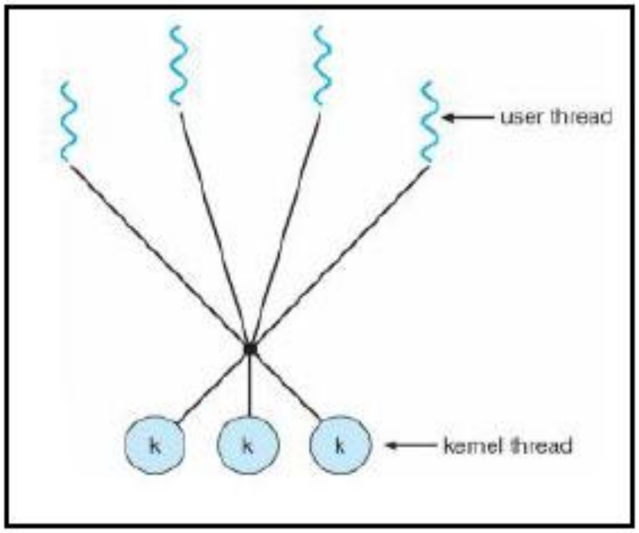
E. g Solaris 2 and Windows NT/2000 with the *Thread Fiber* package

Advantages:-

1. Threads can run in parallel or multiprocessor
2. When a thread performs blocking system call, the kernel can schedule anther tread for execution.
3. Developer can create as many user threads as required



OR



(b) Calculate average waiting time for FCFS and SJF for following table:

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

(For each scheduling – Gantt chart – 1 Mark, waiting time calculation - 2 Marks, average waiting time – 1 Mark)

Ans:

FCFS

Gantt chart

P1	P2	P3	P4

0 8 12 21 26

Waiting time

P1=0

P2=(8-1)=7

P3=(12-2)=10

P4= (21-3)=18

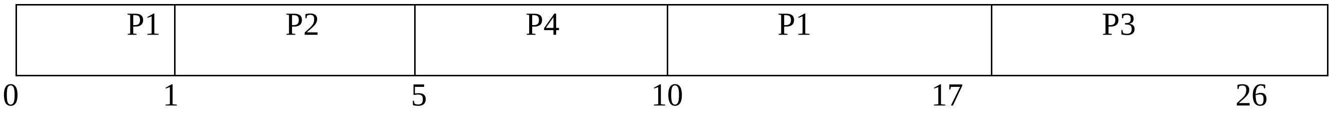
Average waiting time=waiting time of all processes/number of processes

=waiting time of (p1+p2+p3+p4)/4

=0+7+10+18/4

=8.75 milli seconds (ms)

SJF



Waiting time

P1=0+(10-1)=9

P2=0

P3=(17-2)=15

P4= (5-3)=2

Average waiting time=waiting time of all processes/number of processes

=waiting time of (p1+p2+p3+p4)/4

=9+0+15+2/4

=6.5 milli seconds (ms)

(c) **Explain how UNIX is differ from LINUX w.r.t. architecture, applications, case of operation and system requirement.**

(Difference - 2 Marks to each point)

[**Note Any relevant explanation shall be consider]

Ans:

	UNIX	LINUX
Architecture	Small scale (is available on PA-RISC and Itanium machines. Solaris also available for x86/x64 based systems.)	Highconfiguration(Originally developed for Intel's x86 hardware, ports available for over two dozen CPU types including ARM)
Applications	Command base (The UNIX operating system is used in internet servers, workstations & PCs. Backbone of the majority of finance 32 infrastructure and many 24x365 high availability solutions. GUI: Initially Unix was a command based OS, but later a GUI was created called Common Desktop Environment. Most distributions now ship with Gnome.)	GUI based(Linux can be installed on a wide variety of computer hardware, ranging from mobile phones, tablet computers and video game consoles, to mainframes and supercomputers. GUI: Linux typically provides two GUIs, KDE and Gnome. But Linux GUI is optional.)
Case Of Operation	It requires Less no. of Kernal for operation.	It requires Multiple kernel for operation
System Requirement (any version shall be consider)	Hardware on UNIX your computer must have the following minimum configuration to successfully install WebLogic integration – Business Connect on a unix platform. • Random Access Memory(RAM) • 256MB recommended • 128MB minimum • 250MB available hard drive space (see note) • CD-ROM drive • TCP/IP network interface • A persistent Internet connection	• operating system- Red Hat Enterprise Linux 4 or 5 with the latest patches and upgrades • CPU Types –Pentium4 or higher; to GHz or higher • Memory/RAM – 1GB minimum, up to the system limit • Hard Disk – 4GB minimum • Other - to read the Directory Server using port number lass then 1024, such as the default you can port 389, you can set up and start the directory server as root, but it is not necessary to run the Directory Server root.

6. Attempt any FOUR of the following: 16

(a) What is real time system? Explain its types.

(Description of real time system – 2 Marks, Explanation of two types – 1 Mark each)

Ans:

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

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

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

Types of real time system:

1. **Hard real time**:-

Hard real time means strict about adherence to each task deadline. When an event occurs, it should be serviced within the predictable time at all times in a given hard real time system.

Example: - video transmission, each picture frame and audio must be transferred at fixed rate.

2. **Soft real time**:-

Soft real time means that only the precedence and sequence for the task operations are defined, interrupt latencies and context switching latencies are small. There can be few deviations between expected latencies of the tasks and observed time constraints and a few deadline misses are accepted.

Example:-Mobile phone, digital cameras and orchestra playing robots.

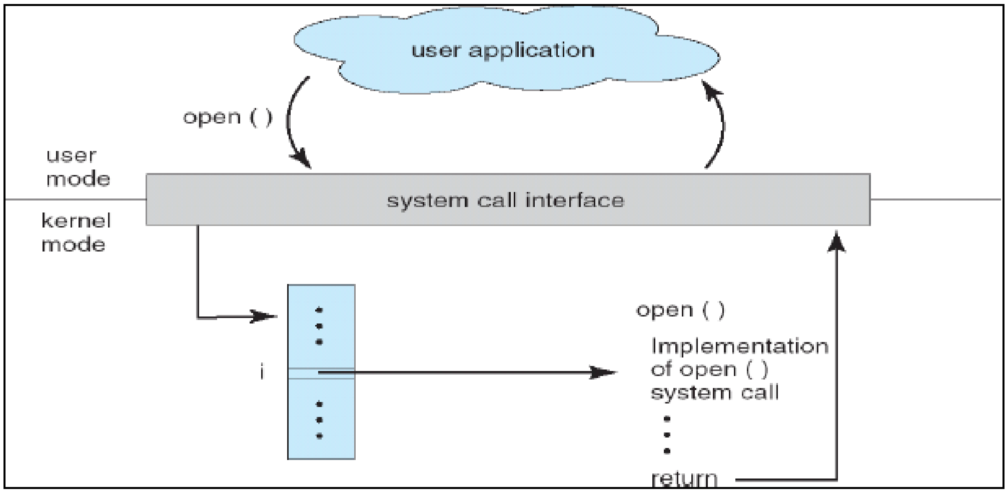
(b) **What is system call? Explain open ( ) system call and close ( ) system call.**

(Description of system call – 2 Marks, open ( ) system call -1 Mark, close ( ) system call -1 Mark)

Ans:

System Calls: System calls are programming interface to the services provided by the operating system

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.
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.



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.

Close ( ) system call

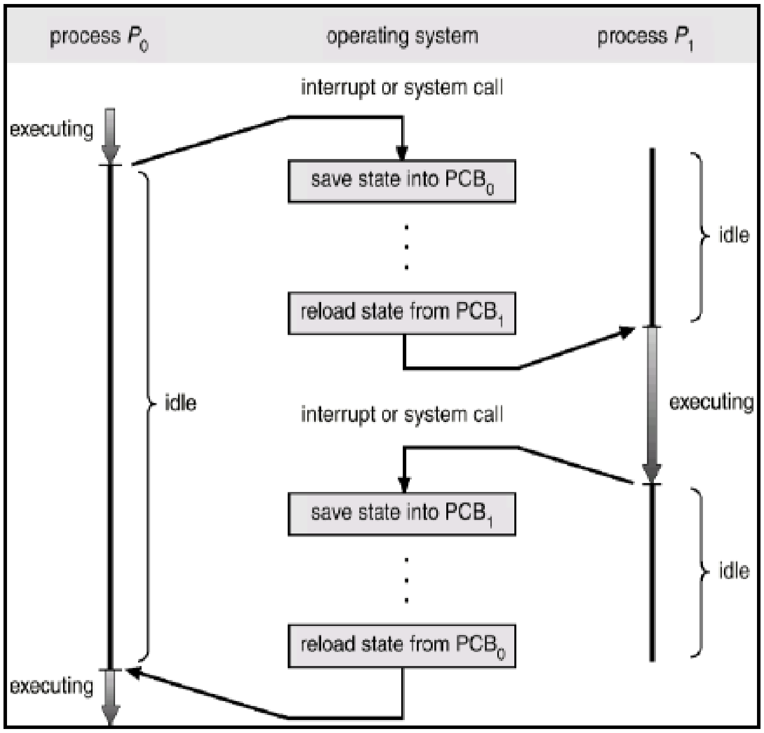
For most file systems, a program terminates access to a file in a file system using the close system call. This flushes buffers, updates file metadata (which may include and end of file indicator in the data), de-allocates resources associated with the file (including the file descriptor) and updates the system wide table of files in use.

(c) Explain the concept of context switching.

(Description - 3 Marks, Diagram – 1 Mark)

Ans:

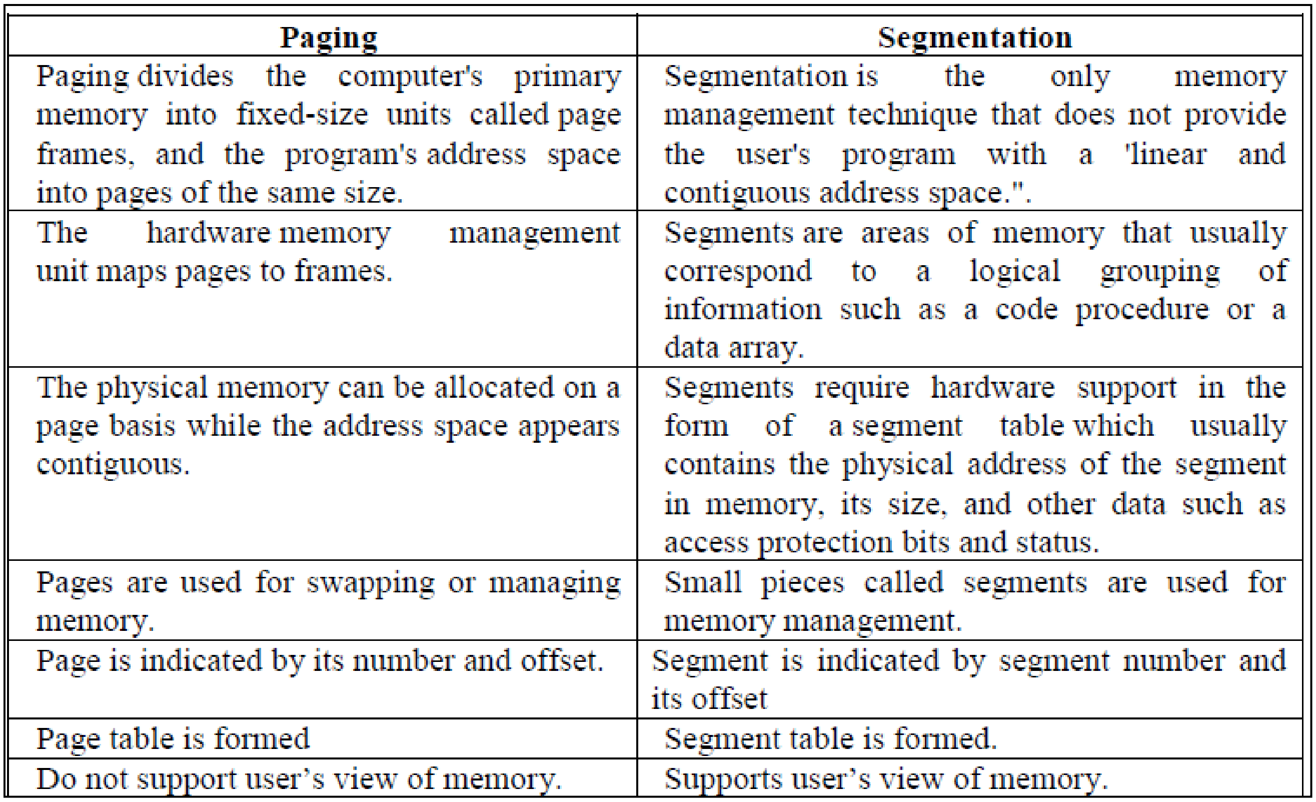
Switching the CPU to another process requires saving the state of current process and loading the saved state for new process. This process is known as a context switch. The context switch is represented in PCB. Saves context of old process in its PCB and loads context of new process into the memory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



(d) Differentiate between paging and segmentation.

(Any four points - 1 Mark each)
*[**Note: any other valid point shall be considered]*

Ans:



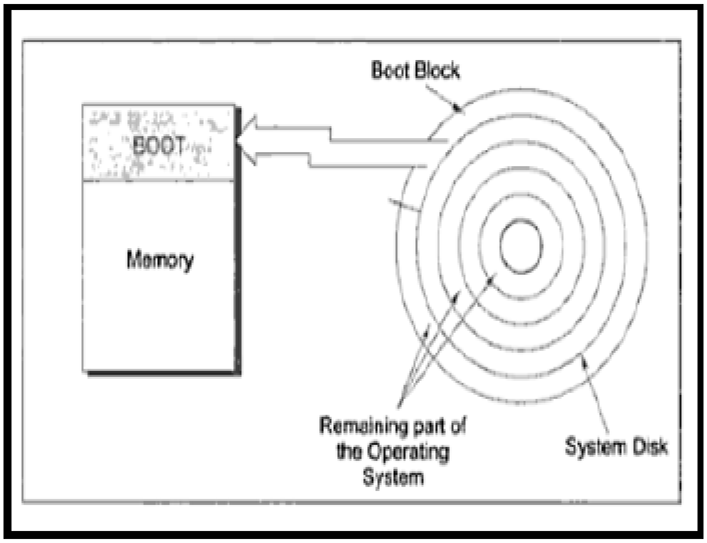
Paging	Segmentation
Paging divides the computer's primary memory into fixed-size units called page frames, and the program's address space into pages of the same size.	Segmentation is the only memory management technique that does not provide the user's program with a 'linear and contiguous address space.".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

(e) Explain booting system of UNIX.

(Explanation – 2 Marks, Diagram – 2 Marks)

Ans:

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


Boot Block
BOOT
Memory
Remaining part of the Operating System
System Disk