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.

1. Attempt any FIVE of the following : Marks 20

a) Describe the multiprocessor systems concepts.

(Explanation - 4 Marks)

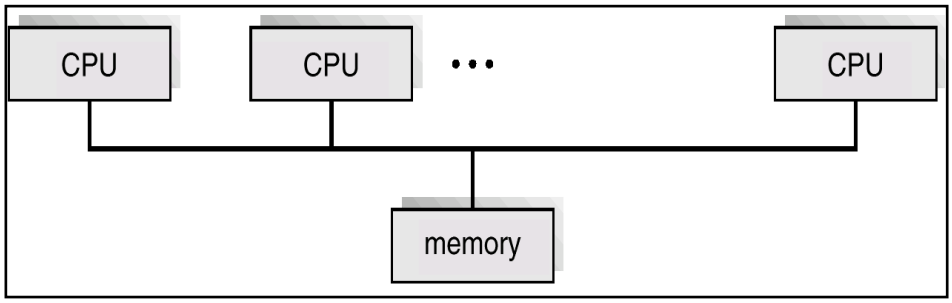
Ans: Multiprocessor Systems:

- Multiprocessor systems with more than on CPU in close communication.
- Tightly coupled system – processors share memory and a clock; communication usually takes place through the shared memory.
- Advantages of Multiprocessor system:
  - Increased throughput
  - Economical
  - Increased reliability
  - graceful degradation
  - fail-soft systems

Symmetric multiprocessing (SMP)

- Each processor runs and identical copy of the operating system.
- Many processes can run at once without performance deterioration.
- Most modern operating systems support SMP

Symmetric Multiprocessing Architecture



Asymmetric multiprocessing

- Each processor is assigned a specific task; master processor schedules and allocated work to slave processors.
- More common in extremely large systems.

b) Describe monolithic operating system structure.

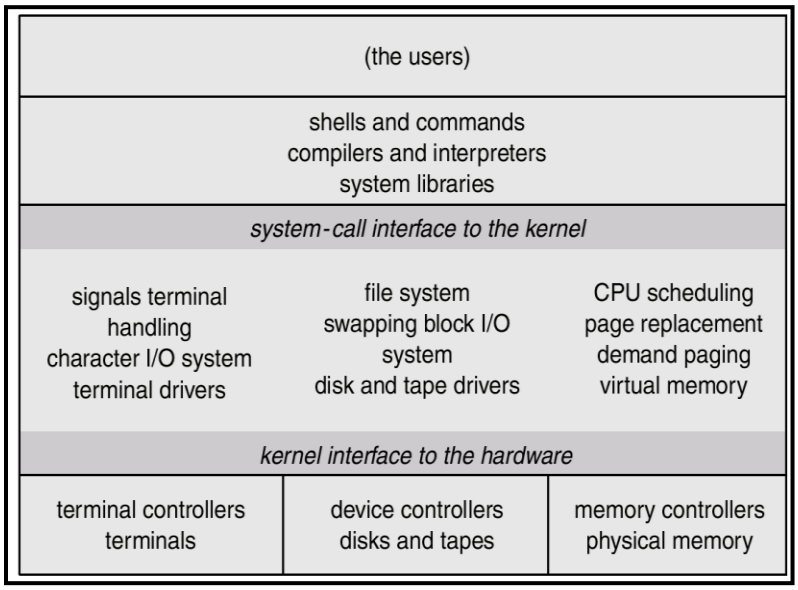
(Explanation- 3 Marks, Diagram-1 Mark)

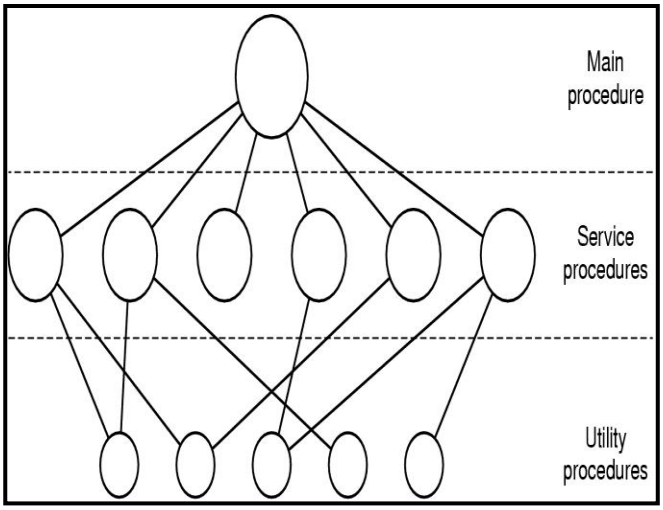
Ans: **Monolithic Systems**: This approach well known as "The Big Mess". The structure is that there is no structure. The operating system is written as a collection of procedures, each of which can call any of the other ones whenever it needs to. When this technique is used, each procedure in the system 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.

Simple structuring model for a monolithic system.





c) With neat diagram explain process control block.

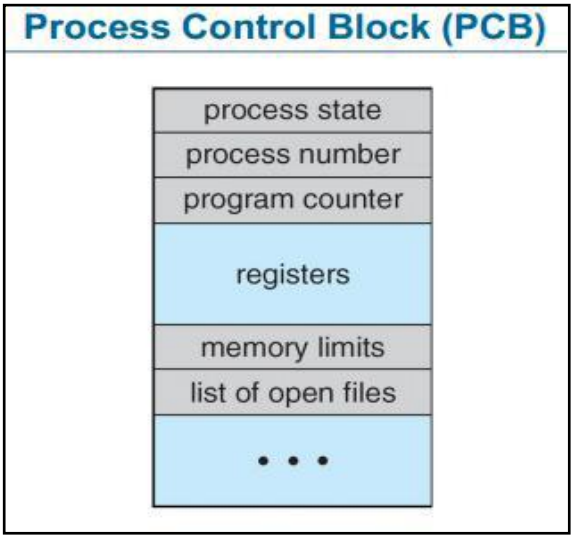
(Explanation-2 Marks, Diagram-2 Marks)

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

A PCB may contain several types of information depending upon the process to which PCB belongs. The information stored in PCB of any process may vary from process to process.

In general, a PCB may contain information regarding:

1. **Process Number**: Each process is identified by its process number, called process identification number (PID). Every process have a unique process-id through which it is identified. The process-id is provided by the OS. The process id of two process could not be same because ps-id is always unique.
2. **Priority:** Each process is assigned a certain level of priority that corresponds to the relative importance of the event that it services Process priority is the preference of the one process over other process for execution. Priority may be given by the user/system manager or it may be given internally by OS. This field stores the priority of a particular process.
3. **Process State:** This information is about the current state of the process. I.e. whether process is in new, ready, running, waiting or terminated state.
4. **Program Counter:** This contains the address of the next instruction to be executed for this process.
5. **CPU Registers:** CPU registers vary in number and type, depending upon the computer architectures. These include index registers, stack pointers and general purpose registers etc. When an interrupt occurred, information about the current status of the old process is saved in registers along with the program counters. This information is necessary to allow the process to be continued correctly after the completion of an interrupted process.
6. **CPU Scheduling Information**: This information includes a process priority, pointers to scheduling queues and any other scheduling parameters.
7. **Memory Management Information**: This information may include such information as the value of base and limit registers, the page table or the segment table depending upon the memory system used by operating system.
8. **Accounting**: This includes actual CPU time used in executing a process in order to charge individual user for processor time.
9. **I/O Status**: It includes outstanding I/O request, allocated devices information, pending operation and so on.
10. **File Management:** It includes information about all open files, access rights etc.



d) Describe CPU and I/O burst cycle.

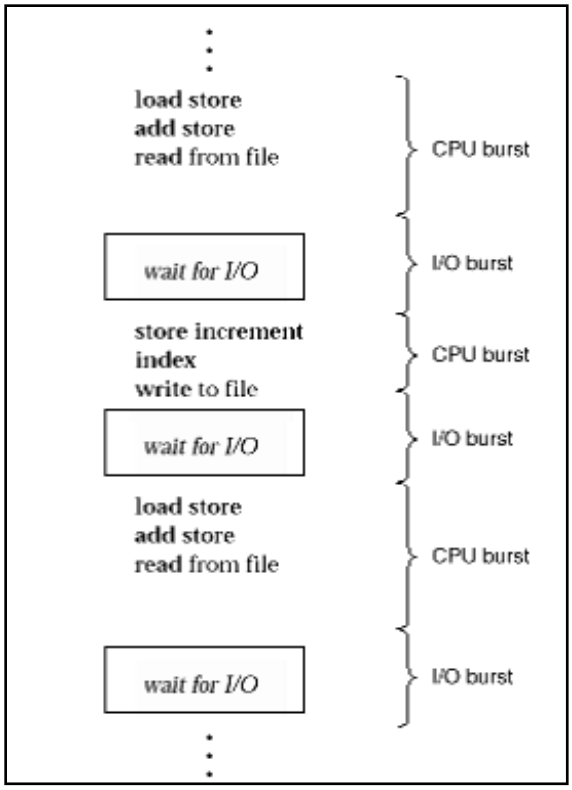
(Explanation- 2 Marks, Diagram -2 Marks)

Ans: I/O bound process:

The process which spends more time in I/O operation than computation (time spends with CPU) is I/O bound process.

CPU bound process:

The process which spends more time in computations or with CPU and very rarely with the I/O devices is called as CPU bound process.



e) State the benefits of multithreading.

(Any 4 points - 1 Mark each)

Ans: Threads benefits

The benefits of multithreaded programming can be broken down into four major categories:

1. 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 example: A multithreaded web browser could still allow user interaction in one thread while an image is being loaded in another thread. A multithreaded Web server with a front-end and (thread) processing modules.

2. Resource sharing: By default, threads share the memory and the resources of the process to which they belong. The benefit of code sharing is that it allows an application to have several different threads of activity all within the same address space.

A word processor with three threads.

For example: A multithreaded word processor allows all threads to have access to the document being edited.

3. Economy: Because threads share resources of the process to which they belong, it is more economical to create and switch threads, than create and context switch processes (it is much more time consuming).

For example: in Sun OS Solaris 2 creating a process is about 30 times slower than is creating a thread (context switching is about five times slower than threads switching).

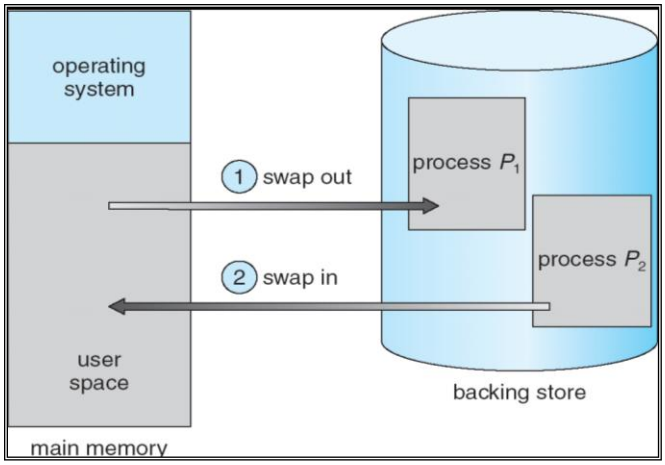
4. Utilization of multiprocessor architectures: The benefits of multithreading can be greatly increased in a multiprocessor architecture (or even in a single-CPU architecture), where each thread may be running in parallel on a different processor.

f) What is swapping and when it is used.

(Explanation- 3 Marks, Diagram- 1 Mark)

Ans: Swapping

- Swapping is a simple memory/process management technique used by the operating system (os) to increase the utilization of the processor by moving some blocked process from the main memory to the secondary memory (hard disk); thus forming a queue of temporarily suspended process and the execution continues with the newly arrived process. After performing the swapping process, the operating system has two options in selecting a process for execution.
- Assume a multiprogramming environment with Round-Robin CPU scheduling algorithm. When a quantum expires memory manager will start to swap out the process that just finished, and swap in another price to the memory space that has been freed. In the meantime, CPU scheduler will allocate a time slice to some other process in memory. When each process finishes its quantum it will be swapped back with another process. Ideally, memory manager can swap process fast enough so that there are always process in memory, ready to execute, when CPU scheduler wants to reschedule the CPU. The quantum must also be sufficiently large that reasonable amounts of computing are done between swaps.
- Swapping can be implemented in various ways. For example, swapping can be priority based.



Swapping of two processes using a disk as a backing store.

g) Explain the concept of mutual exclusion in detail.

(Explanation - 4 Marks)

Ans: Mutual exclusion is one of the four conditions that should hold simultaneously in a system for a deadlock to occur.

Mutual Exclusion: At least one resource must be held in a non-sharable mode that is only one process at a time can use the resource. If another process requests that resource the requesting process must be delayed until the resource has been released.

Mutual Exclusion can be used for deadlock prevention:

The mutual exclusion condition must hold for non sharable resources. For example a printer cannot be simultaneously shared by several processes sharable resources on the other hand do not require mutually exclusive access and thus cannot be involved in a deadlock. Read only files are good example of a sharable resource. If several processes attempt to open a read only file at the same time they can be granted simultaneous access to the file. a process never needs to wait for a sharable resource. in general however it is not possible to prevent deadlock by denying the mutual exclusion condition some resources are intersically non-sharable.

2. Attempt any FOUR of the following : Marks 16

a) List memory allocation methods. Explain any one.

(Listing-1 Mark, Explanation of any one- 3 Marks)

Ans: Memory Allocation Methods are as follows:

1. Contiguous Allocation
2. Paging
3. Segmentation
4. Segmentation with Paging
5. Demand Paging

1. **Contiguous Allocation**

Main memory usually into two partitions:

Resident operating system, usually held in low memory with interrupt vector. User processes then held in high memory. Single-partition allocation Relocation-register scheme used to protect user processes from each other, and from changing operating-system code and data. Relocation register contains value of smallest physical address; limit register contains range of logical addresses each logical address must be less than the limit register.

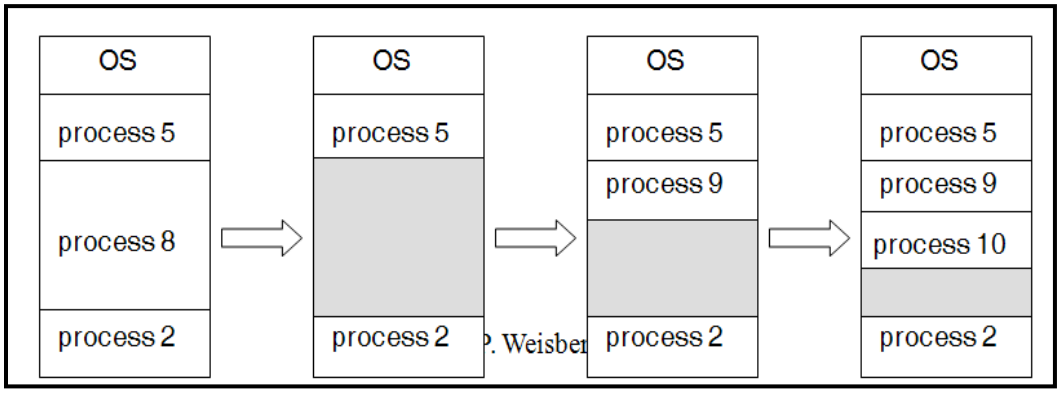
Multiple-partition allocation

Hole – block of available memory; holes of various size are Scattered throughout memory.

When a process arrives, it is allocated memory from a hole Large enough to accommodate it.

Operating system maintains information about:

a) allocated partitions b) free partitions (hole)



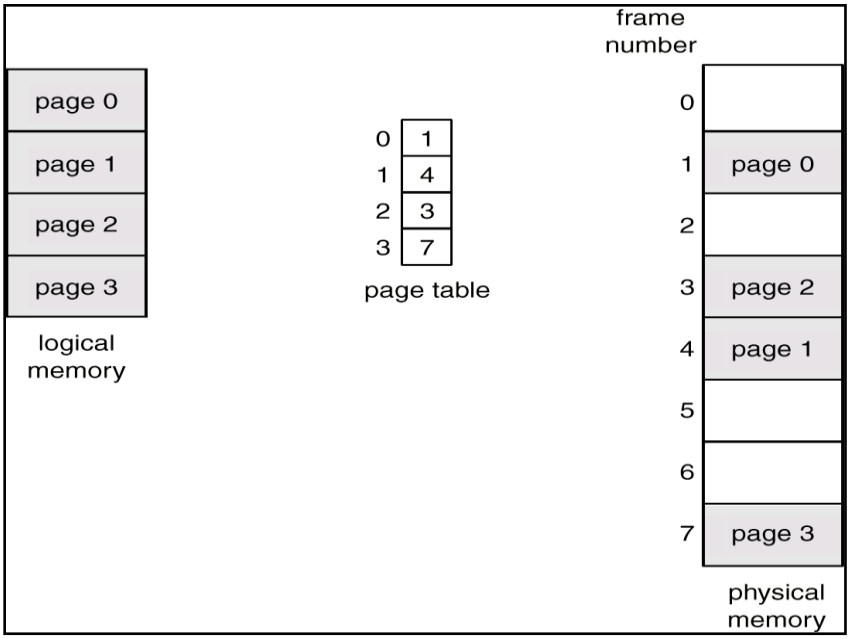
2. Paging

Logical address space of a process can be non contiguous. 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**. Keep track of all free frames. To run a program of size n pages, need to find n free frames and load program. Set up a page table to translate logical to physical addresses. Internal fragmentation.

Address generated by CPU is divided into: Page number (p) – used as an index into a page table which contains base address of each page in physical memory.

Page offset (d) – combined with base address to define the physical memory address that is sent to the memory unit.

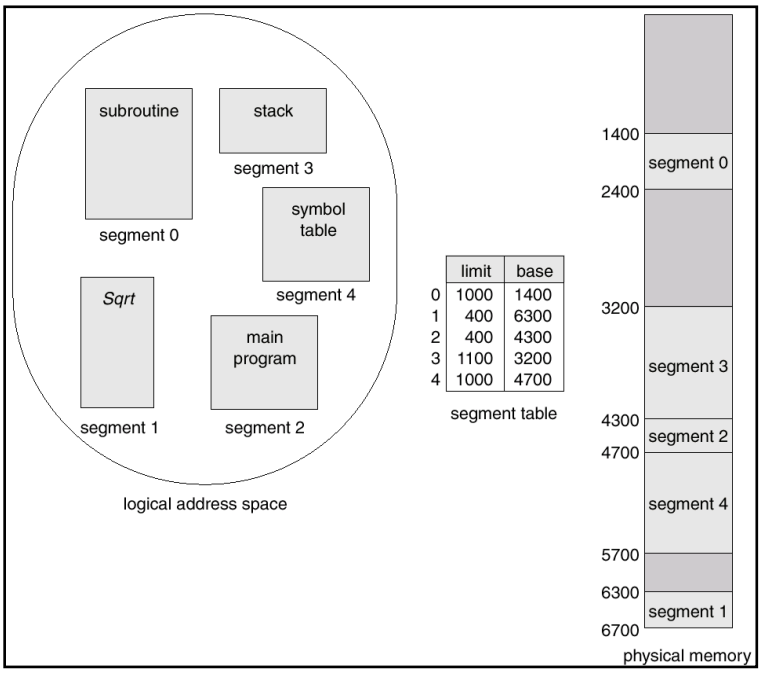
Example of Paging



3. Segmentation

Memory-management scheme that supports user view of memory. A program is a collection of segments. A segment is a logical unit such as: main program, procedure, function, method, object, local variables, global variables, common block, tack, symbol table, arrays

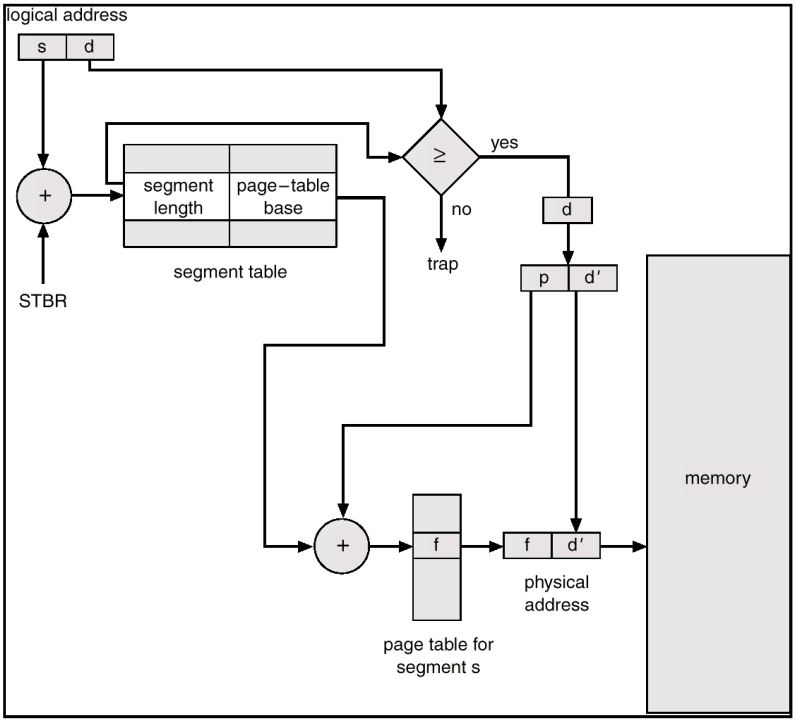
Example of Segmentation



4. Segmentation with Paging

The MULTICS system solved problems of external fragmentation and lengthy search times by paging the segments. Solution differs from pure segmentation in that the segment-table entry contains not the base address of the segment, but rather the base address of a page table for this segment.

MULTICS Address Translation Scheme



5. Demand Paging

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

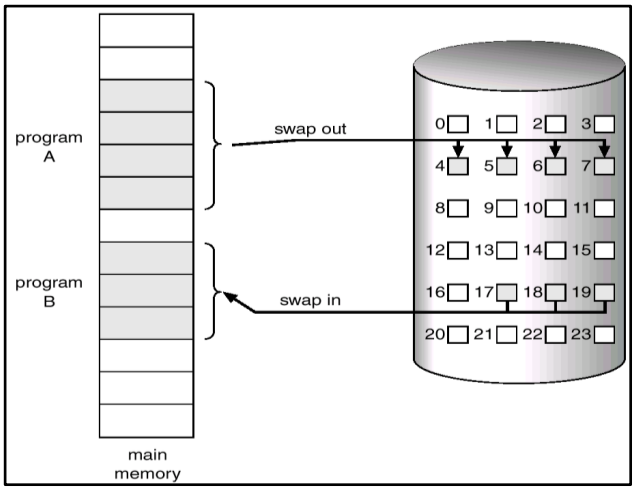
Virtual memory uses a technique called demand paging for its implementation.

When we want to execute a process, it is swapped into the memory. However, a pager (lazy swapper) does not bring the whole process into the memory. Only those pages, which are needed, are brought into the memory. That is, bring a page into memory only when it is needed.

Demand paging has the many benefits such as

- Less I/O needed
- Less memory needed
- Faster response
- More users

Logical address space of a process can be noncontiguous; process is allocated physical memory whenever that memory is available and the program needs it. Divide physical memory into fixed-sized blocks called frames (size is power of 2, between 512 bytes and 8192 bytes).



Demand Paging

b) What is real time operating system?

(Explanation- 4 Marks)

Ans: A real time system has well defined fixed time constraints. Processing should be done within the defined constraints -Hard and Soft real time system

Hard real-time

- Guarantees critical task completion on time.
- Secondary storage limited or absent, data stored in short term memory, or read-only memory (ROM)
- Conflicts with time-sharing systems, not supported by general-purpose operating systems.
- Advanced OS features are absent (e.g. virtual memory is absent).

Soft real-time

- Less restrictive.
- A critical real time task gets priority over other tasks and it retains its priority until it completes.
- Limited utility in industrial control of robotics

Example – Flight Control System

All tasks in that system must execute on time.

Example: Satellite application of real time OS-

The satellite connected to the computer system sends the digital samples at the rate of 1000 samples per second. The computer system has an application program that stores these samples in

a file. The sample sent by the satellite arrives every millisecond to the application. So computer must store or respond the sample in less than 1 millisecond. If the computer does not respond to the sample within this time, the sample will lost.

Some of the examples of Real time systems are: A web server, A word processor, An audio/video media center, A microwave oven, A chess computer.

c) Explain microlevel OS structure.

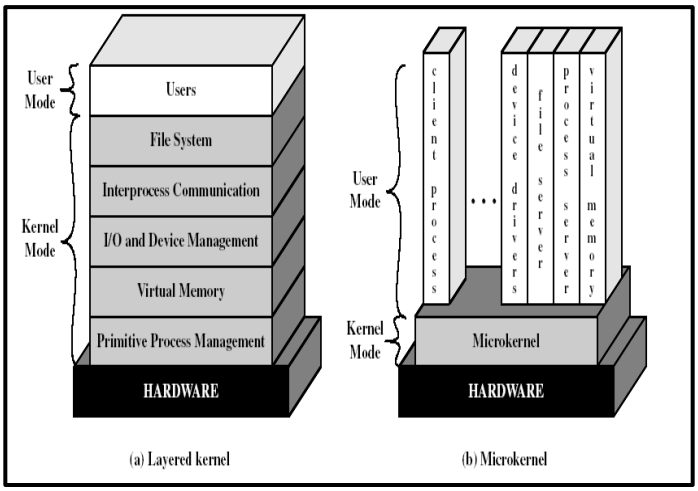
(Explanation- 3 Marks, Diagram-1 Mark)

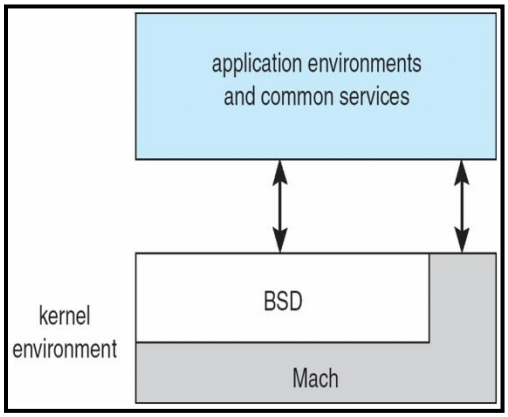
(**Note: Explanation of Microkernel OS is given below. Relevant answer for any OS structure, marks should be given)

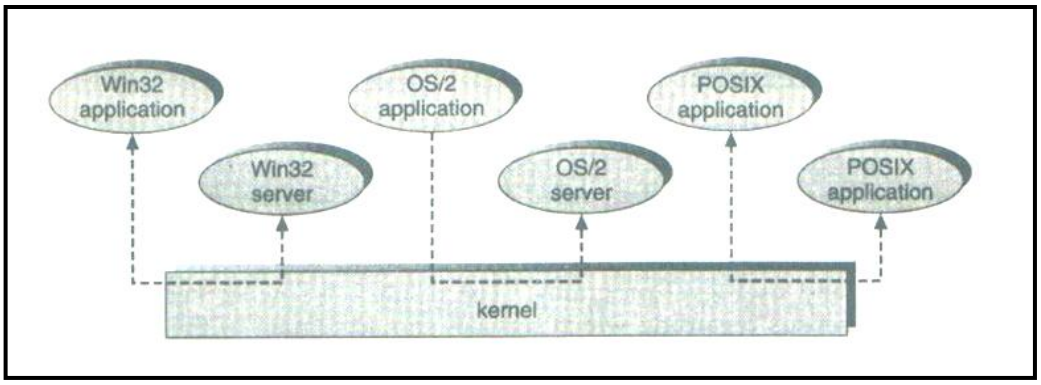
Ans: A **microkernel** (also known as μ-kernel) is the near-minimum amount of software that can provide the mechanisms needed to implement an operating system (OS). These mechanisms include low-level address space management, thread management, and inter-process communication (IPC).If the hardware provides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

Benefits:

1. easier to extend a microkernel
2. easier to port the operating system to new architectures
3. more reliable (less code is running in kernel mode)
4. more secure







d) What is deadlock? Write necessary condition of dead lock.

(Definition- 1 Mark, 4 conditions- 3 Marks)

Ans: Deadlock

A deadlock consists of a set of blocked processes, each holding a resource and waiting to acquire a resource held by another process in the set

Deadlock can arise if four conditions hold simultaneously.

Deadlock Characterization

Mutual exclusion: only one process at a time can use a resource

Hold and wait: a process holding at least one resource is waiting to acquire additional resources held by other processes

No pre-emption: a resource can be released only voluntarily by the process holding it after that process has completed its task

Circular wait: there exists a set $\{P_0, P_1, \ldots, P_0\}$ of waiting processes such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by

$P_2, \ldots, P_{n-1}$ is waiting for a resource that is held by P_n, and P_n is waiting for a resource that is held by P_0

e) What is file? List any four attributes of files.

(Definition- 2 Marks, Any 4 attributes- ½ Mark each)

Ans: A file is named for convenience of its human users and is referred to by its name when a file is named, it becomes independent of the process, the user, and even the system that created it. For instance, one user might create the file example.c, and another user might edit that file by specifying its name.

A file's attributes vary from one OS to another but typically consist of these:

- **Name**. It is a string of characters which is in human readable form.
- **Identifier**. This unique tag, usually a number, identifies the file within the file system; it is the non-human-readable name for the file.
- **Type**. This is the information used by the system to support different types of the files.
- **Location**. This information is a pointer to a device and to the location of the file on that device.
- **Size**. The current size of the file (in bytes, words, or blocks) and possibly the maximum allowed size are included in this attribute.
- **Protection**. Access-control information determines who can do reading, writing, executing, and so on.
- **Time, date, and user identification**. This information may be kept for creation, last modification, and last use.

The information about all files is kept in the directory structure, which also resides on secondary storage. Typically, a directory entry consists of the file's name and its unique identifier.

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

f) Describe pre emptive and Non-pre emptive scheduling.

(Any four points- 1 Mark each)

Ans: Preemptive Scheduling:

Even if CPU is allocated to one process, CPU can be preempted to other process if other process is having higher priority or some other fulfilling criteria. Throughput is less. It is suitable for RTS. Only the processes having higher priority are scheduled. It doesn't treat all processes as equal.

Algorithm design is complex. Circumstances for preemptive:

- process switch from running to ready state
- process switch from waiting to ready state

For e.g.: Round Robin, Priority algorithms

Non Preemptive Scheduling:

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. Throughput is high. It is not suitable for

RTS. Processes having any priority can get scheduled. It treats all process as equal. Algorithm design is simple. Circumstances for Non preemptive

- process switches from running to waiting state
- process terminates

For e.g.: FCFS algorithm

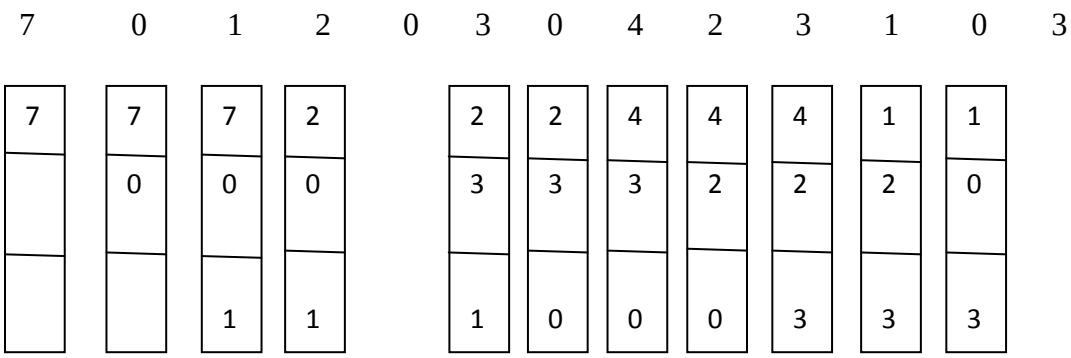
3. Attempt any Four of the following : Marks 16

a) Explain FIFO page replacement algorithms for reference string.

7 0 1 2 0 3 0 4 2 3 10 3

(Explanation 4 Marks)

Ans: First-In-First-Out (FIFO) Algorithm: A FIFO replacement associates with each page the time when that page was bought into memory. When the page must be replaced we replace the page at the, the oldest page is chosen. We replace the page at the head of the queue. When a page is brought into the memory, we insert it at the tail of the queue.



Page hit 2

Page fault 11

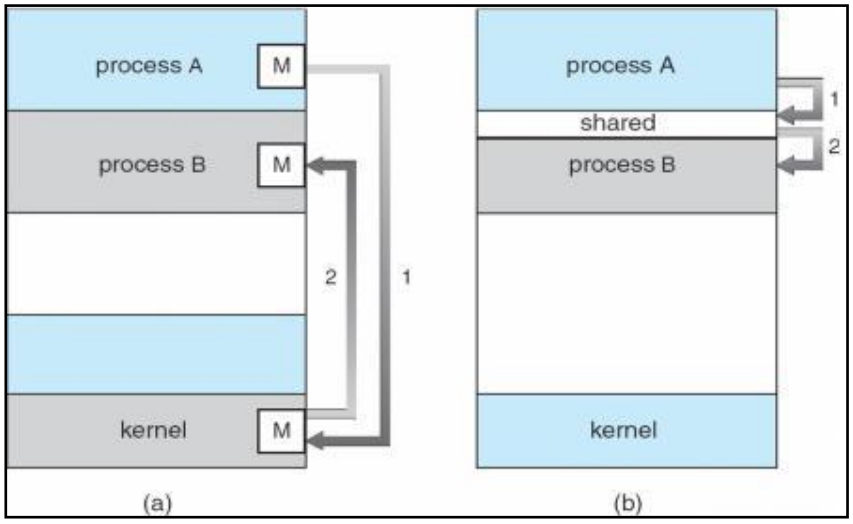
b) Draw and explain interprocess communication model.

(Diagram -1 Mark each, Explanation -2 Marks)

Ans: **Inter-process communication:** Cooperating processes require an Inter- process communication (IPC) mechanism that will allow them to exchange data and information. There are two models of IPC

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

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



c) Describe the terms :

i) Scheduling queues

ii) Context switch

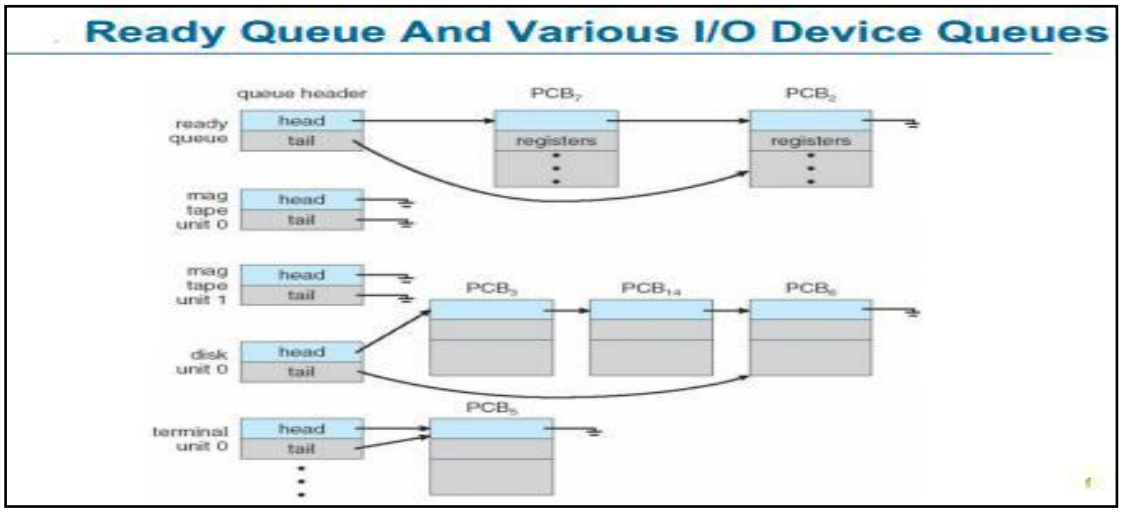
Ans: i) Scheduling queue

(Description of queues- 2 Marks)

ready queue : The processes that are residing in main memory and are ready and waiting to execute are kept on a list called the ready queue.

job queue : As processes enter the system they are put into a job queue.

device queue : The list of processes waiting for a particular I/O device is called device queue



ii) Context switch

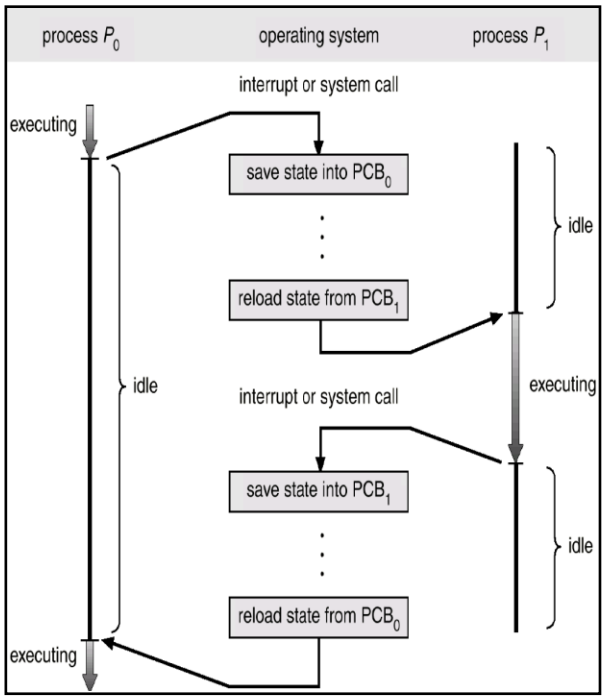
(Description - 2 Marks)

Switch the CPU to another process requires saving the state of old process and loading the saved state for new process. This time is known as a context switch. The context switch represented in PCB.

Saves context of old process in its PCB and loads context of new process schedule to run.

Pure overhead

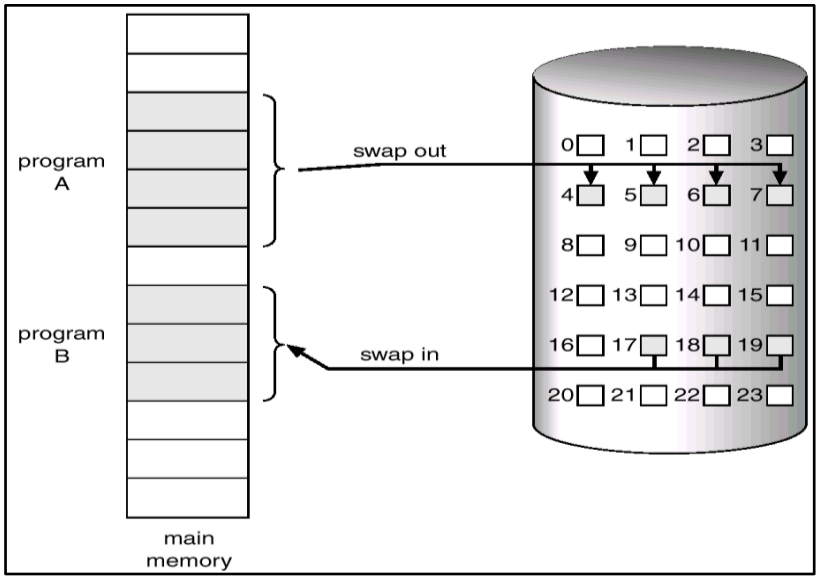
Depend on hardware support



d) Describe demand paging in detail.

(Explanation - 3 Marks, Diagram - 1 Mark)

Ans:



Paging refers to the transfer of memory pages from the physical memory to disk. Virtual memory uses a technique called demand paging for its implementation. When we want to execute a process, it is swapped into the memory. However, a pager (lazy swapper) does not bring the whole process

into the memory. Only those pages, which are needed, are brought into the memory. That is, bring a page into memory only when it is needed. Demand paging has the many benefits such as

Less I/O needed

- Less memory needed
- Faster response
- More users

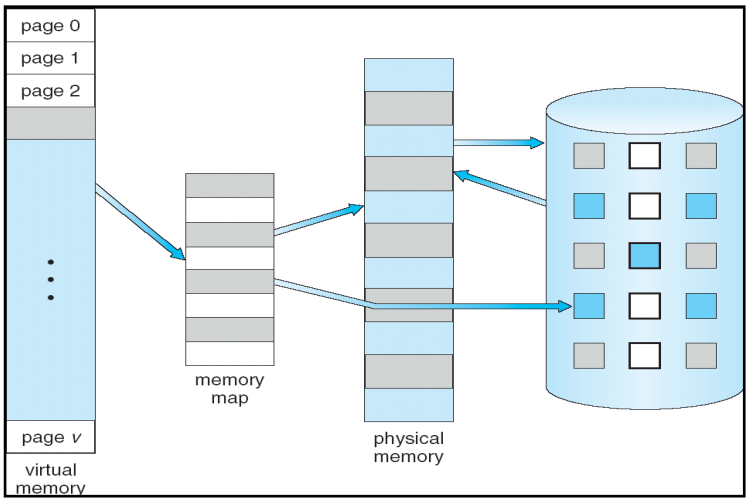
Logical address space of a process can be noncontiguous; process is allocated physical memory whenever that memory is available and the program needs it. Divide physical memory into fixed-sized blocks called frames (size is power of 2, between 512 bytes and 8192 bytes).

e) Explain concept of virtual memory in detail with example.

(Diagram- 1 Mark, Explanation -2 Marks, example 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.



f) Differentiate between multiprogramming and multitasking O.S.

(Any relevant 4 points - 1 Mark each)

Ans:

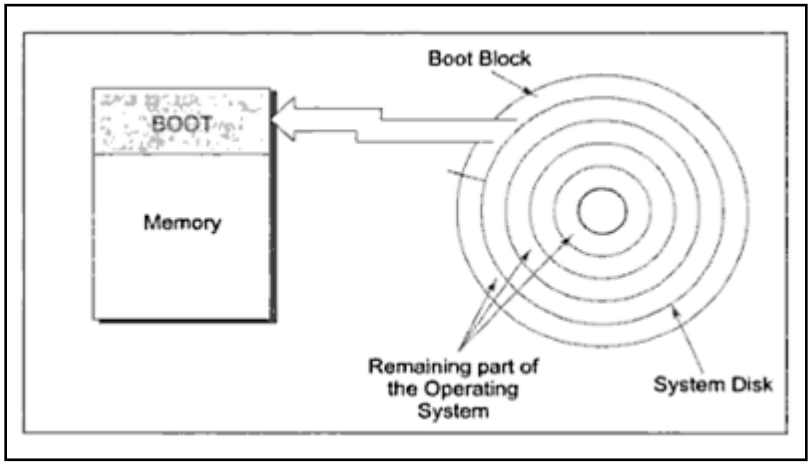
Multiprogramming	Multitasking
In multiprogramming, more than one program lies in the memory i.e. in terms of operating system, the schedular selects the jobs to be placed in ready queue from a number of programs. The ready queue is placed in memory and the existence of more than one program in main memory is known as multiprogramming.	Multitasking means performing multiple tasks in parallel. Usually, CPU processes only one task at a time but the switching of CPU between the processes (also known as Context Switching) is so fast that it looks like CPU(or processor) is executing multiple processes at a time.
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.	In multitasking systems the CPU executes multiple jobs by switching among them, but the switching occurs so frequently that the users can interact with each program while it is running.
Multiprogramming is the simple form of parallel processing in which several programs run at the same time on a processor	Multitasking is a logical extension of multiprogramming in which several tasks executed at same time
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 used at the same time.	Example of multitasking, we listen to music and do internet browsing at the same time (they execute parallely).

4. Attempt any FOUR of the following. Marks 16

a) Explain system booting in detail.

(Diagram-1 Mark, Explanation -3 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 boot strapping or its short form booting.



b) Explain process termination.

(*Explanation - 4 Marks*)

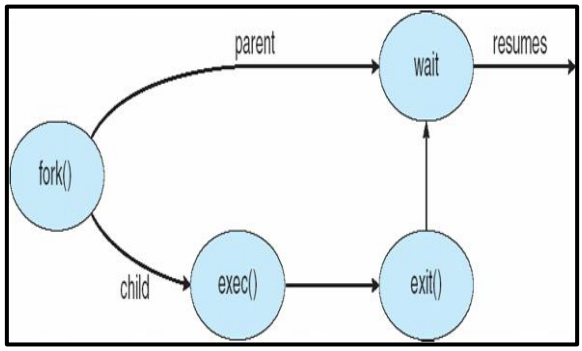
Ans: Process executes last statement and asks the operating system to delete it (**exit**).

i) Output data from child to parent (**via wait**).

ii) Process" resources are deallocated by operating system DELETE system call is used for terminating a process. A process may delete itself or by another process. A process can cause the termination of another process via an appropriate system call. The operating system reacts by

reclaiming all resources allocated to the specified process, closing files opened by or for the process. PCB is also removed from its place of residence in the list and is returned to the free pool. The DELETE service is normally invoked as a part of orderly program termination. Following are the resources for terminating the child process by parent process. Parent may terminate execution of children processes (**abort**).

1. The task given to the child is no longer required.
2. Child has exceeded its usage of some of the resources that it has been allocated.
3. Operating system does not allow a child to continue if its parent terminates, all children terminated - **cascading termination.**



c) Differentiate between short term and long term scheduler.

(*Any Relevant 4 points - 4 Marks*)

Ans:

Short term scheduler	Long term scheduler
Short term scheduler is known as CPU scheduler.	Long-term scheduler is known as job scheduler.
Short-term scheduler selects which process should be executed next and allocates CPU.	Long-term scheduler selects which processes should be brought into the ready queue
Short-term scheduler is invoked very frequently (milliseconds)	Long-term scheduler is invoked very infrequently (seconds, minutes)
Short term scheduler is fast	Long term scheduler is relatively slow

d) What are different free space management techniques?

Describe any one in detail.

(List – 1 Mark, Explanation of any one- 3 Marks)

Ans: Different free space management techniques are:

1. Bit vector
2. Linked list
3. Grouping
4. Counting

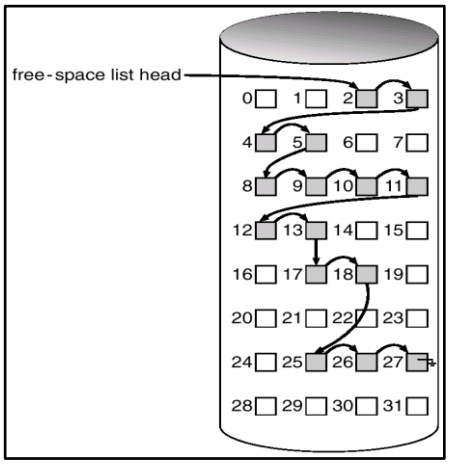
1. **Bit Vector:** frequently, the free-space list is implemented as bit map or bit vector. Each block is represented by 1 bit. If the block is free, the bit is 1: if the block is allocated, the bit is 0.

   For example, consider a disk where blocks 2, 3, 4, 5, 8, 9, 10,11,12,13,17, 18, 25, 26, and are free, and the rest of the block are allocated. The free-space bit map would be

   001111001111110001100000011100000…

   The main advantage of this approach is that it is relatively simple and efficient to find the first free block, or n consecutive free block on the disk. Indeed many computers supply bit-manipulation instructions that can be used effectively for that purpose.

2. **Linked list:** In Linked list, all free space disk are linked, keeping a pointer to the first free block in a special location on a disk and caching if in memory. This method has negligible space overhead because there is no need for a disk allocation table, merely for a pointer to the beginning of the chain and the length of the first portion. This method is suited to all the file allocation methods.



3. **Grouping:** A modification of the free-list approach is to store the address of n free blocks in the first free block. The first n-1 of these blocks are actually free. The last block contains the addresses of another n free blocks, and so on. The importance of this implementation is that the addresses of large number of free blocks can be found, quickly, unlike in the standard linked-list approach.
4. **Counting:** Another approach is to take advantage of the fact tha, generally, several contiguous blocks may be allocated or freed simultaneously, particularly when space is allocated with the contiguous allocation algorithm or through clustering. Thus, rather than keeping a list of n free disk addresses, we can keep the address of the first free block and the number n of free contiguous blocks that follows the first block. Each entry in the free-space list then consists of a disk address and a count. Although each entry requires more space than would a simple disk address, the overall list will be shorter, as long as the count is generally greater than 1.

e) Explain different process scheduling criteria.

(All criteria-4 Marks)

Ans: **CPU utilization**: Keep the CPU as busy as possible.

Throughput: No of processes that complete their execution per time unit.

Turnaround time: Amount of time to execute a particular process. The interval from the time of submission of a process to the time of completion is the turnaround time.

Waiting time: Amount of time a process has been waiting in the ready queue

Response time: Amount of time it takes from when a request was submitted until the first response is produced, **not** output (for time-sharing environment)

f) State the rules for naming files. How is file security achieved?

(Any 4 rules - ½ Mark each, file security methods 2 Marks)

Ans: 1.The exact rules for file naming vary somewhat from system to system, but all operating system allow strings of one to eight letters as legal file names.

2. Frequently digits and special characters' are also permitted
3. Many file system support names as long as 255 characters.
4. Some file system distinguish between upper case letters and lower case letters.

5. Many Operating System support two part file names, e.g. porg.c

The three most popular access to a file protection are the following:

File Naming: It is important that no two users try to create the file with the same name.

Passwords: This scheme associates a password to each file. If user does not know the password associated to a file then he cannot access it. This is a very effective way of protecting files but for a user who owns many file, and constantly changes the password to make sure that nobody accesses these files will require that users have photographic memories.

Access control: An access list is associated to each file. the access list contains information on the type of users and accesses that they can do on a file.

5. Attempt any FOUR of the following : Marks 16

a) Enlist system components. Describe any one in detail.

(List - 1 Mark, Description of any one - 3 Marks)

Ans: List of System Components:

1. Process management
2. Main memory management
3. File management
4. I/O system management
5. Secondary storage management

Process Management

The operating system manages many kinds of activities ranging from user programs to system programs like printer spooler, name servers, file server etc. Each of these activities is encapsulated in a process. A process includes the complete execution context (code, data, PC, registers, OS resources in use etc.). The five major activities of an operating system in regard to process management are

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

Main-Memory Management

Primary-Memory or Main-Memory is a large array of words or bytes. Each word or byte has its own address. Main-memory provides storage that can be access directly by the CPU. That is to say for a program to be executed, it must in the main memory. The major activities of an operating in regard to memory-management are:

1. Keep track of which part of memory are currently being used and by whom.
2. Decide which process are loaded into memory when memory space becomes available
3. Allocate and deallocate memory space as needed.

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

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

I/O System Management I/O subsystem hides the peculiarities of specific hardware devices from the user. Only the device driver knows the peculiarities of the specific device to whom it is assigned.

Secondary-Storage Management

Systems have several levels of storage, including primary storage, secondary storage and cache storage. Instructions and data must be placed in primary storage or cache to be referenced by a running program. Because main memory is too small to accommodate all data and programs, and its data are lost when power is lost, the computer system must provide secondary storage to back up main memory. Secondary storage consists of tapes, disks, and other media designed to hold information that will eventually be accessed in primary storage (primary, secondary, cache) is ordinarily divided into bytes or words consisting of a fixed number of bytes. Each location in

storage has an address; the set of all addresses available to a program is called an address space. The three major activities of an operating system in regard to secondary storage management are:

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

b) Explain process state with diagram.

(Process states diagram – 2 Marks, Description -2 Marks)

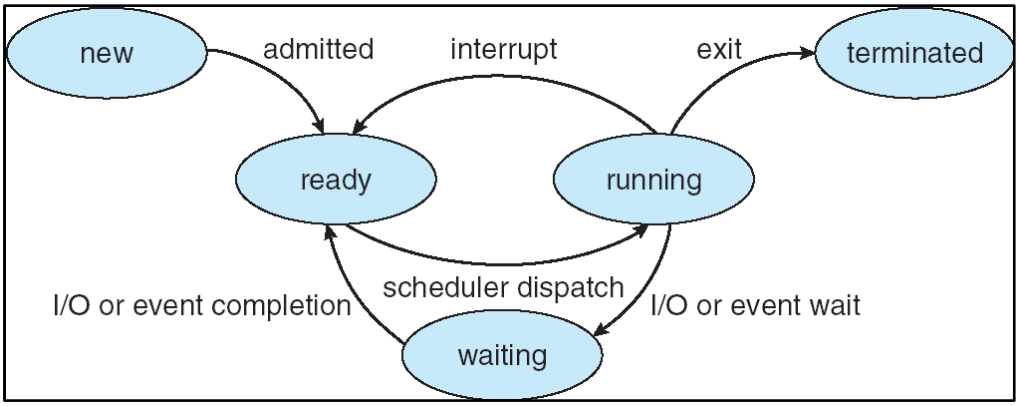
Ans: Process is a program in execution. A process does not mean only program but it could contain some part called as text section. It may contain the current activity represented by the value of the program counter & the contents of CPU register.

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.

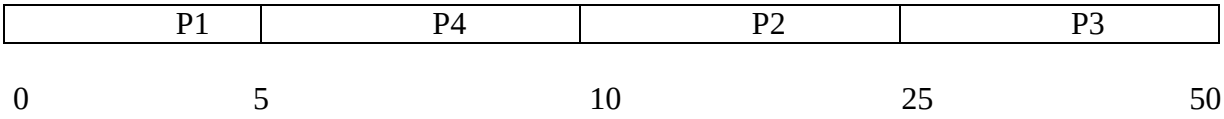
Terminated / Halted State: The process whose operation is completed, it will go the terminated state from the running state. In halted state, the memory occupied by the process is released.

c) The jobs are scheduled for execution as follows. Solve following problem using SJF scheduling algorithm.

Process	Burst time
P1	**5**
P 2	**15**
P 3	**25**
P 4	**5**

(Gantt chart- 1Mark, Waiting time/turnaround time for each process -2 Marks, Average waiting time/Average Turnaround time -1 Mark)

Ans: Gantt chart:



Waiting time for each process: P1 =0;

P2 =10;

P3 =25 ;

P4 =05

Average waiting time = (0+10+25+05)/4

= 40/4 = 10

OR

Turnaround time for each process: P1 =5;

P2 =25;

P3 =50;

P4 =10

Average waiting time = (5+25+50+10)/4

= 90/4 = 22.5

d) Explain banker's algorithm for deadlock prevention.

(Algorithm - 2 Marks, Data structure - 2 Marks)

Ans: Banker's Algorithm:

Multiple instances. Each process must a priori claim maximum use. When a process requests a resource it may have to wait. When a process gets all its resources it must return them in a finite amount of time. 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 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.

Data Structures for the Banker's Algorithm

Available*:* Vector of length *m*. If available [*j*] = *k*, there are *k* instances of resource type *Rj*available.

Max: *n x m* matrix. If *Max* [*i,j*] = *k*, then process *Pi* may request at most *k* instances of resource type *Rj*.

Allocation*:* *n* x *m* matrix. If Allocation[*i,j*] = *k* then *Pi* is currently allocated *k* instances of *Rj*.

Need: *n* x *m* matrix. If *Need*[*i,j*] = *k*, then *Pi* may need *k* more instances of *Rj*to complete its task.

Need [*i,j] = Max*[*i,j*] – *Allocation* [*i,j*]. Let *n* = number of processes, and *m* = number of resources types.

e) Describe sequential and direct access method.

(Sequential diagram- 1 Mark, Explanation- 1 Mark, Direct access method explanation- 2 Marks)

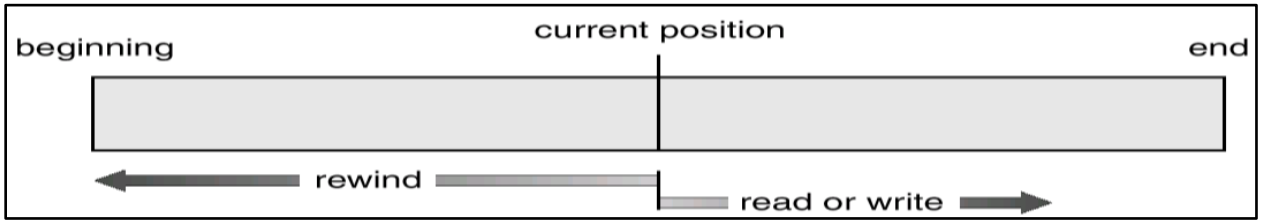
Ans: Most programming languages provide two different ways to access data stored in a file i.e. information. When it is used, this information must be accessed and read into computer memory. The information in the file can be accessed in several ways **Sequential Access Method:**

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).



If you want to read a piece of data that is stored at the very end of the file, you have 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 you want to listen to the last song on a cassette tape, you have 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.

f) What is the purpose of the system calls? State two system calls with its functions.

(Purpose - 2 Marks, Any 2 System calls with functions - 1 Mark each)

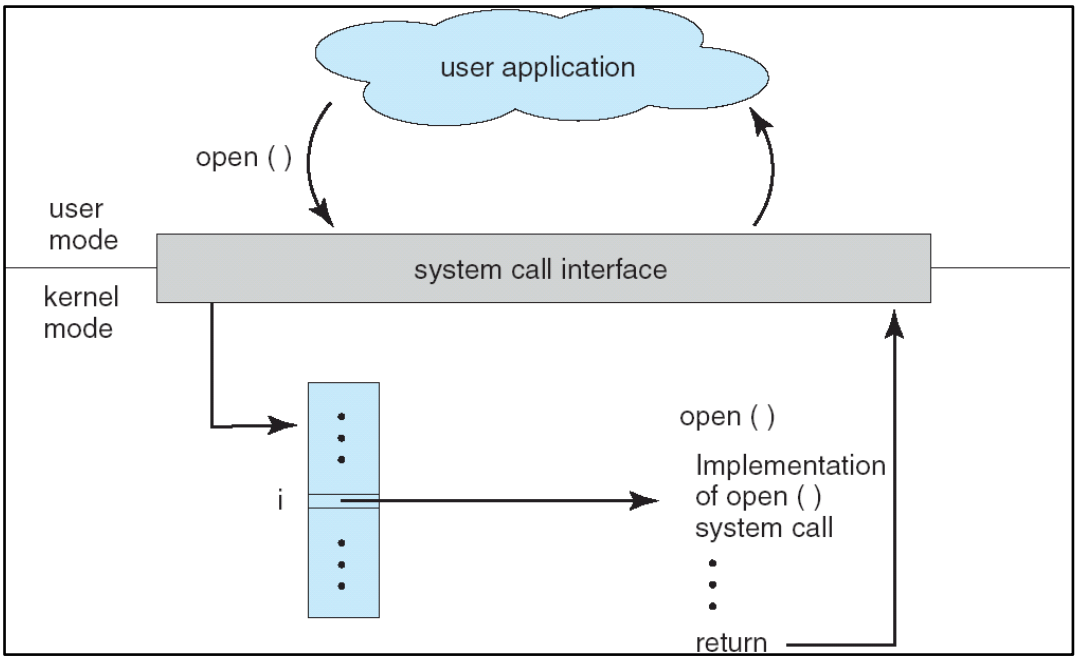
Ans: **System Calls:** System calls are programming interface to the services provided by the operating system.

Implementation:

1. Number to number the system calls, 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 & 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.



System calls can be roughly grouped into the following major categories.

a. Process or Job control

b. File Management

c. Device Management

d. Information Maintenance

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

System calls Related to File management: Create file, delete file Open file , Close file Create directory Read, write, Reposition Get file attribute , set file attribute Create a link Change the working directory

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 Information Maintenance: Get Time or Date, Set Time or Date Get System data, Set system data Get process, file or device attributes Set process, file or Device attributes.

6. Attempt any FOUR of the following : Marks 16

a) What are different responsibilities of memory management?

(Explanation - 4 Marks)

Ans: The services provided under memory management are directed to keeping track of memory and allocating/deallocating it to various processes. The OS keeps a list of free

Memory locations.(Initially, after the booting but before any process starts, the full memory, excepting the part occupied by the OS itself is free.)Before a program is loaded in the memory from the disk, this module (MM) consults this free list, allocates the memory to the process, depending upon the program size and updates the list of free memory.

Main memory management

Primary-Memory or Main-Memory is a large array of words or bytes. Each word or byte has its own address. Main-memory provides storage that can be access directly by the CPU. That is to say for a program to be executed, it must in the main memory

Functions of memory management:

1) Keeping track of which part of memory are currently being used and by whom

2) Deciding which processes are to be loaded in to memory when memory space becomes available.

3) Allocating& De allocating space as needed

b) Explain multilevel feedback scheduling algorithm in detail.

(Diagram- 1 Mark, Explanation -3 Marks)

Ans: Multilevel Feedback Queue A process can move between the various queues; aging can be implemented this way. Multilevel-feedback-queue scheduler defined by the following parameters:

Number of queues

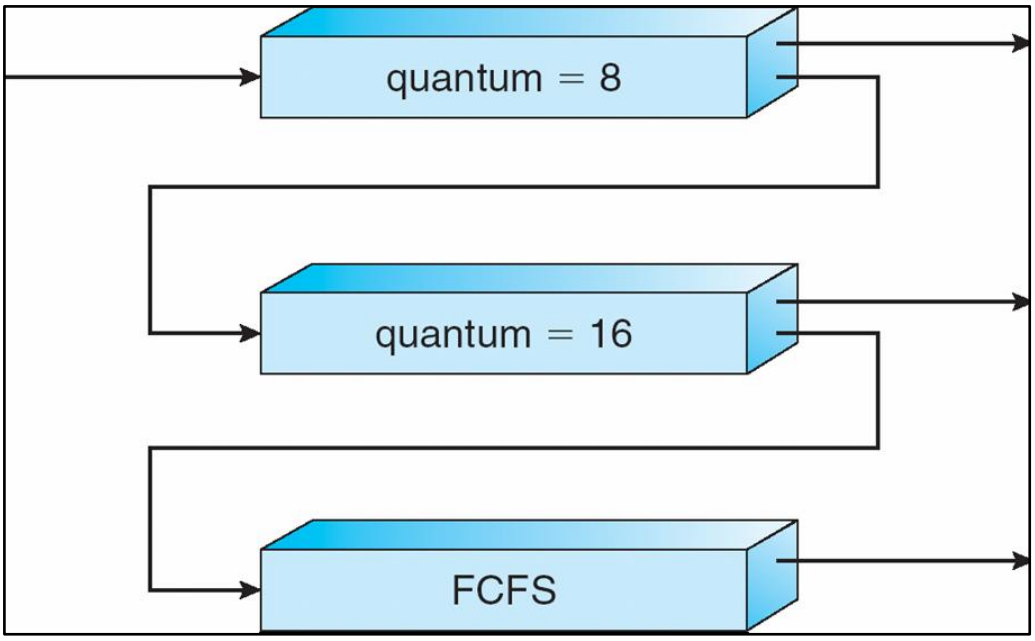
Scheduling algorithms for each queue

Method used to determine when to upgrade a process.

Method used to determine when to demote a process.

Method used to determine which queue a process will enter when that process needs service

Example of Multilevel Feedback Queue Three queues: *Q*0 – time quantum 8 milliseconds *Q*1 – time quantum 16 milliseconds *Q*2 – FCFS Scheduling A new job enters queue *Q0* which is served FCFS. When it gains CPU, job receives 8 milliseconds. If it does not finish in 8 milliseconds, job is moved to queue *Q*1. At *Q*1 job is again served FCFS and receives 16 additional milliseconds. If it still does not complete, it is preempted and moved to queue Q2.



c) Describe priority scheduling algorithm.

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

Ans: Priorty scheduling algorithm:

Number (integer) is associated with each process

The CPU is allocated A priority to the process with the highest priority (smallest integer = highest priority)

- Preemptive
- Nonpreemptive

SJF is a priority scheduling where priority is the predicted next CPU burst time

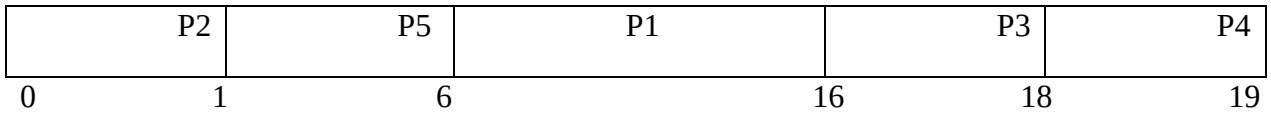
Problem = **Starvation** – low priority processes may never execute

Solution = **Aging** – as time progresses increase the priority of the process

Example:

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

Gantt chart:



Waiting time for each process : p1 =6 ,p2 =0, p3 =16 ,p4 =18 ,p5 =1

Average waiting time:= (6+0+16+18+1) /5

=41/5 = 8.2 milliseconds

d) Describe LRU page replacement algorithm.

(Explanation - 2 Marks, Example - 2 Marks)

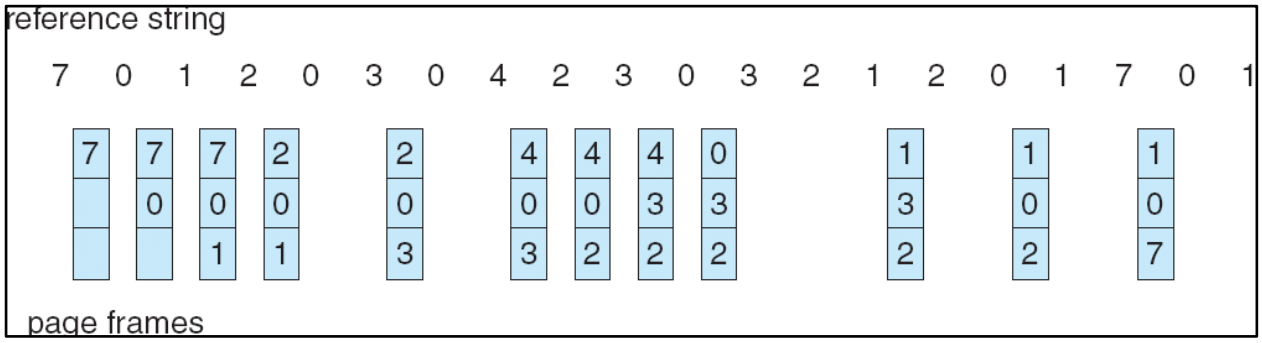
Ans: **Least Recently Used (LRU) Algorithm** Counter implementation Every page entry has a counter; every time page is referenced through this entry, copy the clock into the counter. When a page needs to be changed, look at the counters to determine which are to change Stack implementation – keep a stack of page numbers in a double link form: Page referenced:

move it to the top

requires 6 pointers to be changed

No search for replacement

Advantage: Less number of faults as compared to FIFO



LRU can be implemented using a clock-based approach

- when referenced, record the time in the page table (takes up a lot of space)
- when selecting victim, sort reference times to locate the earliest (time consuming) alternatively, could use a stack-based implementation
- maintain a (pseudo) stack of page numbers, most recent at top. when a page is referenced, move its number to the top (requires some work)
- when need to select a page, LRU is at bottom of stack.

e) What are function of OS?

(***Any four functions- 1 Mark each)***

Ans: The major functions of an operating system are:

1. Resource Management: This function of OS allocates computer resources such as CPU time, main memory, secondary storage and input and output devices for use.

2. **Data management**: It observes input and output of the data and their location, storage and retrieval.

3. **Task management**: Task is a collection of one or more related programs and their data. This function prepares, schedules, controls and monitors jobs submitted for execution to ensure the most efficient processing.

4. **Allocation of Resources:** Handles system resources such as computer's memory and sharing of the central processing unit (CPU) time by various applications or peripheral devices

5. **Communication between User and Computer** : Provides a user interface, e.g. command line, graphical user interface (GUI)

6. Operating system enables start up application programs. OS must have text editor, a translator and an editor.

7. Operating system provides number of services such as for the programmer it provides utilities ie debugger, editors, file management which refers to the way that the operating system manipulates, stores, retrieves and saves data. It interpret the commands executed by the user. It handle disk input/output settings.

OR

1. **Process Management** – Managing the programs that are running.

2. **Memory Management** – Managing and rationing the memory between processes and data.

3. **Storage Management** – Managing the permanent Storage of data on disks or other media

4. **I/O Management** – Managing the input and output

5. **Device / Resource Management** – Managing devices and resources and allowing the users to share the resources

6. **Security and Protection** – Securing the system against possible unauthorized access to data or any other entity. Protecting the parts of the system against damage.

7. **Booting the System and getting it ready to work.**

8. **Data communications –** Providing interface to connect to other computers or allowing others to connect.

f) Explain contiguous memory allocation for file.

(Diagram- 2 Marks, Explanation -2 Marks)

Ans: Contiguous Allocation:

1) Each file occupies a set of contiguous blocks on the disk. Disk addresses define a linear ordering on the disk. Contiguous allocation of a file is defined by the disk address and length.

2) Simple – only starting location (block #) and length (number of blocks) are required.

3) Both sequential and direct access can be supported.

4) Wasteful of space (dynamic storage-allocation problem).

5) Files cannot grow.