Question No.: Q. 2 (A)

→ i) Microkernel is one of the classification of the Kernel.

ii) Being a kernel it manages all system resources.

iii) But in a microkernel, the user services and kernel services are implemented in different address space.

iv) The user services are kept in user address space, and kernel services are kept under Kernel address space, Thus also reduces the size of kernel and size of operating system as well.

v) The microkernel is solely responsible for the most important services of operating system they are named as follows :-
   a] Inter process - Communication
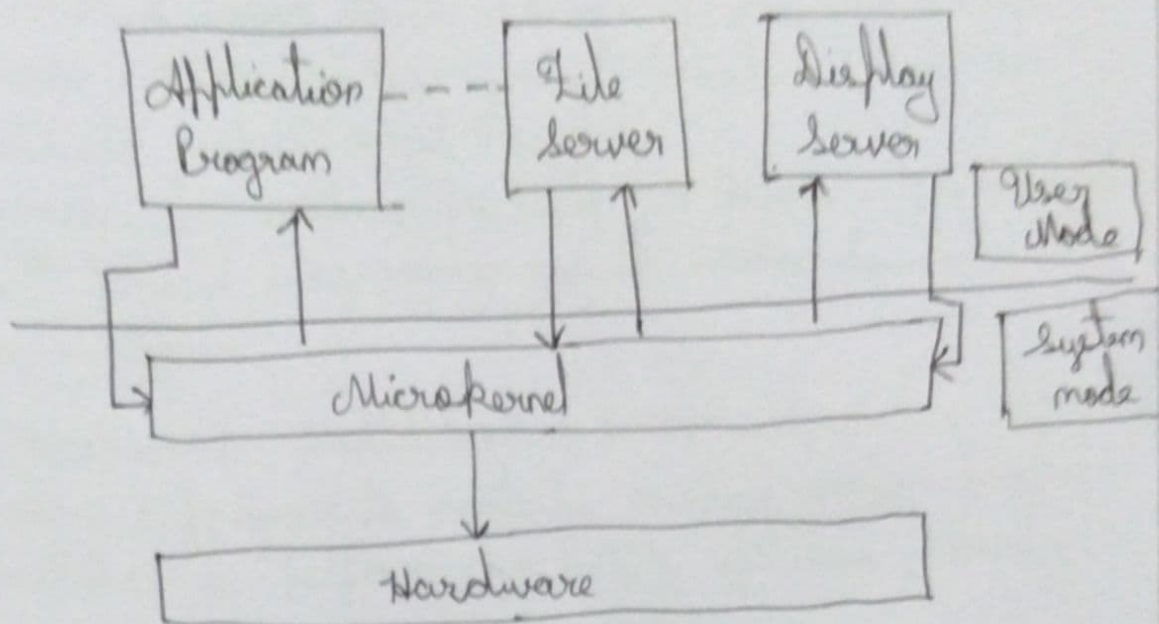   b] Memory Management.
   c] CPU Scheduling.

Advantages of Microkernel :-

1) The architecture of this kernel is small and isolated hence it can function better.

ii) Expansion of the system is easier; It is simply added in the system application without disturbing the kernel.

Eclipse IDE is a good example of Microkernel architecture

(P.T.O.)

Question No.: Q.2(A)

Question no.: Q. 2 (B)

→ **Thread :-** A thread is a path of execution within a process. There are two types of threads. i.e. (i) User level threads and (ii) Kernel thread.

A process can contain Multiple threads. The benefits of multi threaded programming can be broken down into four major categories :-

1] Responsiveness :- Multithreading in an interactive application may allow a program to continue running even if a part of it is blocked or is performing a lengthy operation, thereby increasing responsiveness to the user.

2] Resource sharing :- Processes may share resources only through techniques such as :- a] Message passing and b] Shared Memory.

Such techniques must be explicitly organized by programmer. However, threads share the memory and the resources of the process to which they belong by default.

3] Economy :- Allocating memory and resources for process creation is a costly job in terms of time and space. Since, threads share memory with the process it belongs, it is more economical to create and context switch threads.

Question no. : Q. 2(B)

iv) <u>Scalability</u> :- The benefits of multi - programming greatly increase in case of multiprocessor architecture, where threads may be running parallel on multiple processors. If there is only one thread then it is not possible. Multi - threading on a multiple CPU machine increases parallelism.

Question no. : Q.2 (c)

→ i) A semaphore S is integer variable whose value can be accessed and changed only by two operations wait (P or sleep or down) and signal (V or wakeup or up). Wait and signal are atomic operations.

ii) Binary semaphores does not assume all the integer values.

iii) It assumes only two values 0 and 1. On the contrary, counting semaphores (general semaphores) can assume only non-negative values.

iv) The wait operation on semaphores S, written as wait (s) or P (s), operates as follows:

wait (s) : IF S > 0
           THEN $S_r = S - 1$
           ELSE (wait on S)

The signal operation on semaphore S, written as signal (s) or V (s), operates as follows:

signal (s) : IF (one or more process are waiting on S)
             THEN (let one of these process proceed)
             ELSE $\{ S_r = S + 1$

v) The two operations, wait and signal are done as single indivisible atomic operation. It means, once a semaphore operation has initiated, no other process can access

Question no. :   Q. 2(c)

the semaphore until operation has finished.
Mutual exclusion on the semaphore S is enforced
within wait (S) and signal (S).

Question no.: Q.2(D)

→ (i) Size :- The current size of the file.

(ii) Type :- Information is needed for systems that support different types of files.

(iii) Creator :- ID of the person who created the file.

(iv) Owner :- Current owner of the file.

(v) Password :- Password needed to access the file.

(vi) Hidden flag :- 0 for normal; 1 for do not display in listing.

(vii) Read-only flag :- 0 for read/write; 1 for read only.

(viii) Lock flags :- 0 for unlocked; non-zero for locked

(ix) Key length :- Number of bytes in the key field.

(x) System flag :- 0 for normal files, 1 for system files.