**Multitasking (Time sharing­)**

Multitasking refers to term where multiple jobs are executed by the CPU simultaneously by switching between them. Switches occur so frequently that the users may interact with each program while it is running .

Time sharing or (multitasking) is a logical extension of multi-programming. Time sharing systems, the CPU executes multiple jobs by switching among them, but the switches occur so frequently that the users can interact with each program while its running. Time sharing requires an interactive or (hands on) computer system, which provides direct communication between the user and the system. The user gives instructions to the operating system or to a program directly, using a input device such as a keyboard or a mouse, and waits for immediate results on an output device. Accordingly, the response time should be short typically less than one second. A time-shared operating system allows many users to share the computer simultaneously. Since each action or command in a time-shared system tends to be short, only a little CPU time is needed for each user. As the system switches rapidly from one user to the next, each user is given the impression that the entire computer system is dedicated to his use, even though it is being shared among many users.

A time-shared operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer. Each user has at least one Separate program in memory. A program loaded into memory and executing is called a Process . When a process executes, it typically executes for only a short time before it either finishes or needs to perform I/0. I/0 may be interactive; that is, output goes to a display for the user, and input comes from a user keyboard, mouse, or other device. Since interactive I/0 typically runs at "people speeds," it may take a long time to complete. Input, for example, may be bounded by the user's typing speed; seven characters per second is fast for people but incredibly slow for computers. Rather than let the CPU sit idle as this interactive input takes place, the operating system will rapidly switch the CPU to the program of some other user.

Time sharing and multiprogramming require that several jobs be kept simultaneously in memory. If several jobs are ready to be brought into memory, and if there is not enough room for all of them, then the system must choose among them. Making this decision is job scheduling . When the operating system selects a job from the job pool, it loads that job into memory for execution. Having several programs in memory at the same time requires some form of memory management. In addition, if several Jobs are ready to run the same time, the system must choose among them. Making this decision is CPU scheduling. Finally, running multiple jobs concurrently requires that their ability to affect one another be limited in all phases of the operating system, including process scheduling, disk storage, and memory management. These considerations are discussed later.

In a time-sharing system, the operating system must ensure reasonable response time, which is sometimes accomplished through swapping where processes are swapped in and out of main memory to the disk. *A* more common method for achieving this goal is virtual memory, a technique that allows the execution of a process that is not completely in memory. The main advantage of the virtual-memory scheme is that it enables users to run programs that are larger than actual



Operating system does the following activities related to multitasking :

* The user gives instructions to the operating system or to a program directly, and receives an immediate response.
* Operating System handles multitasking in the way that it can handle multiple operations / executes multiple programs at a time.
* Multitasking Operating Systems are also known as Time-sharing systems.
* These Operating Systems were developed to provide interactive use of a computer system at a reasonable cost.
* A time-shared operating system uses concept of CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared CPU.
* Each user has at least one separate program in memory.
* A program that is loaded into memory and is executing is commonly referred to as a process.
* When a process executes, it typically executes for only a very short time before it either finishes or needs to perform I/O.
* Since interactive I/O typically runs at people speeds, it may take a long time to completed. During this time a CPU can be utilized by another process.
* Operating system allows the users to share the computer simultaneously. Since each action or command in a time-shared system tends to be short, only a little CPU time is needed for each user.
* As the system switches CPU rapidly from one user/program to the next, each user is given the impression that he/she has his/her own CPU, whereas actually one CPU is being shared among many users.

**Interactivity**

Interactivity refers that a User is capable to interact with computer system. Operating system does the following activities related to interactivity:

* OS provides user an interface to interact with system.
* OS managers input devices to take inputs from the user. For example, keyboard.
* OS manages output devices to show outputs to the user. For example, Monitor.
* OS Response time needs to be short since the user submits and waits for the result.

**Real Time System**

Real time systems represents are usually dedicated, embedded systems. Operating system does the following activities related to real time system activity.

* In such systems, Operating Systems typically read from and react to sensor data.
* The Operating system must guarantee response to events within fixed periods of time to ensure correct performance.

**Distributed Environment**

Distributed environment refers to multiple independent CPUs or processors in a computer system. Operating system does the following activities related to distributed environment:

* OS Distributes computation logics among several physical processors.
* The processors do not share memory or a clock.
* Instead, each processor has its own local memory.
* OS manages the communications between the processors. They communicate with each other through various communication lines.