* Operating System

An OS is a program that controls the execution of application programs and acts as an interface between applications and the computer hardware.

* Objectives:

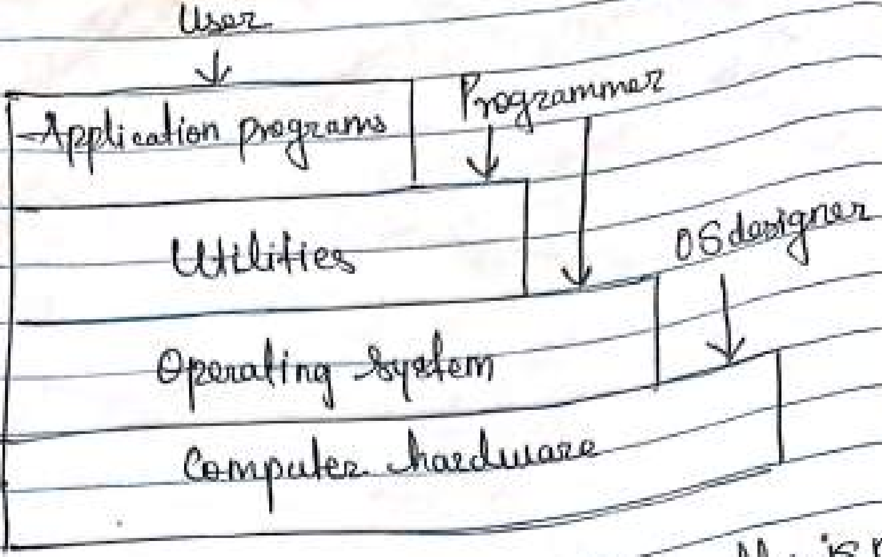
1) Convenience : An OS makes a computer more convenient to use.

2) Efficiency : An OS allows the computer system resources to be used in an efficient manner.

3) Ability to evolve : An OS should be constructed in such a way as to permit the effective development, testing, and introduction of new system functions without interfering with service.

* OS as a User/Computer Interface

The hardware and software used in providing applns to a user can be viewed in a layered or hierarchical fashion, as depicted in fig below:



The user of those appl^{ns}, the end user, generally is not concerned with the details of computer hardware. Thus, the end user views a computer system in terms of a set of appl^{ns}. System programs or utilities assist in program creation, the management of files, & the control of I/O devices.

A programmer will make use of these facilities in developing an application, & the application, while it is running, will invoke the utilities to perform certain functions. The most imp. collection of system programs comprises the OS. The OS masks the details of the h/w from the programmer & provides the programmer with a convenient interface for using the system.

Briefly, the OS typically provides services in the following areas:

1) Program development: The OS provides a variety of facilities & services, such as editor and debuggers, to assist the programmer in creating programs.

2) Program execution: A no. of steps need to be performed to execute a program. Instructions and data must be loaded into main memory, I/O devices & files must be initialized, & other resources must be prepared. The OS handles these scheduling duties for the user.

3) Access to I/O devices: Each I/O device requires its own peculiar set of instructions or control signals for operation. The OS provides a uniform interface that hides these details so that programmers can access such devices using simple reads & writes.

4) Controlled access to files: OS must reflect a detailed understanding of not only the nature of the I/O device but also the structure of the data contained in the files on the storage medium.

5) System Access: The OS controls access to the system as a whole & to specific system resources. The access function must provide protection of resources & data from unauthorized users & must resolve conflicts for resource contention.

6) Error detection & response: OS must provide a response that clears the error condition with the least impact on running applns. The response may range from ending the program that caused the error, to retrying the operation, to simply reporting the error to the applns.

7) Accounting: A good OS will collect usage statistics for various resources & monitor performance parameters such as response time.

* OS as Resource Manager.

A computer is a set of resources for the movement, storage, & processing of data & for the control of these functions. The OS is responsible for managing these resources. By managing the computer's resources the OS is in control of the computer's basic function in two ways:

functions in the same way as ordinary computer S/w; that is it a program or suite of programs executed by the processor.

- The OS frequently relinquishes control & must depend on the processor to allow it to regain control.

* Evolution of OS

1) Serial processing → 1940's - 1950's programmer interacted directly with h/w. No OS.

problems:

Scheduling → users sign up for m/c time. wasted computing time

Setup Time → Setup ~~the~~ included loading the compiler, source program, saving compiled program & loading & linking If an error occurred start over

2) Simple Batch systems → Improve the utilization of computer.

Jobs were submitted on cards or tape to an operator who batches jobs together sequentially. The program that controls the execution of the jobs was called monitor - a simple version of an OS.

The interface to the monitor was accomplished through Job Control Lang (JCL).

For eg: a JCL request could be to run the compiler for a particular programming lang, then to link & load the program, then to run the user prog.

H/w features:

Memory protection: do not allow the memory area containing the monitor to be altered

Timer: prevents a job monopolizing the system.

Prob: Bad utilization of CPU time - the processor stays idle while I/O deuices are in use

3) Multiprogrammed Batch Systems:

More than one program resides in the main memory While a program A uses an I/O device the processor does not stay idle, instead it runs another program B.

Program A

Run $\xrightarrow{\text{Wait}}$ Run $\overline{\text{Wait}}$

Time ⟶

(a) Uniprogramming

Program A Run $\xrightarrow{\text{Wait}}$ Run $\xrightarrow{\text{Wait}}$

Program B Wait Run $\xrightarrow{\text{Wait}}$ Run $\xrightarrow{\text{Wait}}$

Combined Run A|Run B| $\xrightarrow{\text{Waft}}$ Run A|RumB $\xrightarrow{\text{Wait}}$

Time ⟶

(b) Multiprogramming with two programs.

New features:

Memory management - to have several jobs ready to run, they must to kept in MM.

Job scheduling - the processor must decide which program to run.

4) Time-Sharing Systems:

Multiprogramming systems: several programs use the computer system

Time-sharing systems: several (human) uses use the computer system interactively.

Characteristics:

- Using multiprogramming to handle multiple interactive jobs.