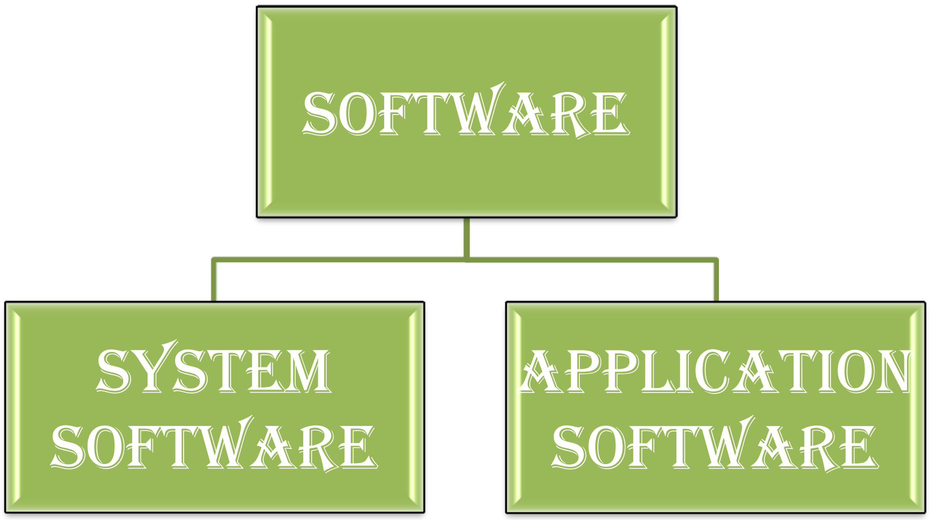
Software

Software: is a general term for the various kinds of programs used to operate computers and related devices. Software can be thought of as the variable part of a computer and hardware the invariable part. Software is often can be divided into:



1. ***System software*** is responsible for controlling, integrating, and managing the individual hardware components of a computer system so that other software and the users of the system see it as a functional unit without having to be concerned with the low-level details such as transferring data from memory to disk, or rendering text onto a display. Generally, system software consists of an operating system and some fundamental utilities such as disk formatters, file managers, display managers, text editors, user authentication (login) and management tools, and networking and device control software.
2. ***Application software***, on the other hand, is used to accomplish specific tasks other than just running the computer system. Application software may consist of a single program, such as an image viewer; a small collection of programs (often called a software package) that work closely together to accomplish a task, such as a spreadsheet or text processing system, or a programming package (ex. C++, FORTRAN packages).

   A larger collection (often called a software suite) of related but independent programs and packages that have a common user interface or shared data format, such as Microsoft Office, which consists of closely integrated word processor, spreadsheet, database, Presentation, etc.; or a software system, such as a database management system, which is a collection of fundamental programs that may provide some service to a variety of other independent applications.

The term ***Middleware*** is sometimes used to describe programming that mediates between application and system software or between two different kinds of application software (for example, sending a remote work request from an application in a computer that has one kind of operating system to an application in a computer with a different operating system**).**

An additional and difficult-to-classify category of software is the ***utility***, which is a small useful program with limited capability. Some utilities come with operating systems. But applications utilities tend to be separately installable and capable of being used independently from the rest of the operating system.

Operating System

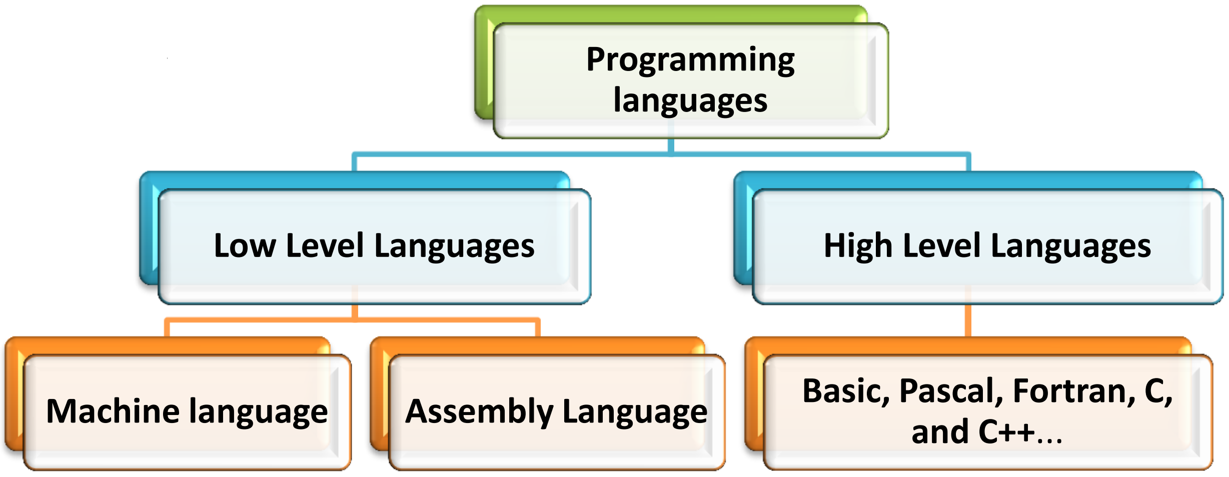
It is a type of software program that is responsible for the management and coordination of activities and the sharing of the limited resources of the computer. Without it the computer is useless. Examples of OS are: DOS, Windows, UNIX and Mac OS.

Every operating system performs a **variety of functions** as illustrated below:

- Managing Resources: it coordinates all computer resources including memory, processing, storage, and devices.
- Organizing data transformation between different devices of computer.
- Controlling input and output devices such as the way information is displayed on the screen, how it is printed, and how the mouse works.
- Control how data is written on disks.
- Providing a group of common functions, that other program can utilize.

Programming Language

Programming languages contain the series of commands that create software. In general, a language that is encoded in binary numbers or a language similar to binary numbers that a computer's hardware understands is understood more quickly by the computer. A program written in this type of language also runs faster. Languages that use words or other commands that reflect how humans thinks are easier for programmers to use , but they are slower because the language must be translated first so the computer can understand it. Programming languages can be divided in to the following:



Low Level Languages (L.L.L).

A. Machine language:

Computer programs that can be run by a computer's operating system are called executable. An executable program is a sequence of extremely simple instructions known as machine code. These instructions are specific to the individual computer's CPU and associated hardware; for example, Intel Pentium and Power PC microprocessor chips each have different machine language and require different sets of codes to perform the same task. Typical instructions are for copying data from a memory location or for adding the contents of two memory location. Machine code instructions are binary that is, sequences of bit (0s and 1s). Because these numbers are not understood easily by humans, computer instructions usually are not written in machine code.

B. Assembly Language:

Assembly language uses commands that are easier for programmers to understand than the machine-language commands. Each machine language instruction has an equivalent commands in assembly language. For example ,in assembly language , the statement "MOV A,B " instructs the computer to copy data from one location to another. The same instruction in machine code is a string of 16 0s and 1s. Once an assembly-language program is written, it is converted to a machine-language program by another program called an **assembler**. Assembly language is fast and powerful because of its correspondence with machine language. However, it is still difficult to use, because assembly-language instruction are a series of abstract codes. In addition, different CPUs use different machine languages and therefore require different assembly language. Assembly language is sometimes inserted into a higher-level language program to carry out specific hardware tasks or to speed up a higher-level program

The Higher Level Languages (H.L.L.)

Higher Level languages were developed because of the difficulty of programming assembly language. Higher level languages are easier to use than machine and assembly languages because their commands resemble natural human language. languages that use words or other commands that reflect how humans thinks are easier for programmers to use, but they are slower because the language must be translated first so the computer can understand it. This can be done by software called compiler. In addition, these languages are not CPU specific. Instead, they contain general commands that work on different CPUs. A compiler turns a higher-level program into a CPU specific machine language. Examples of H.L.L are Basic, Pascal, FORTRAN, C++, etc...

Complier

It is a program translator that translates the instruction of higher level languages to machine language. Thus, compiler is program translator like assembler but more sophisticated. It scans the entire program first and then translates it into machine code.

Interpreter

An interpreter is another type of program translator used for translating H.L.L into machine languages. It takes one statement of program translated and immediately executes it. Translation and executions are carried out for each statement.

The advantage of it compared to compiler is its fast response to changes in source program. It eliminates the need for a separate compilation after changes to each program. Interpreters are easy to write and do not require large memory in computer.

The disadvantage of interpreter is that it is time consuming method because each time a statement in program is executed then it is first translated. Thus, compiled machine languages program runs much faster than an interpreted program.