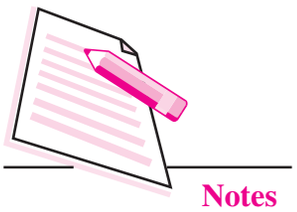


Notes

3

COMPUTER SOFTWARE

In the previous lesson, you have learnt about binary logic. Now you will be able to understand the working of computer. As you know that computer is a machine which cannot do anything without instructions from the user. In order to do any specific job you have to give a sequence of instructions to the computer. These set of instructions are called as programs and these set of computer programs are called as software. In this lesson, you will learn about computer software, computer languages and translators.

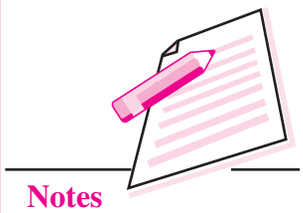
OBJECTIVES

After reading this lesson, you will be able to:

- define software;
- distinguish between system software and application software;
- identify various levels (or generations) of language such as machine language, assembly language, high-level language;
- compare compilers and interpreter.

3.1 WHAT IS SOFTWARE

Computer system is broadly classified as hardware and software. Hardware means physical components of the computer system. We can say that hardware cannot perform any calculation, comparison or manipulation without being instructed to do so. These instructions play a vital role in the performance of a computer. A complete set of instructions written to solve a problem on a computer is called software. i.e., software refers to the set of computer programs that cause the hardware (computer system) to function in the desired manner.



Computer software is normally classified into two broad categories:

(i) System Software

(ii) Application Software

3.1.1 System Software

System software includes general programs written for a computer. It consists of pre-written programs and documentation supplied by the manufacturer along with the computer. These programs are held permanently in the machine. The primary objectives of this software are to:

(i) enhance the efficiency of hardware utilization, and

(ii) make computers simple to use.

The first objective is realized by that component of system software which is commonly known as operating system. System software is a set of instruction to the machine to interpret and execute application software. For example, language translators (called compilers and interpreters), operating systems, utilities and special purpose software.

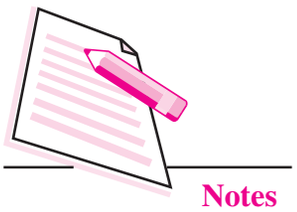
Operating System

An operating system is the most important system software and it is an essential software to operate the computer system. An operating system manages computer's resources very effectively, takes care of scheduling multiple jobs for execution and manages the flow of data and instructions between the input/output units and the main memory.

Operating system like Microsoft Windows, Linux are examples of system software.

The first operating system, called batch processing operating system was developed for the second generation computers. This operating system executes jobs serially one after another from a batch of jobs submitted for execution. The central processing unit is kept busy only during the processing cycle of a job and remains idle during the input and output operations. A multi programming operating system handles multiple jobs simultaneously by overlapping the input/ output and processing cycles of various jobs.

Other types of operating system which are popular today are 'multi-processing operating systems' and 'real time operating systems'. A multi-processing operating system uses multiple CPUs to process multiple jobs. A real time operating system is a type of interactive operating system with strict time limitation. Receiving and processing data is done quickly enough to produce output, to control, direct or effect the outcome of ongoing activity. The reservation system used by railway, airlines, and hotel are the examples of real time applications.



Notes

An operating system governs the working of the computer and input/output devices. The operating system programs act as an interface between the user's programs and the computer's components and helps in the execution of user's programs. The major functions of an operating system are:

(i) User identification and managing of the resources used by the users. Thus un-authorized users cannot use the computer.

(ii) Sharing of computer resources among many users. The sharing is achieved by permitting simultaneous executions of more than one user program. This is usually called multi-programming. A mix of programs can keep the whole memory occupied, all devices active, and the control unit and ALU constantly busy, thus increasing utilization of hardware.

(iii) Executive batches of programs, one after another, without human intervention.

(iv) Protection of user's data and programs.

(v) Controlling the transfer of data and programs between the main memory and secondary storage and other I/O devices.

(vi) Providing programs to select appropriate translators.

(vii) Providing facilities to detect and correct errors in a user's program.

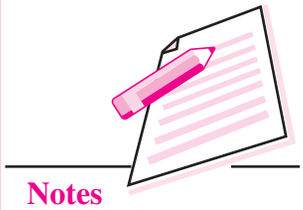
An operating system understands a fixed set of commands. This set of commands is often called Job Control Language (JCL). The JCL commands are used by the computer users to indicate their requirements to the operating system. The operating system which is used with a micro-computer is called CP/M (control program for microprocessor). Another operating system which is gaining popularity and which is available on a variety of different machines is UNIX (UNIX was developed by Bell laboratories). You will be learning more about operating system in the lesson 4.

Utility Software

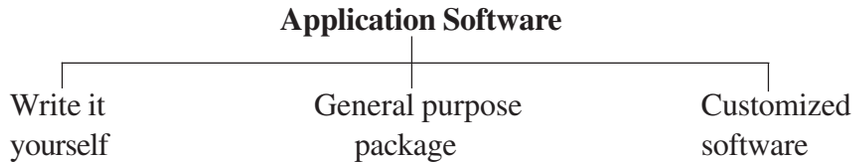
Utility software may be considered as system software which is used quite often in the development of a program. Sort merge programs are used to sort records in a particular manner or sequence. Such programs are normally provided by the manufacturers. The programmer can also develop his/her own utility software and keep it in the secondary memory of the computer.

3.1.2 Application Software

Application software is written to perform a specific task or process, such as accounting, payroll, mailing list, result preparation and so on. The majority of application software is written in high-level languages.



Assuming that the task to be carried out has been correctly identified, carefully defined, the prospective user will come across the following alternative sources for this application software.



(a) **Write it yourself** : The program written by the user can be the most satisfactory solution. It will be an exact match to the needs of the business. The program can grow with the business.

(b) **General purpose application packages** : Application packages refer to a set of computer programs, which have been written to perform specific, commonly required tasks. Each program is written in such a way that it is applicable to a large number of users. The main advantage is that it is relatively cheap as cost of the package is spread over a number of customers.

The major disadvantage of application package is that it is not likely to fulfil all the requirements of the prospective users.

(c) **Customized software** : It refers to computer programs specially written to match the exact needs of the user. The most important advantage is that such software fulfils all the needs of the customer. The major disadvantage is that customised software costs much more than general purpose application software, because the package is specially made for one particular customer.

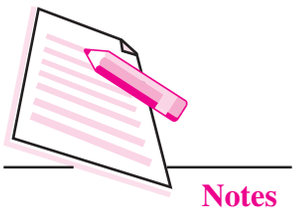
Common Application Packages

Some of the common requirements of the users of personal computers have been identified and common applications packages have been developed to meet their needs. These packages include word processor, database processing, spreadsheet calculations, mail-merge, presentations and communications (email). These packages have been prepared so that they are simple to use. They also provide graphical user interface to make them very user friendly. These packages are readily available in the market and one can purchase them, install it on his/her computer easily and start using the package.

INTEXT QUESTIONS 3.1

1. State whether the following statements are true or false.

   (a) Computer software is normally classified into two broad categories. Those are system software and application software.



Notes

(b) An operating system is an essential software to operate a computer system.

(c) JCL stands for Job Common Language.

(d) Application software is written to perform a specific task or process.

(e) Utility software cannot be considered as an application software.

3.2 COMPUTER LANGUAGE

You know that software is a set of computer programs. These computer programs will be written in the language what computer understands. If you want to get something done by a person, you will tell him what to do in a language that he understands. Similarly, if you want to get some work done by the computer, you have to tell the computer in a language that the computer understands, i.e., machine language. The machine language consists of only binary digits, i.e. 0 and 1. It was felt quite difficult and tedious for human beings to think in binary numbers. For communicating with the computer, it was thought that it is advisable to develop a third language, a programming language that can be understood by both human beings and the computer. Thus a programming language is a set of rules that provides a way of instructing the computer to perform certain operations.

Programming languages are said to be lower or higher, depending on whether they are closer to the language the computer itself uses (lower, which means 0s and 1s) or to the language that people use (higher, which means more English like).

The languages in which programs are written are called programming languages. These languages can be classified into following categories.

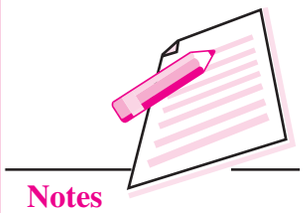
- Machine language
- Assembly language
- High level language

Machine Languages

Do you know that computer can understand 0's and 1's? Yes 0's and 1's can be directly understood by computers. Programs that have only binary digits are called a machine language programs. It is difficult to write or understand machine language. Letters of the alphabet are also represented in binary numbers. In one system, the letter A is represented as 1000001. Commas, semicolons and other special characters are also represented as bunches of 0s and 1s.

Assembly Languages

In the 1950s, to reduce programming complexity and provide some standardization, assembly languages were developed. Assembly languages, also known as symbolic



Notes

languages use abbreviations or memonic code - codes that are more easily memorized to replace the 0s and 1s of machine languages.

The hypothetical machine language segment we saw above is as follows:

11110010 01110011 11010010 00010000 01110000 00101011

This could be expressed in assembly language statement as :

PACK 210 (8, 13), 02B (4,7)

Actually, assembly languages do not replace machine languages. In fact, for an assembly language program to be executed, it must be converted to machine code. The assembly language program is referred to as a 'source program' whereas, the machine language program is an 'object program'.

Assembly language code is very similar in the form to machine language code. In fact, assembly languages had a one-to-one correspondence which means that 15 assembly statements, would be translated into 15 machine language statements. This one-to-one correspondence was still laborious. However, assembly language instructions (called macro instructions) were devised, which executed batches of one-to-one instructions.

Assembly languages offer several advantages:

- They are more standardized and easier to use than machine languages.
- They operate very efficiently, although not as efficient as the machine languages.
- They are easier to debug.

However, there are still some disadvantages:

- Assembly language programs are usually very long and difficult to debug.
- Though less abstract than machine languages, assembly language programs are still complex.
- Though more standardized than machine languages, assembly languages are still machine dependent.

High Level Languages

High Level Languages helped programmers by reducing further the number of computer operations details they had to specify, so that they could concentrate more on the logic needed to solve the problem.

Basic, C, C++, Java, Perl and COBOL are the example of high level language (HLL)

Languages are often referred to as generations, the idea being that machine languages were the first generation and assembly languages were the second

Notes

generation. High-level languages are sometimes used to refer all languages above the assembly level. Here we will subdivide high-level languages into three generation.

- Procedural-oriented or third generation
- Problem-oriented or fourth generation
- Natural or fifth generation

Procedural-oriented Languages

High-level languages are often classified according to whether they solve general problems or specific problems. General-purpose programming languages are called procedural languages or third generation languages. These are languages such as Pascal, BASIC, COBOL, and FORTRAN, which are designed to express the logic, the procedure, of a problem. Because of their flexibility, procedural languages are able to solve a variety of problems.

Procedural languages have many advantages over machine and assembly languages:

- The program statements resemble English language and hence are easier to work with.
- Because of their English-like nature, less time is required to develop a program for a given problem.
- Once coded, programs are easier to understand and modify.
- The procedural languages are machine-independent.

However, procedure-oriented languages still have some disadvantages compared to machine and assembly languages:

- Programs are executed more slowly.
- The languages use computer resources less efficiently.

Problem-oriented Languages and Application Generators

Third-generation languages, such as BASIC or Pascal, require you to instruct the computer in step-by-step fashion. Fourth-generation languages, also known as problem-oriented languages, are high-level languages designed to solve specific problems or develop specific applications by enabling you to describe what you want rather than step-by-step procedures for getting there.

Table 3.1 summarizes some of the major difference between third-generation languages (3GLs) and fourth-generation languages (4GLs).

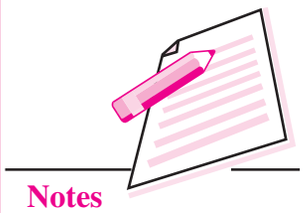


Table 3.1: Difference between 3 GLs and 4 GLs

Third-generation Languages	Fourth-generation Languages
Intended for use by professionals	May be used by a non-programming personnel as well as a professional programmer.
Requires specification of how to perform task.	Requires specification of what task is to be performed (system determines how to perform the task).
All alternatives must be specified.	Default alternatives are built in; an end user need not specify these alternatives
Require large number of procedural instructions.	Require fewer instructions.
Code may be difficult to read, understand and maintain.	Code is easy to understand and maintain because of English-like commands.
Language developed for batch operations.	Language developed primarily for online use.
Can be difficult to learn.	Easy to learn.
Difficult to debug.	Easy to debug.
Typically file-oriented	Typically database-oriented.

Natural Languages

Natural languages are still in the developmental stages, but they promise to have profound effect, particularly in the areas of artificial intelligence and expert systems. Natural languages have two characteristics:

- They are designed to make the connections that humans have with computers more natural - more human like.
- They are designed to allow the computer to become "smarter" - to actually simulate the learning process by remembering and improving upon earlier information.

Two popular natural languages are LISP and PROLOG.

3.3 COMPILERS AND INTERPRETERS

For a high-level language to work on the computer it must be translated into machine language. There are two kinds of translators - compilers and interpreters. High-level languages are called either compiled languages or interpreted languages. In a compiled language, a translation program is run to convert the high-level

language program (which is called the source code) into a machine language code. This translation process is called compilation. The machine language code is called the object code and can be saved and either run (executed) immediately or later. Some of the most widely used compiled languages are COBOL, C, C ++, FORTRAN, etc.

In an interpreted language, a translation program converts each program statement into machine code just before the program statement is to be executed. Translation and execution occur immediately, one after another, one statement at a time.

Unlike the compiled language, no object code is stored and there is no compilation. The most frequently used interpreted language is BASIC.

Compiled languages are better than interpreted languages as they can be executed faster and more efficiently once the object code has been obtained. On the other hand, interpreted languages do not need to create object code and so are usually easier to develop and test.

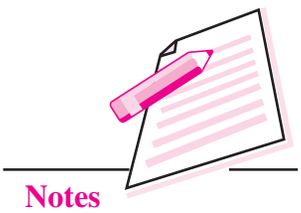
Compiler reads the entire program for compilation. Interpreter reads single statement at a time for interpretation.

INTEXT QUESTIONS 3.2

1. State whether the following statements are true or false:
   (a) Machine language is machine-dependent.
   (b) Assembly language is second generation language.
   (c) A program written in a high level language is called an object program.
   (d) Portability is one of the features of high level language.
   (e) Query languages allow people who are not programmers to search a database using certain selection commands.

WHAT YOU HAVE LEARNT

- A complete set of instructions written to solve a problem on a computer is called software. i.e., software refers to the set of computer programs that cause the hardware (computer system) to function in the desired manner.
- Computer software is normally classified into two broad categories (i) system software and (ii) application software.
- System software includes general programs written for a computer.
- Application software is written to perform a specific task or process.



- Programming languages can be classified as machine language, assembly language and high-level language.
- Machine language and assembly language programs are difficult to write and read. High level languages are easy to learn and write.
- Compiler and interpreter convert the program written in high level language into machine language code.
- A translation program is run to convert the high-level language program, which is called the source code, into a machine language code. This translation process is called compilation.

TERMINAL EXERCISE

1 Write any one difference between hardware and software.

2 What are the advantages and disadvantages of machine language?

3 What is the difference between source program and object program?

4 Explain assembly language. What are its advantages over machine language?

5 Define operating system. Write down its various functions.

6 Describe briefly about application software.

ANSWERS TO INTEXT QUESTIONS

3.1

1. (a) True (b) True (c) False (d) True (e) True

3.2

1. (a) True (b) True (c) False (d) True (e) False