UNIT-I

Computer programming Language

"The computer language is defined as code or syntax which is used to write programs or any specific applications."

It is used to communicate with computers. Broadly the computer language can be classified into three categories:- assembly language, machine language, and high-level language. For computer language processing the system needs a compiler and interpreter to convert the language into computer language so that it can be processed by a machine.

Different Types of Computer Language:

1. Machine Language

The machine language is considered as oldest computer language among all three. In machine language, the input is directly given as binary input which is processed by the machine. Binary inputs mean one and zero form. The machine language is sometimes referred to as machine code or object code which is a set of binary digits 0 and 1. These binary digits are understood and read by a computer system and interpreted easily. It is considered a native language as it can be directly understood by a central processing unit (CPU). The machine language is not so easy to understand, as the language uses the binary system in which the commands are written in 1 and 0 form which is not easy to interpret. There is only one language that is understood by computer which is machine language. The operating system of the computer system is used to identify the exact machine language used for that particular system.

The operating system defines how the program should write so that it can be converted to machine language and the system takes appropriate action. The computer programs and scripts can also be written in other programming languages like C, C++, and JAVA. However, these languages cannot be directly understood by a computer system so there is a need for a program that can convert these computer programs to machine language. The compiler is used to convert the programs to machine language which can be easily understood by computer systems. The compiler generates the binary file and executable file.

Example of machine language:

01001000 0110101 01101100 01101100 011011

2. Assembly Language

The assembly language is considered a low-level language for microprocessors and many other programmable devices. The assembly language is also considered a second-generation language. The first generation language is machine language. The assembly language is mostly famous for writing an operating system and also in writing different desktop applications. The operations carried out by programmers using assembly language are memory management, registry access, and clock cycle operations. The drawback of assembly language is the code cannot be reused and the language is not so easy to understand. The assembly language is considered a group of other languages. It is used to implement the symbolic representation of machine code which is used to program CPU architecture. The other name of assembly language is assembly code. For any processor, the most used programming language is assembly language.

In assembly language, the programmer does the operation which can be directly executed on a central processing unit (CPU). The language has certain drawbacks as it does not contain any variables or functions in programs and also the program is not portable on different processors. The assembly language uses the same structure and commands which machine language does use but it uses names in place of numbers. The operations performed using the assembly language are very fast. The operations are much faster when it is compared to high-level language.

3. High-Level Language

The development of high-level language was done when the programmers face the issue of writing programs as the older language has portability issues which means the code written in one machine cannot be transferred to other machines. This led to the development of high-level language. The high-level language is easy to understand and the code can be written easily as the programs written are user-friendly in a high-level language. The other advantage of code written in a high-level language is the code is independent of a computer system which means the code can be transferred to other machines. The high-level of language uses the concept of abstraction and also focuses on programming language rather than focusing on computer hardware components like register utilization or memory utilization.

The development of higher-level language is done for a programmer to write a human-readable program that can be easily understood by any user. The syntax used and the programming style can be easily understood by humans if it is compared to low-level language. The only requirement in a high-level language is the need for a compiler. As the program written in a high-level language is not directly understood by the computer system. Before the execution of high-level programs, it needs to be converted to machine-level language. Examples of high-level languages are C++, C, JAVA, FORTRAN, Pascal, Perl, Ruby, and Visual Basic.

C: The C is a procedural and general-purpose programming language used for writing programs. This language is mostly used for writing operating system applications and desktop applications.

PASCAL: Pascal is a procedural programming language that is based on data structures. It uses the concept of recursive data structures such as graphs, lists, and graphs.

Translator

"A program written in high-level language or assembly language is called as source code. To convert the source code into machine code, translators are needed."

A translator takes a program written in source language as input and converts it into a program in target language as output.It also detects and reports the error during translation.

Roles of translator are:

• Translating the high-level language program or assembly language input into an equivalent machine language program.

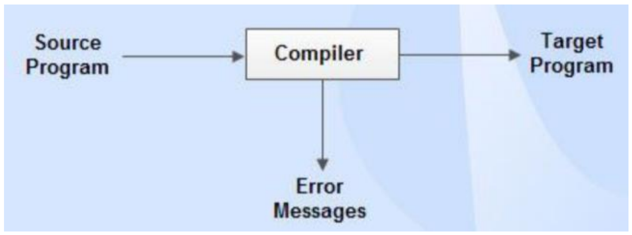
• Providing diagnostic messages wherever the programmer violates specification of the high-level language program.

Different type of translators

The different types of translator are as follows:

1. Compiler

Compiler is a translator which is used to convert programs in high-level language to low-level language. It translates the entire program and also reports the errors in source program encountered during the translation.



Compilers have several advantages:

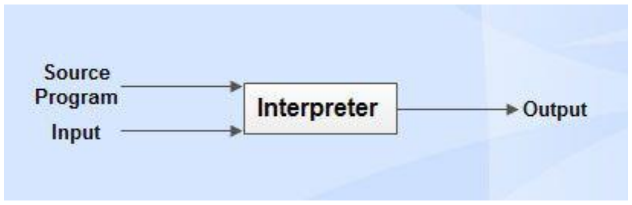
- Compiled programs run quickly since they have already been translated.
- A compiled program can be supplied as an executable file. An executable file is a file that is ready to run. Since an executable file cannot be easily modified, programmers prefer to supply executables rather than source code.
- Compilers optimise code. Optimised code can run quicker and take up less memory space

Compilers have several disadvantages:

- Because the source code is translated as a whole, there must be enough memory space to hold the source code, the compiler and the generated object code. There also needs to be temporary working space for the compiler to perform the translation. Modern systems either have enough memory or use virtual memory to hold all the data.
- Compilers do not usually spot errors - the program has to be compiled and run before errors are encountered. This makes it harder to see where the errors lie.
- The source code must be recompiled every time the programmer changes the program.
- Source code compiled on one platform will not run on another - the object code is specific to the processor's architecture.

2.Interpreter

Interpreter is a translator which is used to convert programs in high-level language to low-level language. Interpreter translates line by line and reports the error once it encountered during the translation process.



Interpreters have several advantages:

- Instructions are executed as soon as they are translated.
- Since instructions are executed once translated, they are not stored for later use. As a result, interpreters require less available memory.
- Errors can be spotted quickly. Once an error is found, the program stops running and the user is notified at which part of the program the interpretation has failed. This makes interpreters extremely useful when developing programs.

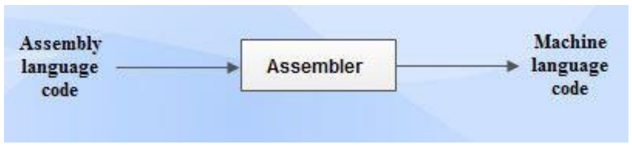
Interpreters also have several disadvantages:

- Interpreted programs run more slowly. The processor has to wait for each instruction to be translated before it can be executed.
- Additionally, the program has to be translated every time it is run.

- Interpreters do not produce an executable file that can be distributed. As a result, the source code program has to be supplied and this could be modified without permission.
- Interpreters do not optimise code - the translated code is executed as it is.

3. Assembler

Assembler is a translator which is used to translate the assembly language code into machine language code.


Assembly language code
Assembler
Machine language code