1. PROGRAMMING LANGUAGES

Introduction

Language is a medium of communication. It is used to share ideas, information, feelings and opinion to each other. In day to day life we use natural languages such as English, Hindi, Marathi etc. we use these languages to communicate with people but to communicate with computer system we need to use special language as natural languages cannot be understood by computer system.

Programming Language

A programming language is a **computer language** that is used by **programmers (developers) to communicate with computers**. It is a set of instructions written in any specific language to perform a specific task. Every computer programming language contains a set of predefined words and a set of rules (syntax) that are used to create instructions of a program.

For Ex:- C, C++, Java, python etc.

The Computer programming languages are classified as –

1) Low Level Language
2) High Level Language
3) Middle Level Language

1) Low Level Language

Low level language is also known as Machine Language. Low level language is divided into two parts that are Machine language and Assembly language.

1.1 Machine Language

It is also called as a binary language. It consist of only two digits that are zero and one. A computer can directly understand the binary language. Low Level language is as the First generation language.

1.2 Assembly Language

Assembly language is similar to that of machine language. This language consist of some English words instead of 0's and1's.In assembly language, we use predefined words called mnemonics. A program written is an assembly language using mnemonics called assembly language program or symbolic program. Programs written in assembly language cannot be directly understood by computer so it required some additional software which converts assembly language to machine language and from machine language to assembly language.

Advantages

- A computer can easily understand the low level language.
- Low level language instructions are executed directly without any translation.
- Low level language instructions require very less time for their execution.

Disadvantages

- Low level language instructions are very difficult to use and understand.
- Low level language instructions are machine dependent, that means a program written for a particular machine does not executes on other machine.

- In low level language, there is very difficult to find errors, debug and modify

2) High Level Language

High level language can be easily understood by the users. It is very similar to the human language and has a set of grammar rules that are used to make instructions more easily. Every high level language has a set of predefined words known as keywords and a set of rules known as syntax. High level language is a programming language. Languages like COBOL, BASIC, FORTRAN, C,C++, JAVA etc. All these programming languages are to write program instructions. These instructions are converted to low level language by the complier or interpreter.

Advantages

- Writing instructions in high level language is easier.
- High level language is readable and understandable.
- High level language program can runs on different machines without any modification.
- It is easy to understand, create programs, find errors and modify.

Disadvantages

- High level language instructions need to be translated to low level language by using compiler or interpreter.
- Slower in execution as compared to low level and middle level language.
- Lack of flexibility.
- Lower efficiency.

3) Middle Level Language

It is a special type of programming language, which consist of advantages of low level language and high level language.

Ex:- C language

Advantages

- Fast Processing
- Simplicity

Programming Language Translator

A programming language translator is a software that translate computer program (instructions) written in some specific programming language into another programming language. A program written in high level language is called source code. To convert the source code into machine code, translators are needed. There are three types of programming language translator –

1) Compliers
2) Interpreters
3) Assemblers
4) Linker and Loader

1) Compliers

Compiler is a translator which is used to convert programs in high level langue to low level language. It translates the entire program that is group of statements at a time and also reports the errors in source program encountered during the translation.

2) Interpreters

Interpreter is a translator which is used to convert programs in high level language to low level language. Interpreter translates line by line statements and reports the error once it encountered during the translation process. It gives better error diagnostics than a compiler.

3) Assemblers

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

4) Linker and Loader

Linker is a computer program that links and merges various object files together in order to make an executable file. All these files might have been compiled by separate assembler. The major task of a linker is to search and locate referenced module/routines in a program and to determine the memory location where these codes will be loaded making the program instruction to have absolute reference. Loader is a part of operating system and is responsible for loading executable files into memory and executes them. It calculates the size of a program (instructions and data) and cerates memory space for it. It initializes various registers to initiate execution.

Differentiate between Compiler and Interpreter

Compiler	Interpreter
1.A compiler is a programming language translator which convert high level language program in to its equivalent code i.e. object code.	1.Aninterpreter is a programming language translator which convert high level language program in to its equivalent code i.e. machine code.
2.Compiler generates an intermediate code.	2.Interpreter generates machine code.
3.Compiler reads entire program for compilation.	3.Interpreter reads single line or statement at a time for interpretation.
4.Compiler displays all errors at a time and without fixing all errors program can't be executed.	4.As interpreter reads single line or statement, so it display one error at a time and we have to fix the error to interpret the next statement.
5.Compiler need more memory because of generation of intermediate code.	5.Interpreter needs less memory because it does not generate intermediate code
6.Execution is faster	6.Execution is slower
7.For Ex. C, C++, Java	7. For Ex. Ruby, Python,MATLAB