Code at Random
(OPC) PVT. LTD.

Types Of Computer Languages

Introduction

- The computer language is defined as **code or syntax** which is used to write programs or any specific applications.
- The computer language is used to communicate with computers.
- Broadly the computer language can be classified into three categories **assembly language, machine language, and high-level language.**
- The machine language is considered as oldest computer language among all three.
- For computer language processing the system needs **compiler and interpreter** to convert the language in computer language so that it can be processed by a machine.

Machine Language

- It is also known as Low-Level Language which can be directly understood by the machine.
- The machine language is sometimes referred to as machine code or object code which is set of binary digits 0 and 1.
- These binary digits are understood and read by a computer system and interpret it easily.
- It is considered a native language as it can be directly understood by a central processing unit (CPU).
- Example of machine language for the text “Hello World”.

  01001000 0110101 01101100 01101100 01101111 00100000 01010111 01101111 01110010 01101100 01100100.

Assembly Language

- The assembly language is considered a low-level language for microprocessors and many other programmable devices.
- It is also considered as second-generation language.
- The assembly language contains some human-readable commands such as mov, add, sub, etc.
- As we know that computers can only understand the machine-level instructions, so we require a translator that converts the assembly code into machine code.
- The translator used for translating the code is known as an **assembler**.
- 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.

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 high-level language is closer to human languages than machine-level languages.
- A compiler or interpreter is required to translate a high-level language into a low-level language.
- The high-level language is easy to read, write, and maintain as it is written in English like words.
- The high-level languages are designed to overcome the limitation of low-level language, i.e., portability. The high-level language is portable; i.e., these languages are machine-independent.
- Example: JAVA, C, Python,etc.

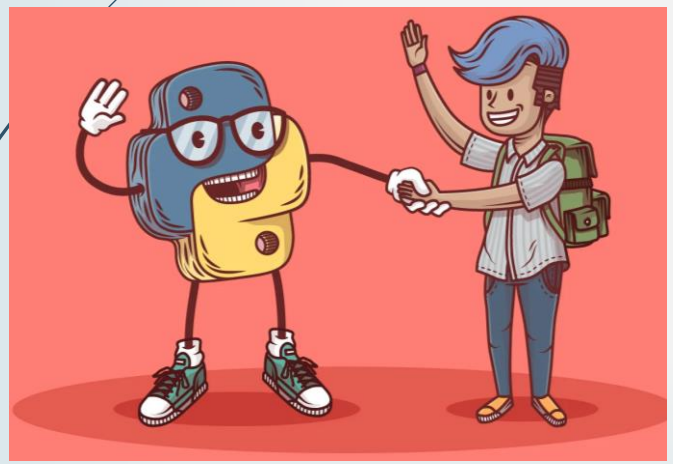
Low-Level Vs High-Level Language

Low-level Language	High-level Language
• It is a machine-friendly language, i.e., the computer understands the machine language, which is represented in 0 or 1.	• It is a user-friendly language as this language is written in simple English words, which can be easily understood by humans.
• The low-level language takes more time to execute.	• It executes at a faster pace.
• It is memory efficient.	• It is less memory efficient.
• Debugging and maintenance are not easier in a low-level language.	• Debugging and maintenance are easier in a high-level language.
• It is a native language as it can be directly understood by a central processing unit (CPU).	• The compiler is used to convert the programs to machine language which can be easily understood by computer systems.

Compiler Vs Interpreter

Compiler	Interpreter
• A compiler is a software program that follows the syntax rule of programming language to convert a high-level language to machine-level language.	• An interpreter is also a software program that translates a high-level language to machine-level language.
• The compiler executes the programming statements all at once and generates an error message if there is any error in the program.	• The interpreter executes the programming statements line-by-line. If an error is found at any specific statement, it stops further execution until the error gets removed.
• As the source code is already converted into machine code, the code execution time becomes short.	• As the source code is interpreted line-by-line, error detection and correction become easy.
• It stores the converted machine code from your source code program on the disk.	• It never stores the machine code at all on the disk.

Happy Learning!!



Code at Random
(OPC)
PVT. LTD.