Low-level programming languages directly deal with the hardware component of the computer. Low-level language is also termed a computer’s native language since the computer can instantly understand it without any extra effort of compiling and interpreting like high-level language. But if the program is written in human-readable form and still in assembly language then we need an assembler to translate it into machine-level language.

Machine language and assembly language are examples of low-level languages.

For example:

LDA 1020

MOV H, A

LDA 1021

ADD H

MOV L, A

MVI A,00

ADC A

MOV H, A

SHLD 2010

HLT

The above program is written in Assembly language which adds two 8-bit numbers.

High-level programming languages are the higher-level abstraction in computer science which are designed and built by focusing on the user’s logic rather than dealing with the hardware of the computer directly. It enables a normal user to develop programs by typing codes in human-readable form. We need tools like compiler and interpreter to convert high-level languages into machine-level languages.

For example, if we start to build an application in a machine-level language which operates very high-level computing then wouldn’t it be a nightmare for the developer?

So, high-level languages enable a developer to ease in writing and maintaining quality code that can be reused again and again.

Python, Java, and Javascript are examples of high-level languages.

Let’s transfer the above program written in the Assembly language into python code.

a=int(input(“Enter first number”);

b=int(input(“Enter second number”)

print(“Sum of the given number is:”,a+b)