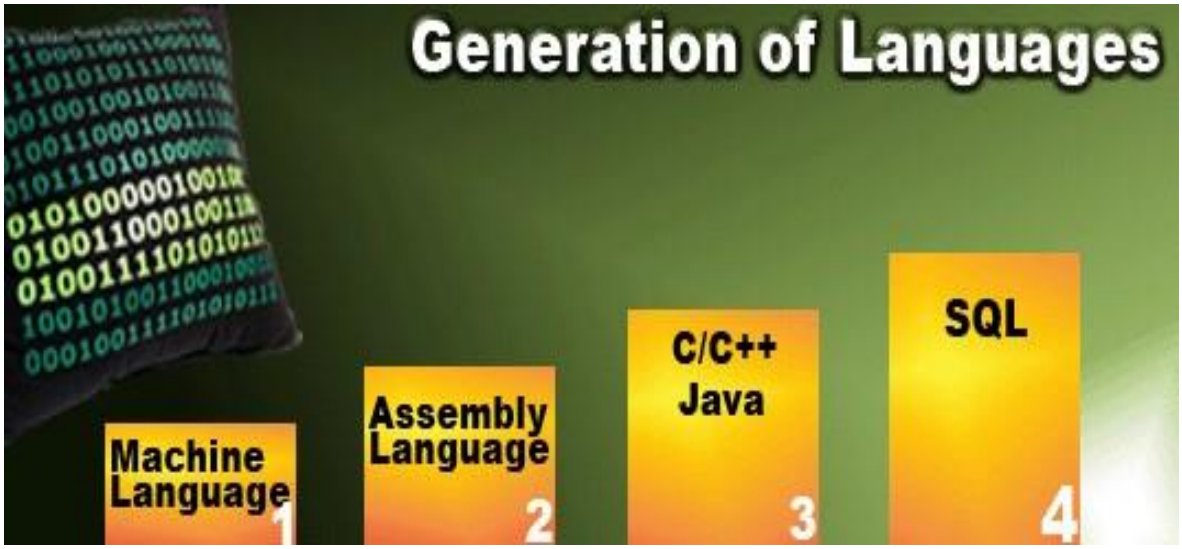
Generations of Programming Languages



Generation of Languages

The First Generation

The **first-generation programming language** is machine-level programming language. It consists of 1s and 0s, the **machine code** i.e., the only language that the computer understands. It did not require a translator because it could be understood by the computer. Machine code and assembly language are called **low-level languages** because to use them, the programmer has to have a knowledge of the architecture of the computer.

In the first electronic computers, instructions were entered through the front panel switches of the computer system.

Advantages

- Machine language makes fast and efficient use of the computer.
- It requires no translator to translate the code. It is directly understood by the computer.

Disadvantages

- Machine language is somewhat difficult to learn.
- It is very easy to make mistakes.
- Very difficult to edit if errors occur.
- A program is not portable to other processors since they have a different language.

The Second Generation

A second-generation programming language is a term refers to the **assembly language**. Unlike first-generation programming languages, the code can be read and written easily by a human, but it must be converted into a machine-readable form to run on a computer. The conversion process is simply a mapping of the assembly language code into binary machine code (the first-generation language).

As with machine code, assembly language is specific to and dependent on a particular processor family and environment. Since it is the native language of a processor it has significant speed advantages, but it requires more programming effort than high-level languages, and is difficult to use effectively for large applications.

```
; Example of IBM PC assembly language
; Accepts a number in register AX;
; subtracts 32 if it is in the range 97-122;
; otherwise leaves it unchanged.

SUB32  PROC          ; procedure begins here
       CMP  AX,97    ; compare AX to 97
       JL   DONE     ; if less, jump to DONE
       CMP  AX,122   ; compare AX to 122
       JG   DONE     ; if greater, jump to DONE
       SUB  AX,32    ; subtract 32 from AX
DONE:  RET           ; return to main program
SUB32  ENDP          ; procedure ends here
```

A Program in Assembly Language

Advantages

- Assembly language is easier to understand and use as compared to machine language.
- It is easy to locate and correct errors.
- It is easily modified.

Disadvantages

- Like machine language, it is also machine dependent/specific.
- Not well suited for large programs.

The Third Generation

A third-generation language (3GL) is a programming language designed to be easier for a human to understand. The languages, of the like of Java and Python are called **high-level languages**. Most modern languages are third generation. To use these languages, one does not need to know the architecture of the computer. For this reason they are called high-level. These languages are also called **problem-oriented**, while low-level languages are called **machine-oriented**.

Advantages

- High-level languages are user-friendly.
- They are similar to English and use English vocabulary and well-known symbols.
- They are easier to learn.
- They are easier to maintain.
- They are problem-oriented rather than 'machine'-based (i.e., programmer focus on problem rather than on architecture).
- A program written in a high-level language can be translated into many machine languages and can run on any computer for which there exists an appropriate translator.

Disadvantages

- A high-level language has to be translated into the machine language by a translator, which takes up time.
- The object code generated by a translator might be inefficient compared to an equivalent assembly language program.

The Fourth Generation

A fourth-generation programming language (abbreviated **4GL**) is a programming language designed with a specific purpose in mind, such as the development of commercial business software. Such languages are meant to be easier than HLLs. Many 4GLs are database languages e.g., SQL (Structured Query Language).

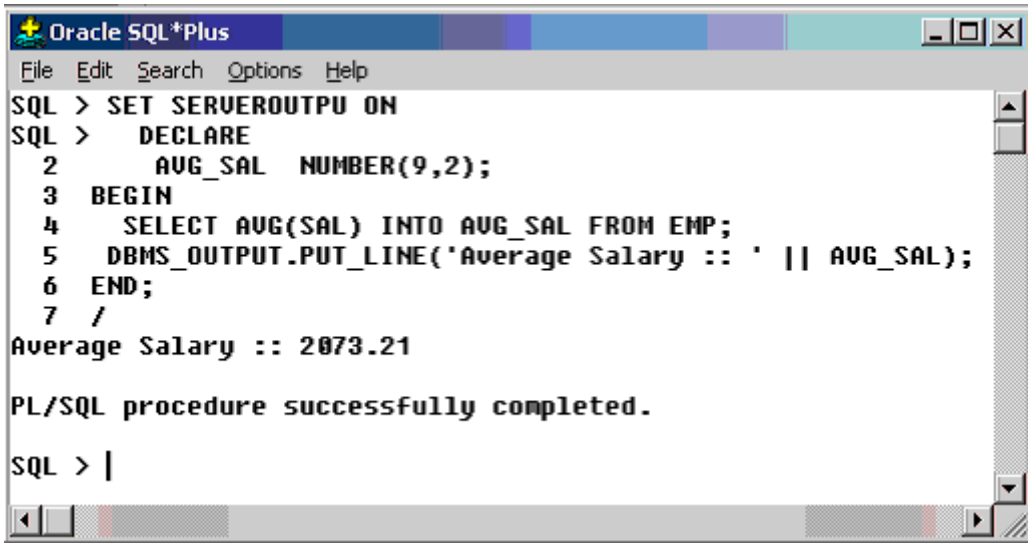
Advantages

- Programming productivity is increased. One line of 4GL code is equivalent to several lines of 3GL code.
- Program maintenance is easier.
- End user can often develop their own applications.

- Programs developed in 4GLs are more portable than those developed in other generation of languages.

Disadvantages

- The programs developed in the 4GLs are executed at a slower speed by the CPU.
- The languages are limited and are not as versatile as HLL.



An SQL program