

# Introduction to Programming Languages SLO #1.4.1

---

## **INTRODUCTION:**

A programming language is an artificial language designed to express computations that can be performed by a machine, particularly a computer. Programming languages can be used to create programs that control the behavior of a machine,

## **MACHINE-ORIENTED LANGUAGE:**

A programming language designed for use on a specific class of computers is called machine-oriented language.

## **PROBLEM-ORIENTED LANGUAGES:**

The term problem-oriented languages, if taken literally, are too general to be useful in the taxonomy of programming languages. In its most general meaning, one would have to include any programming language that helps solve problems. Thus, FORTRAN (q.v.) is a problem-oriented language when one solves scientific or numeric problems. COBOL (Common Business-Oriented Language--q.v.) is problem oriented, even in its title, for business problems. However, accepted usage in computer science literature has imposed a narrower context for problem-oriented languages than one that could encompass FORTRAN and COBOL. From this more restricted point of view, synonyms for problem-oriented are applications-oriented or special-purpose or specialized-application or domain-specific

## **HIGH LEVEL LANGUAGES:**

An advanced computer programming language that isn't limited by the computer or for one specific job and is more easily understood today, there are dozens of high-level languages; some commonly used high-level languages are BASIC, C, FORTAN and Pascal.

## **GENERATIONS OF PROGRAMMING LANGUAGES:**

Programming languages are use to write application programs which are used by end users. The programming languages are generally used only by professional programmers to write programs. The development of programming languages has improved considerably with the ease and ability of programmers to write powerful applications programs that can solve any task in the world today.

Each computer programming language has its own distinctive grammars and syntax and its own manner of expressing ideas. In principle most computational task could be accomplish by any of

## Introduction to Programming Languages SLO #1.4.1

---

the languages but the programs would look very different moreover, writing a program for a particular task could be easier with some languages than the others. The various generations of computer programming languages are discussed below.

### **1<sup>ST</sup> GENERATION LANGUAGES:**

The first generation computer language was machine language, all the machine used machine code which consisted of 0s and 1s. Machine language is highly efficient and allows direct control of each operation; however programmers had to write computer programs using 0 and 1. Some of the drawbacks of the first generations languages were.

- Programs were difficult to write and debug
- Programming process was tedious
- Programming was time confusing
- Programs were error prone

### **2<sup>ND</sup> GENERATION LANGUAGES:**

These were developed in the early 1950s with the ability to use acronyms to speed programming and coding of programs. They were known generational languages were called **assembly languages**. They had the capability to performs operation such like add, sum. Like machine languages, assembly languages were designed for specific machine and microprocessor; this implies that the program cannot be move from one computer architecture without writing the code which means learning another language where you are to transfer the programs.

### **3<sup>RD</sup> GENERATION LANGUAGES:**

These were introduced between 1956 and 1963 which saw a major breakthrough in computing history with the development of high level computer languages popularly known as 3<sup>rd</sup>(3GLS). Example of the 3<sup>rd</sup> generation languages includes the following

1. FORTRAN – Formula Translation
2. COBOL – Common Business Oriented Languages