

Computer Languages

Gagan

rozyph2017@gmail.com

www.rozyph.com

YouTube : https://youtu.be/fgPjzxH_jyA

Book : Fundamentals of Computer Vol. I by Gagan Deep

INTRODUCTION

- A natural language is a medium of communication between human being. The natural languages such as Hindi, Punjabi, English, etc. is used to communicate with each other our ideas and emotions. Similarly, a computer language is a means of communication used to communicate between people and the computer.

- Means, with the help of a computer language, a programmer tells a computer what he wants it to do.
- All natural languages use a standard set of symbols for the purpose of communication.
- These symbols are understood by everyone using that language. We normally call this set of symbols the vocabulary of that particular language.

- For example, the words we use in English are the symbols of English language that make up its vocabulary. Each word has definite meaning which can be looked up in a dictionary.
- In a similar manner, all computer languages have a vocabulary of their own.
- Each symbol of the vocabulary has definite unambiguous meaning which can be looked up in the manual meant for that language.

WHY PROGRAMMING LANGUAGES?

- Computer programming languages are developed with the priority objective of facilitating a large number of people to use computers without the need to know in detail the internal structure of the computer.
- Languages are matched to the type of operations to be performed in algorithms for various applications.
- Languages are also designed to be machine-independent.
- In other words, the structure of a programming language would not depend upon the internal structure of a specified computer.
- Ideally, one should be able to execute a program on any computer regardless of who manufactured it or what model it is.

GENERATIONS OF PL

- Programming languages have improved throughout the years, just as computer hardware has improved. They have progressed from machine-oriented languages that use strings of binary 1s and 0s to problem-oriented languages that use common mathematical and/or English terms.
- **However, the development of programming language can be distinctly divided into four generation :**
 - 1st Generation Language : Machine Languages (1940-50)
 - 2nd Generation Language : Assembly Language (1950-58)
 - 3rd Generation Language : High Level Languages (1958-85)
 - 4th Generation Language : 4-GLs (1985 onwards)

MACHINE LANGUAGE (1ST GENERATION)

- A computer system is programmed to understand many computer languages but only language is understood by computer without using any type of translation, is called the machine language or the machine code of the computer.
- So, we can say that the Machine Language is the language directly understood by a computer.
- In other words, the binary language (the language of 0's and 1's) is the machine language is normally written as strings of binary 1s and 0s.
- The circuitry of a computer is also designed to recognize the machine language immediately.

- Any information or instruction in this language is to be represented in terms of 0s and 1s, the symbol 0 standing for the absence of an electric pulse and 1 for the presence of an electric pulse.
- As a computer is able to recognise the presence or absence of an electric pulse, it is able to understand the machine language.
- For example, a sequence of 0s and 1s such as 01110001 has a specific meaning for a computer, although it may appear as an ordinary binary number to us.
- So, the writing of programs in machine language is very complicated and difficult task. Only experts can use this language.

- In Machine language, instruction format is divided into two parts.
- The first part is operation code which tell the computer to perform such a function. This is also called 'opcode'.
- Second part of the instruction is the operand which tell the computer about the location of data or another instruction on which the operation will be performed. This is also Called 'operand'

OPCode

- Operation Code

OPERAnd

- Operation Address

Machine language consists of strings of binary numbers and is the only one which directly understands by the CPU. For example, a program instruction is as under which consists of binary numbers (1s and 0s) :

1011001111101001110110

- The program to add two numbers in memory and print the result might look like the following :

0010000000000001100111001

0011000000000010000100001

0110000000000011100101110

10100011111011100101110

00000000000000000000000000

Advantage :


- Programs written in machine language can be executed very fast by the computer. This is mainly because machine instructions are directly understood by the CPU and no translation of the program is required.
- Small in size
- All programs a combination of 1 and 0

Disadvantages :

- Machine dependent
- **Difficult to program**
- Error prone
- **Difficult to modify**

ASSEMBLY LANGUAGE (2ND GENERATION)

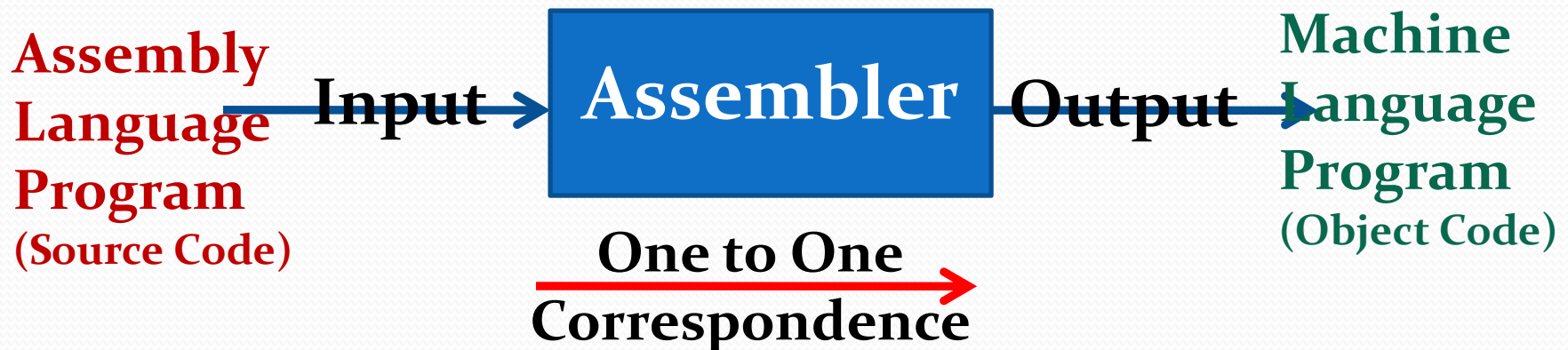
- Some of the difficulties arising in use of machine language could be overcome by use of Assembly Language for writing of programs.
- Step in improving the program preparation process was to substitute letter symbols mnemonics for numeric operation codes of machine language.

- 
- A mnemonic is any kind of mental trick we use to help us remember.
 - Mnemonics comes in various shapes and sizes.
 - For example, a computer is designed to interpret machine code of 1111 (binary) or 15 (decimal) as operation 'subtract' but it is easier for a human being to remember it as SUB.

- Then we can write program for the computer using symbols instead of numbers, and have the computer do its own translating.
- This makes it easier for the programmer, because he can use letters, symbols, and mnemonic instead of numbers for writing his programs.
- Example of a program adding two numbers and printing the result :

```
CLA    A
ADD    B
STA    C
TYP    C
HLT
```

- Which could mean take “A” , add “B”, store the result in “C” and type “C” and halt. The computer translate each line of this program into corresponding machine language code.
- Translator who translate Assembly Language Program to Machine Language is known as Assembler.



Advantages of Assembly Language (Over Machine Language) :

- Easier to understand and use
- Easy to modify a program
- Easy to error correction
- Easy to relocate the address
- Easy about addresses
- Efficiency of machine language

Limitation of Assembly Language

- Machine Dependent
- Knowledge of hardware required
- Machine level coding

HIGH-LEVEL LANGUAGE (3RD GENERATION)

- Writing of programs in machine language or assembly language requires a deep knowledge of the internal structure of the computer.
- While writing programs in any of these languages, a programmer has to remember all the operation codes (numeric or mnemonic) of the computer and know in detail what each code does and how it affect the various registers of the computer.
- But, writing a good program, a program should mainly concentrate on the logic of the problem rather than be concerned with the details of the internal structure of the computer. In order to facilitate the programmers to use computers without the need to know in detail the internal structure of the computer, high-level languages were developed.

HIGH-LEVEL LANGUAGE (3RD GENERATION)

- High-level languages, instead of being machine based, are oriented more towards the problem to be solved. These languages enable the programmer to write instructions using English words and familiar mathematical symbols.
- It becomes easier for him to concentrate on the logic of his problem rather than getting involved in programming details. For example, let us consider the problem of adding two numbers (A and B) and store the sum in SUM. Using a high-level language, say FORTRAN or instance, to instruct the computer to do this job, only one instruction need be written :

$SUM = A+B$

- The instruction is obviously very easy to understand and write because it resembles the familiar algebraic notation for adding to number : $a=b+c$.

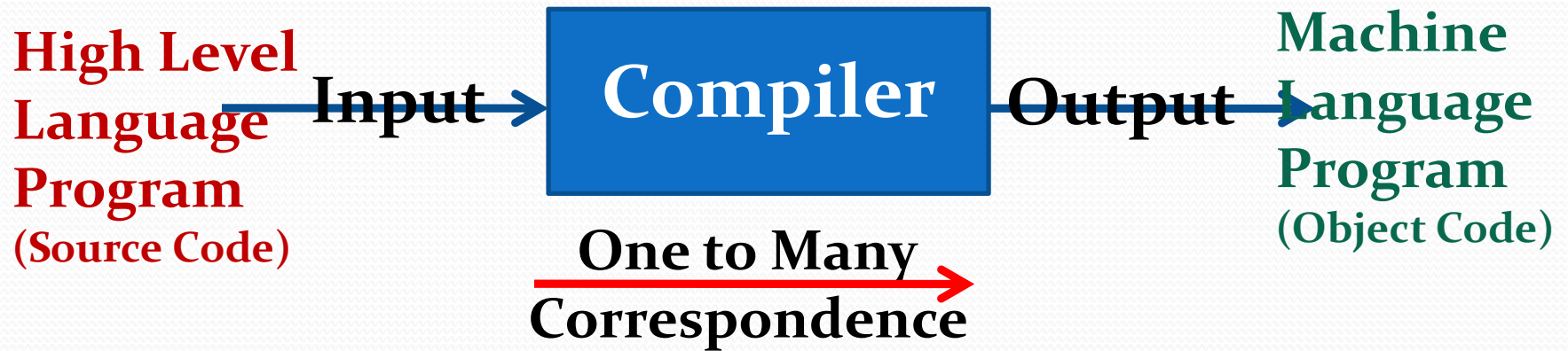
HIGH-LEVEL LANGUAGE (3RD GENERATION)

- High-level languages are basically symbolic languages that use English words and/or mathematical symbols rather than mnemonic codes.
- Every instruction which the programmer writes in high-level language is translated into many machine language instructions.
- This is one to many machine translation and not one-to-one as in the case of assembly language.
- The Translator who translate these HLL instructions to MLL instructions is known as Compiler.

HIGH-LEVEL LANGUAGE (3RD GENERATION)

- High-level languages are basically symbolic languages that use English words and/or mathematical symbols rather than mnemonic codes.
- Every instruction which the programmer writes in high-level language is translated into many machine language instructions.
- This is one to many machine translation and not one-to-one as in the case of assembly language.
- The Translator who translate these HLL instructions to MLL instructions is known as Compiler.

COMPILER



HIGH-LEVEL LANGUAGE (3RD GENERATION)

High Level languages may be divided into three categories :

- **Procedure Oriented Language**
- **Problem Oriented Language**
- **Interactive Programming Language**

High Level Languages are sometimes classified as

- **General Purpose Language**
- **Special Purpose Language**

Characteristics of a Good PL

- Simplicity
- Efficiency
- Environment suitable
- Compactness
- Locality
- Naturalness
- Structured
- Extensibility

FOURTH GENERATION LANGUAGES (4GLs)

- The Fourth Generation languages also called 4GLs are the highly users friendly languages, which means these languages are easy to understand and to write programs with, for the users.
- The advancement in computer hardware made the computers simpler machines and it became easier to handle these machines.
- Advancement of computer-hardware necessitated the need of development of better computer languages so that advanced feature of computers could, be fully utilised.
- The efforts to develop better computer language resulted in development of Fourth Generation languages.

FOURTH GENERATION LANGUAGES (4GLs)

- The fourth generation languages are non-procedural languages and hence are highly users friendly.
- In fact the non-procedural nature of these languages is the reason of their popularity.
- For easy use, most of the 4GLs are menu driven languages and provide interactive menus.
- The major advantages of these languages are that even a novice in the field of computers can solve fairly complicated problems.

Major characteristics of 4 GLs

The major characteristics of fourth generation languages are :

- Precise-nature.
- Non-procedural..
- Structure-independent.

Components of a 4GL

- A data dictionary..
- A query language that interrogates a database or data bank.
- A report generator, which automatically executes a program to produce a printed report.
- A data base management system.
- Statistical analysis tools
- Financial analysis tools Decision support tools.
- A Graphic manipulator
- A screen generator.

Components of a 4GL

- A menu.
- A dialogue generator.
- A word processor.
- A spreadsheet.
- Communication interfaces.

All these components available as separate packages are integrated in a typical 4 GL.

Some popular 4 GLs

- Focus
- Mapper
- Ramis II
- Nomad 2
- Use. It
- Linc
- Database query language e.g. SQL
- Metafont
- PostScript
- Mathematica



Thanks!

For any question ...

...Leave Comment!