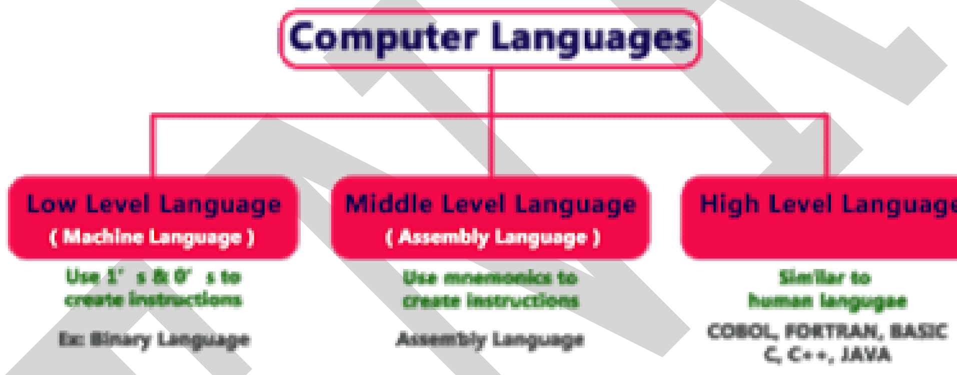
# *Computer languages*

*A Computer language / Programming language is a vocabulary and set of grammatical rules for instructing a computer or computing device to perform specific tasks. It serves the purpose of commanding the computer.*

*They are broadly classified into two types:*

1. *Low level language*
2. *High level language*



## *Low level language :*

*It is a language that is understandable by the computer. Here each programme statements is converted to single machine language statement. There are two types of low level languages.*

### ***Machine level language:***

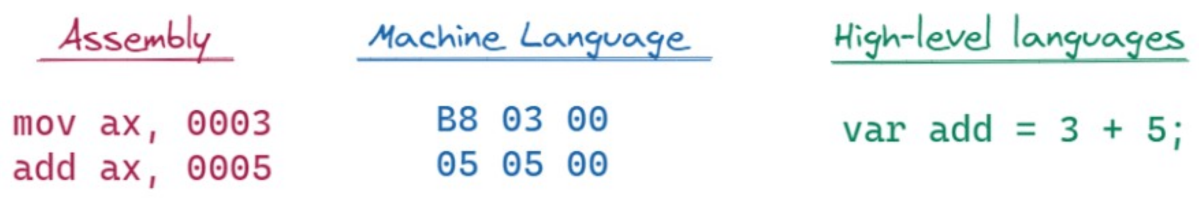
*In this language the data and instruction are represented in the form of strings of 0s and 1s. Every instruction is represented in the form of binary bit pattern. For e.g., the instructions add is represented as "0110001". Similarly the instruction divide has bit pattern of "0110000".*

***Assembly language:***

*In this language, instead of using binary codes, short codes are used which are called "**MNEMONICS**". Each "MNEMONICS" has two parts, the first part is instruction and second part is data for, e.g. ADD A, B is a MNEMONICS, where ADD is instruction part and A & B are data parts. Due to use of short codes, programmers do not require remembering bit pattern which is very tedious task.*

***High level language: ( Third Generation Language ):***

*On the other hand, offer English like keywords, constructs for sequence, selection (decision) and iteration (looping) and use of variables and constants. Thus it is very easy to program with such languages compared to low level languages. The programs written in HLLs are converted into machine language using compiler or interpreter as a computer can work with machine language only*


Assembly

```
mov ax, 0003
add ax, 0005
```

Machine Language

```
B8 03 00
05 05 00
```

High-level languages

```
var add = 3 + 5;
```

***Middle Level Language :***

***A programming language should serve two related purposes , it** should proved a vehicle for the programmer to specify actions to be executed ("**close to the machine)**and it should provide a set of concepts for the programmer to use when thinking about what can be done(**close to the problem to be solved)**.*

*The low level language serve only the first aspect i.e., they are close to the machine and the high level languages serve only the second aspect i.e., they are close to the programmer.*

*However, the languages 'C' and 'C++' serve both the aspects, hence can be called as 'middle level language'.*

## *Language Translators :*

*Language translator does the work of conversation of high level language into machine understandable form.There are three types of language translator:*

***Assembler :***

*It converts assembly language statements into machine understandable form, i.e. of Os and 1s.*

***Interpreter :***

*It converts high level language statements into binary file Os and 1 s. Conversion is done letter by letter, word by word and line by line. When every letter, word and line are correctly written, the interpreter will convent that line into file of O and 1. The conversion process is slow in comparison to compiler.*

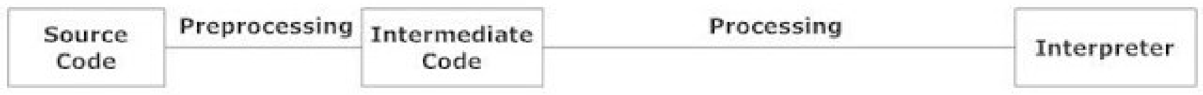


Figure: Interpreter

***Compiler:***

*It converts high level language statements into machine understandable form and is done at single step unlike interpreter. A compiler checks entire user written programmes and if error free, produces the complete program in machine language. If the program use some library functions, the library functions are linked to other object program to generate executable program*

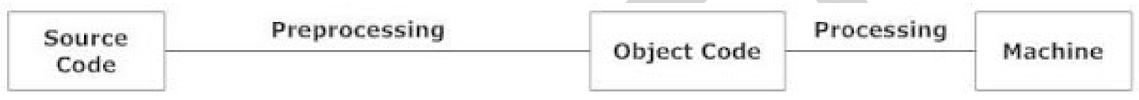
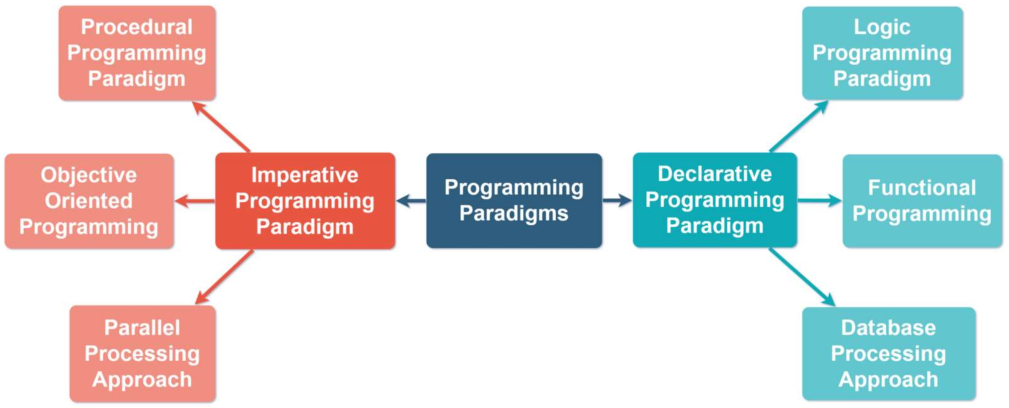


Figure: Compiler

## *Introduction to programming paradigms :*

***The term programming paradigm refers to a style of programming****. It does not refer to a specific language, but rather it refers to the way you program.*

*There are lots of programming languages that are well-known but all of them need to follow some strategy when they are implemented. And that strategy is a paradigm.*


Procedural Programming Paradigm
Objective Oriented Programming
Parallel Processing Approach
Imperative Programming Paradigm
Programming Paradigms
Declarative Programming Paradigm
Logic Programming Paradigm
Functional Programming
Database Processing Approach

***1. Imperative programming approach:***

*Imperative languages are largely focused on interpreting how a program operates. They use a sequence of command steps to alter a program's state, so rather than describing what a program should do, imperative languages communicate* ***how to do it****.*

***Procedural programming approach:***

*In Procedural programming , the problem is bifurcated in to number of smalll promblems known as procedures ( Also alternatively referred as functions or methods)*

***Object-Oriented programming approach :***

*The Object Oriented Programming (OOP) is an approach to software application development where all application component of a program that has its own data and methods*

***Concurrent / Parallel programming approach***

*Concurrent languages support the simultaneous execution of threads and processes by means of structuring a program. Concurrent programming languages are most frequently employed to increase CPU-intensive program efficiency.*

***2. Declarative programming approach***

*Functioning similarly to imperative languages, declarative languages instruct a program on what to do, rather than telling it* ***how to do it.***

***Functional programming approach***

*Functional languages are a type of declarative language that builds the elemental structure of a program by treating computation as the evaluation of mathematical functions.*

***Logic programming approach :***

*Program statements express facts and rules about problems within a system of formal logic*

***Database/Data driven programming approach :***

*This programming methodology is based on data and its movement. Program statements are defined by data rather than hard-coding a series of steps*

***Some more programming paradigms to be known :***

***Scripting Languages***

*Scripting languages automate the execution, integration, and communication between other programming languages. Scripting languages are typically used in conjunction with other programming languages.*

***Multi-Paradigm Languages***

*Multi-paradigm programming languages provide a structured framework in which computer engineers can work with multiple programming languages simultaneously.*

***Dynamic Languages***

*Dynamic programming languages allow operations traditionally done at compile-time to be completed at run-time.*

***" The best way to learn a new programming language is by writing programs in it. "*** ***- Dennis Ritchie***

♡ Happy Coding ♡