**1883: The Journey starts from here…!!**

* **In the early days, Charles Babbage had made the device, but he was confused about how to give instructions to the machine, and then Ada Lovelace wrote the instructions for the analytical engine.**

**1949: Assembly Language**

**1952: Autocode**

* **COBOL and FORTRAN are the languages referred to as Autocode.**

**1957: FORTRAN**

**1958: ALGOL**

**ALGOL stands for ALGOrithmic Language.**

**1959: COBOL**

**1964: BASIC**

* **In 1991 Microsoft released Visual Basic, an updated version of Basic**

**1972: C**

**1978 : SQL**

**1983 : Objective-C , C++**

**1991 : Python**

**1995 : Java**

**2000 : C#**

**2009 : GO**

**2011 : Kotlin**

**2014 : Swift**

**Types of Computer Languages**

**Low level programming languages** : it is written in the form of 0 and 1. These instructions are directly given to computer. Two types of low level languages are machine level and assembly level languages.

* Machine level : written only using 0 and . where 1 is positive and 0 is negative
* Assemply level : It is advanced to machine level language. This language uses symbols. It is more understandable to humans than machine level. With the help of assembler later assembly level language gets converted into machine level language.

**Middle Level Languages :**  As name suggests , it refers to languages between high level and low level languages. It has sufficient abstraction. More used in system application programs. E.g. C , C++ , Java

**High level programming languages** : These are advanced to assembly languages amd machine level language. These are user friendly languages which can be easily understood by humans. These languages and particular syntax, and are independent of particular type of computer. These language abstract machine level language( Low level language). IT uses English words. E.g. Python, Java , Javascript , PHP , swift , Kotlin, go , R

Difference

|  |  |  |
| --- | --- | --- |
| Sr | High Level | Low Level |
| 1 | Easily understanble to humans as it consist of English words | Very hard to understantd, use binary 0 and 1 |
| 2 | Takes long time to execute, because of preprocessor , compiler, assembler, | Executes very fast. As there is no abstraction. |
| 3 | Simple to maintain program | Complex to maintain program |
| 4 | Debugging is easy | Debugging is very hard |
| 5 | Used is today’s technology worldwide | Are not used in current era of technology |

**Classification of languages by characteristics.**

1. **Procedural Languages :** Also known as procedural oriented languages (POP). Programs are written in stepwise format.

E.g : C , Pascal , Fortran , Cobol

1. **Functional Programing Languages :** programs are written based on mathematical functions. Helpful in performing computations.

**E.**g Scala, F#,

1. **Object-oriented Programming** **Languages (OOP)** : program as set of objects are referred to as object oriented languages. Programs contains objects which contains data or code.

**E**.g Python , Ruby , C++, java

1. **Scripting Languages (interpreted languages):** commands or instruction are written for interpreted in runtime environment. There are no compiler used in them. Interpreter is used at runtime.

E.**g :** Python , PHP

1. **Front End Languages :** used for frontend web development. To create what user see in browser.

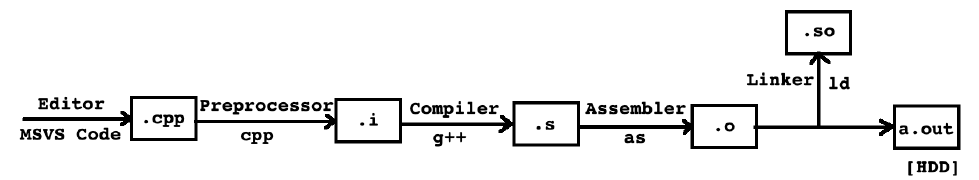
**E.g** html , css, javascript , React.

1. **Backend Languages :**  used for developing server side requests made by front end.

E.g PHP , Javascript , Python, Java,

1. **Compiled Languages :**  use compiler to convert programming language code into machine languages.

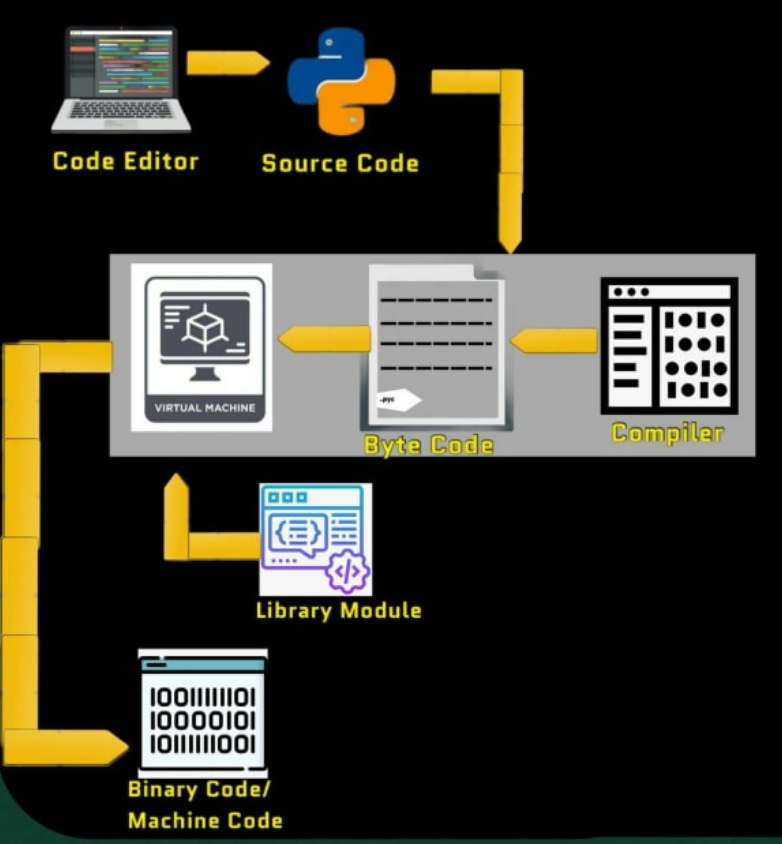
**E,g**  C, C++



1. **Interpreted Languages :** These languages skip the compiler step and are interpreted by an interpreter at execution/runtime

**E.g**  Python ,

**How python works**



**Understanding Memory Management: The Key to Efficient Programming in Any Language**

Memory management is a critical aspect of programming languages that involves allocating and deallocating memory during program execution. Here are some key concepts related to memory management:

1. Memory allocation: When a program needs to store data, it requests memory from the operating system. The memory can be allocated statically, at compile time, or dynamically, at runtime.
2. Memory deallocation: Once the program is finished using the memory, it must be returned to the operating system. Failure to do so can lead to memory leaks, which can cause the program to crash or slow down over time.
3. Garbage collection: In some programming languages, such as Java and Python, memory management is automated through a process called garbage collection. This involves periodically scanning the program’s memory to identify and free up memory that is no longer in use.
4. Pointers: In low-level programming languages such as C and C++, memory is managed using pointers. These are variables that store memory addresses, allowing the program to manipulate memory directly. However, incorrect use of pointers can lead to memory leaks and other errors.
5. Memory safety: Memory safety refers to the prevention of memory-related errors such as buffer overflows, null pointer dereferences, and use-after-free errors. Programming languages that prioritize memory safety, such as Rust and Swift, use techniques such as ownership and borrow checking to ensure that memory is used correctly.