**Name: G A M S Gamlath**

**Index No: 29800**

**Tutorial 1**

1. A programming language is a set of rules and syntax used to write instructions for a computer to execute. It acts as a bridge between humans and computers, allowing programmers to communicate their intentions and solve complex problems effectively.

2.

a. **Source Code vs. Machine Code**

Source Code: It refers to the human-readable code written by programmers in a programming language like C, Java, or Python. Source code is written using high-level constructs and abstractions that are closer to natural language, making it easier for programmers to write and understand.

Machine Code: It consists of binary instructions directly understood and executed by the computer's hardware. Machine code is specific to the computer architecture and is not easily readable or writable by humans. It represents the lowest level of programming language.

b. **High-Level Language vs. Low-Level Language**

High-Level Language: These languages provide a high level of abstraction from the underlying hardware. They are designed to be closer to human language, allowing programmers to write code that is more readable, portable, and easier to understand. Examples include Python, Java, and Ruby.

Low-Level Language: These languages are closer to the hardware and provide minimal abstraction. They are more closely related to machine code and are less readable and portable. Low-level languages include assembly language and machine code.

Compiler: It is a software tool that translates the entire source code into machine code before execution. The compiler performs various stages like lexical analysis, syntax analysis, optimization, and code generation. Examples of compiled languages include C, C++, and Go.

Interpreter: It executes the source code line by line, translating and executing each instruction in real-time. Interpreters are typically slower than compilers but provide more flexibility and dynamic features. Python, JavaScript, and Ruby are examples of interpreted languages.

d. **Structured Language vs. Object-Oriented Language**

Structured Language: It is a programming language that emphasizes the use of structured programming techniques, such as loops, conditionals, and modular programming. Examples of structured languages include C, Pascal, and Fortran.

Object-Oriented Language: It is a programming paradigm that organizes code around objects, which encapsulate data and behavior. Object-oriented languages provide features like classes, inheritance, and polymorphism. Examples include Java, C++, and Python.

e. **C vs. C++**

C: C is a procedural programming language known for its simplicity, efficiency, and low-level programming capabilities. It provides a minimalistic set of features and allows direct manipulation of memory. It is commonly used in systems programming and embedded systems development.

C++: C++ is an extension of the C programming language with added features, primarily object-oriented programming. It introduces classes, inheritance, and polymorphism while maintaining backward compatibility with C. C++ is widely used for general-purpose software development, game development, and high-performance applications.

f. **C++ vs. Java**

C++: C++ is a statically typed, compiled language that offers high performance, direct hardware access, and low-level control. It is popular for system-level programming, game development, and performance-critical applications.

Java: Java is a statically typed, platform-independent language that runs on the Java Virtual Machine (JVM). It emphasizes portability, automatic memory management, and a vast ecosystem of libraries and frameworks. Java is commonly used for enterprise software, web development, and Android app development.

g. **Syntax Error vs. Logical Error**

Syntax Error: It occurs when code violates the rules of the programming language's syntax. These errors are detected by the compiler or interpreter during the compilation or execution phase. Syntax errors prevent the code from running and must be fixed before the program can proceed. - Logical Error: It occurs when the code does not produce the intended results due to flaws in the logic or algorithm. Logical errors are more challenging to detect as they do not cause the program to halt or produce error messages. Debugging and careful analysis of the code are required to identify and fix logical errors