DAY-1

1. WHAT IS PROGRAMMING?

A programming language is a set of instructions and syntax used to create software programs. A programming language is a formal language that specifies a set of instructions for a computer to perform specific tasks. It's used to write software programs and applications, and to control and manipulate computer systems. There are many different programming languages, each with its own syntax, structure, and set of commands. Some of the most commonly used programming languages include Java, Python, C++, JavaScript, and C#. The choice of programming language depends on the specific requirements of a project, including the platform being used, the intended audience, and the desired outcome. Programming languages continue to evolve and change over time, with new languages being developed and older ones being updated to meet changing needs.

2.Types of programming languages (compiled vs. interpreted) with examples?

S.NO.	COMPILED LANGUAGE	INTERPRETED LANGUAGE
1	A compiled language is a programming language whose implementations are typically compilers and not interpreters.	An interpreted language is a programming language whose implementations execute instructions directly and freely, without previously compiling a program into machine-language instructions.
2	In this language, once the program is compiled it is expressed in the instructions of the target machine.	While in this language, the instructions are not directly executed by the target machine.
3	There are at least two steps to get from source code to execution.	There is only one step to get from source code to execution.

4	In this language, compiled programs run faster than interpreted programs.	While in this language, interpreted programs can be modified while the program is running.
5	In this language, compilation errors prevent the code from compiling.	In this languages, all the debugging occurs at run-time.
6	The code of compiled language can be executed directly by the computer's CPU.	A program written in an interpreted language is not compiled, it is interpreted.
7	This language delivers better performance.	This language example delivers relatively slower performance.
8	Example of compiled language – C, C++, C#, CLEO, COBOL, etc.	Example of Interpreted language – JavaScript, Perl, Python, BASIC, etc.

3**. Basic syntax of all languages (including the C, C++, Java, Python) Basic code examples of all language(C,C++, Java, Python)?**

C Language:

- **Preprocessor Directives**:

#include <stdio.h> : This line tells the compiler to include the Standard Input Output library before actual compilation starts.

- **Main Function:**

int main() { ... }: This is the main function where the execution of the program begins. Every C program must have a main function.

- **Variable Declaration:**

int number; : Here, we declare an integer variable named number.

- **Input/Output Operations:**

printf("Enter a number: "); : This function prints the message to the console.

scanf("%d", &number); : This function reads an integer input from the user and stores it in the variable number.

- **Return Statement:**

return 0; : This statement ends the main function and returns 0 to the calling process, indicating that the program executed successfully.

Code Example:

```
#include<stdio.h>

Int main(){

Int n;

Scanf(“enter n value:%d”,&n);

Printf(“%d”,n);

}
```

Output: enter n value:10

10

C++ Language:

1.Preprocessor Directives: These are instructions to the compiler to include libraries or perform specific actions before the actual compilation starts. For example, #include <iostream> includes the standard input-output stream library.

2.Namespace Declaration: The using namespace std; statement allows you to use names from the standard library without needing to prefix them with std::.

3.Function Prototypes: These are optional declarations of functions that will be defined later in the code. They inform the compiler about the function's name, return type, and parameters.

4.Main Function: This is the entry point of every C++ program. The int main() function is where the execution starts.

5.Variable Declarations: Variables are declared and initialized here. This is where you define the data that your program will use.

6.Executable Statements: These are the actual instructions that the program will execute, such as calculations, input/output operations, and function calls.

7.Return Statement: The return 0; statement indicates that the program has ended successfully. The main function should return an integer value.

8.Function Definitions: If you have declared any functions earlier, you define them here. This is where you write the actual code for the functions.

Exmaple code:

```
#include <iostream>

using namespace std;

int main() {

cout << "Hello, World!" << endl;

return 0;

}
```

Java Language:

Basic Structure of Java :

- **Comments:**

comments-line comments start with //. Multi-line comments are enclosed between /* and */.

- **Class Declaration:**

class Main { ... } Every Java program must have at least one class definition. The class name should match the filename (e.g., Main.java).

- **Main Method:**
  public static void main (String [] args) { ... } This is the entry point of any Java application. The Virtual Machine (JVM) looks for this method to start execution.

- **Statements:**

Inside the main method, you can write statements like System.out.println("Hello, World!"); to perform actions. This particular statement prints text to the console.

Example code:

```
Import java.util.*;

public class Main {

// The main method - the entry point of the program
public static void main(String[] args) {
    // This line prints "Hello, World!" to the console
    System.out.println("Hello, World!");
}
```

Output: Hello, World!

Python Language:

Basic Syntax:

• No Semicolons: Statements do not require a semicolon at the end.

Main function: pyhton doesn't require a main() function, it's a good practice

• Indentation: Use indentation to define blocks of code.

• Comments: Use # for single-line comments.

Example code:

```
n=int(input("enter n value"))

Print(n)

Print("Hello python")
```

Output: enter n value:20

20

Hello pyhton