Object Oriented Programming Language

C++

C-OOP

Programming

Features of C-language

- C is a high-level.
- General-purpose programming language.
- Developed by Dennis Ritchie at Bell Labs in the early 1970s.
- It is often used to make things like operating systems and programs that need to run very quickly.
- With C, programmers can control how the computer uses memory, which helps them make their programs work better and faster.
- C is a foundational language, heavily used in system programming, operating systems, making the program better by using faster methods, removing repeated parts, or using less memory and power.

- High-Level: Easy for people to understand and write.
- --
- General-Purpose: Can be used to make all kinds of programs.
- --
- Procedural: Follows steps or instructions to do a task.
- --
- Efficient: Works fast and doesn’t waste memory or power.
- --
- Flexible: Can be used in many different ways.
- --
- Portability: Can work on different computers without big changes.

System Programming:

- C is widely used to write operating systems (like Linux), device drivers, and other low-level software.

Embedded Systems:

- C is a common choice for programming devices like microcontrollers and embedded systems.

Game Development:

- Many popular video games are written in C or C++.

Compilers and Interpreters:

- C is used to write the code for compilers and interpreters of other programming languages.

Applications:

- C can also be used to create a wide range of applications, including web browsers, word processors, and other software tools.

- C++ is a multi-paradigm,(a programming language that supports multiple programming styles or paradigms, such as object-oriented, functional, imperative, or declarative.)
- General-purpose programming language created as an extension of the C language.
- It's known for its speed, efficiency, and low-level memory management, (In C++, you can create memory using new and free it using delete. This gives you more power, but also more responsibility — if you forget to free memory, it can cause problems.)
- making it suitable for applications requiring high performance, such as game development, operating systems, and embedded systems

- C++ is a mix of styles.
- It lets you write programs using steps (procedural) or by organizing things into objects (object-oriented).

Comparison Between C & C++

Aspect	C	C++
Developer	C was developed by Dennis Ritchie between the year 1969 and 1973 at AT&T Bell Labs.	C++ was developed by Bjarne Stroustrup in 1979.
OOPs Support	C does not support polymorphism, encapsulation, and inheritance which means that C does not support object-oriented programming.	C++ supports polymorphism, encap sulation, and inheritance because it is an object-oriented programming language.
Subset/Superset	C is (mostly) a subset of C++.	C++ is (mostly) a superset of C.

Keywords	Number of keywords in C: - C90: 32 - C99: 37 - C11: 44 - C23: 59	Number of keywords in C++: - C++98: 63 - C++11: 73 - C++17: 73 - C++20: 81
Programming Paradigm	For the development of code, C supports procedural programming.	C++ is known as hybrid language because C++ supports both procedural and object-oriented programming paradigms.
Encapsulation	Data and functions are separated in C because it is a procedural programming language.	Data and functions are encapsulated together in form of an object in C++.
Data Hiding	C does not support data hiding.	Data is hidden by the Encapsulation to ensure that data structures and operators are used as intended.

Focus of Language	C is a function driven language because C is a procedural programming language.	C++ is an object driven language because it is an object-oriented programming.
Overloading	Function and operator overloading is not supported in C.	Function and operator overloading is supported by C++.
Function Inside Structures	Functions in C are not defined inside structures.	Functions can be used inside a structure in C++.
Namespaces	Namespace features are not present inside the C.	Namespace is used by C++, which avoid name collisions.
Standard I/O	Standard IO header is stdio.h and uses scanf() and printf() functions are used for input/output in C.	Standard IO header is iostream.h and uses cin and cout are used for input/output in C++.
References	Reference variables are not supported by C.	Reference variables are supported by C++.

Exception Handling	**Direct support for exception handling is not supported by C.**	**Exception handling is supported by C++.**
Access Modifiers	C structures don't have access modifiers.	C ++ structures have access modifiers.
Type Checking	There is no strict type checking in C programming language.	Strict type checking in done in C++. So many programs that run well in C compiler will result in many warnings and errors under C++ compiler.
Type Punning with Unions	Type punning with unions is allows (C99 and later)	Type punning with unions is undefined behavior (except in very specific circumstances)
Named Initializers	Named initializers may appear out of order	Named initializers must match the data layout of the struct
Extension	File extension is ".c"	File extension is ".cpp" or ".c++" or ".cc" or ".cxx"
Generic Programming	Meta-programming using macros and _Generic()	Meta-programming using templates (macros are still supported but discouraged)

Key Differences

Feature	C	C++
Paradigm	Procedural	Object-oriented (supports procedural too)
Data Types	Built-in data types	Built-in and user-defined data types
Pointers & References	Pointers only	Pointers and references
Function Overloading	Not supported	Supported
Input/Output	scanf(), printf()	cin, cout
Memory Management	malloc(), calloc(), free( )	new, delete
Exception Handling	No built-in support	try, throw, catch blocks
File Extension	.c	.cpp

- The primary difference between C and C++ lies in their programming paradigms.

- C is a procedural programming language, focusing on functions and sequential execution, while C++ is an object-oriented programming language (OOP), emphasizing objects, classes, and their interactions.

C program

```c
#include <stdio.h>
int main() {
int num1 = 10, num2 = 20;
int sum = num1 + num2;
printf("Sum of %d and %d is %d\n", num1, num2, sum);
return 0;
}
```

C++ program

```cpp
#include <iostream>
int main() {
    int num1 = 10, num2 = 20;
    int sum = num1 + num2;
    std::cout << "Sum of " << num1 << " and " << num2 << " is " << sum << std::endl;
    return 0;
}
```

In the C++ example, std::cout is used for output, whereas C uses printf(). While this simple example doesn't showcase OOP features, it highlights the different approaches to I/O operations. C++ can incorporate classes, inheritance, and other OOP concepts, which are absent in C

Basic C++ Code Example – Student Info Program

```
#include <iostream>  // For input and output using namespace std;

// A function to display a welcome message
void greet() {
    cout << "Welcome to the Student Info Program!\n";
}
int main() {
    // Variables to store student data
    string name;
    int age;
    float marks;
    // Call the greeting function
    greet();
    // Taking input from the user
    cout << "Enter your name: ";
    cin >> name;
    cout << "Enter your age: ";
    cin >> age;
    cout << "Enter your marks (out of 100): ";
    cin >> marks;
    // Showing the output
    cout << "\n--- Student Details ---\n";
    cout << "Name: " << name << endl;
    cout << "Age: " << age << endl;
    cout << "Marks: " << marks << endl;

    // Simple condition
    if (marks >= 50) {
        cout << "Result: Pass\n";
    } else {
        cout << "Result: Fail\n";
    }

    // Loop example (counting from 1 to 5)
    cout << "\nCounting from 1 to 5:\n";
    for (int i = 1; i <= 5; i++) {
        cout << i << " ";
    }

    cout << "\nThank you!\n";
    return 0;
}
```

Key Concepts Covered:

- #include, main(), and using namespace stdInput/output (cin/cout)
- Variables (string, int, float)
- Functions (greet())
- Condition (if-else)
- Loop (for loop)