C++ Introduction

C++ is a general-purpose programming language that was developed as an enhancement of the C language to include object-oriented paradigm. It is an imperative and a **compiled** language.

1. C++ is a high-level, general-purpose programming language designed for system and application programming. It was developed by Bjarne Stroustrup at Bell Labs in 1983 as an extension of the C programming language. C++ is an object-oriented, multi-paradigm language that supports procedural, functional, and generic programming styles.
2. One of the key features of C++ is its ability to support low-level, system-level programming, making it suitable for developing operating systems, device drivers, and other system software. At the same time, C++ also provides a rich set of libraries and features for high-level application programming, making it a popular choice for developing desktop applications, video games, and other complex applications.
3. C++ has a large, active community of developers and users, and a wealth of resources and tools available for learning and using the language. Some of the key features of C++ include:
4. Object-Oriented Programming: C++ supports object-oriented programming, allowing developers to create classes and objects and to define methods and properties for these objects.
5. Templates: C++ templates allow developers to write generic code that can work with any data type, making it easier to write reusable and flexible code.
6. Standard Template Library (STL): The STL provides a wide range of containers and algorithms for working with data, making it easier to write efficient and effective code.
7. Exception Handling: C++ provides robust exception handling capabilities, making it easier to write code that can handle errors and unexpected situations.

Overall, C++ is a powerful and versatile programming language that is widely used for a range of applications and is well-suited for both low-level system programming and high-level application development.

Here are some simple C++ code examples to help you understand the language:

1.Hello World:

- C++

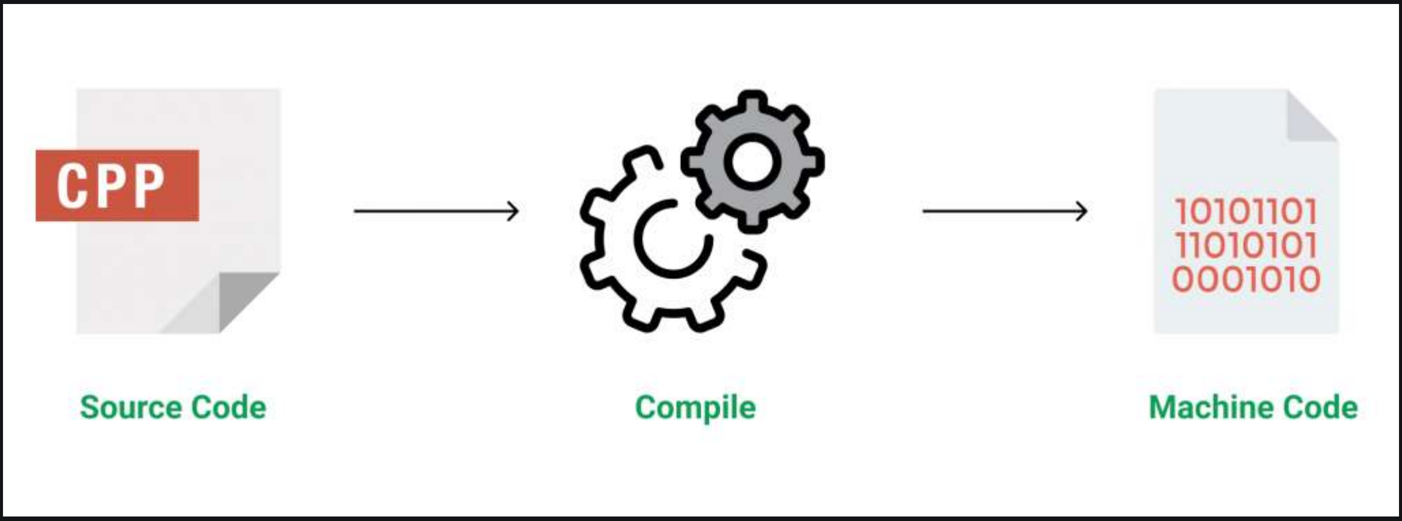
```cpp
#include <iostream>


int main() {

    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

Output

```
Hello, World!
```



C++ is a middle-level language rendering it the advantage of programming low-level (drivers, kernels) and even higher-level applications (games, GUI, desktop apps etc.). The basic syntax and code structure of both C and C++ are the same.
Some of the ***features & key-points*** to note about the programming language are as follows:

- **Simple**: It is a simple language in the sense that programs can be broken down into logical units and parts, has a rich library support and a variety of data-types.
- **Machine Independent but Platform Dependent**: A C++ executable is not platform-independent (compiled programs on Linux won't run on Windows), however they are machine independent.
- **Mid-level language**: It is a mid-level language as we can do both systems-programming (drivers, kernels, networking etc.) and build large-scale user applications (Media Players, Photoshop, Game Engines etc.)
- **Rich library support**: Has a rich library support (Both standard ~ built-in data structures, algorithms etc.) as well 3rd party libraries (e.g. Boost libraries) for fast and rapid development.
- **Speed of execution**: C++ programs excel in execution speed. Since, it is a compiled language, and also hugely procedural. Newer languages have extra in-built default features such as garbage-collection, dynamic typing etc. which slow the execution of the program overall. Since there is no additional processing overhead like this in C++, it is blazing fast.
- **Pointer and direct Memory-Access**: C++ provides pointer support which aids users to directly manipulate storage address. This helps in doing low-level

programming (where one might need to have explicit control on the storage of variables).

- **Object-Oriented**: One of the strongest points of the language which sets it apart from C. Object-Oriented support helps C++ to make maintainable and extensible programs. i.e. Large-scale applications can be built. Procedural code becomes difficult to maintain as code-size grows.
- **Compiled Language**: C++ is a compiled language, contributing to its speed.

Applications of C++:

C++ finds varied usage in applications such as:

- Operating Systems & Systems Programming. e.g. *Linux-based OS (Ubuntu etc.)*
- Browsers *(Chrome & Firefox)*
- Graphics & Game engines *(Photoshop, Blender, Unreal-Engine)*
- Database Engines *(MySQL, MongoDB, Redis etc.)*
- Cloud/Distributed Systems

Here are some key points to keep in mind while working with C++:

1. Object-Oriented Programming: C++ is an object-oriented programming language, which means that it allows you to define classes and objects to model real-world entities and their behavior.
2. Strong Type System: C++ has a strong type system, which means that variables have a specific type and that type must be respected in all operations performed on that variable.
3. Low-level Access: C++ provides low-level access to system resources, which makes it a suitable choice for system programming and writing efficient code.
4. Standard Template Library (STL): The STL provides a large set of pre-written algorithms and data structures that can be used to simplify your code and make it more efficient.
5. Cross-platform Compatibility: C++ can be compiled and run on multiple platforms, including Windows, MacOS, and Linux, making it a versatile language for developing cross-platform applications.
6. Performance: C++ is a compiled language, which means that code is transformed into machine code before it is executed. This can result in faster execution times and improved performance compared to interpreted languages like Python.
7. Memory Management: C++ requires manual memory management, which can lead to errors if not done correctly. However, this also provides more control over the program’s memory usage and can result in more efficient memory usage.
8. Syntax: C++ has a complex syntax that can be difficult to learn, especially for beginners. However, with practice and experience, it becomes easier to understand and work with.

These are some of the key points to keep in mind when working with C++. By understanding these concepts, you can make informed decisions and write effective code in this language.

Advantages of C++:

1. Performance: C++ is a compiled language, which means that its code is compiled into machine-readable code, making it one of the fastest programming languages.
2. Object-Oriented Programming: C++ supports object-oriented programming, which makes it easier to write and maintain large, complex applications.
3. Standard Template Library (STL): The STL provides a wide range of algorithms and data structures for working with data, making it easier to write efficient and effective code.
4. Machine Independent: C++ is not tied to any hardware or processor. If the compiler compiles the program in the system, it will be able to run no matter what the hardware is.
5. Large Community: C++ has a large, active community of developers and users, providing a wealth of resources and support for learning and using the language.

Disadvantages of C++:

1. Steep Learning Curve: C++ can be challenging to learn, especially for beginners, due to its complexity and the number of concepts that need to be understood.
2. Verbose Syntax: C++ has a verbose syntax, which can make code longer and more difficult to read and maintain.
3. Error-Prone: C++ provides low-level access to system resources, which can lead to subtle errors that are difficult to detect and fix.

Reference Books:

1. “The C++ Programming Language” by Bjarne Stroustrup
2. “Effective C++: 55 Specific Ways to Improve Your Programs and Designs” by Scott Meyers
3. “C++ Primer Plus” by Stephen Prata
4. “C++ For Dummies” by Stephen R. Davis
5. “Data Structures and Algorithm Analysis in C++” by Mark Allen Weiss
   Some interesting facts about C++:
   Here are some awesome facts about C++ that may interest you:
6. The name of C++ signifies the evolutionary nature of the changes from C. “++” is the C increment operator.
7. C++ is one of the predominant languages for the development of all kind of technical and commercial software.
8. C++ introduces Object-Oriented Programming, not present in C. Like other things, C++ supports the four primary features of OOP: encapsulation, polymorphism, abstraction, and inheritance.
9. C++ got the OOP features from Simula67 Programming language.
10. A function is a minimum requirement for a C++ program to run.(at least main() function)