Unit 1.Introduction to C++ (6 L, 18M)

1.1Basics of C++

C++ is a middle-level programming language developed by Bjarne Stroustrup starting in 1979 at Bell Labs.

Features of C++

- It provides a lot of features that are given below.
- Simple
- Abstract Data types
- Machine Independent or Portable
- Mid-level programming language
- Structured programming language
- Rich Library
- Memory Management
- Quicker Compilation
- Pointers
- Recursion
- Extensible
- Object-Oriented
- Compiler based
- Reusability
- National Standards
- Errors are easily detected
- Power and Flexibility
- Strongly typed language
- Redefine Existing Operators
- Modeling Real-World Problems
- Clarity
- ---

1) Simple

C++ is a simple language because it provides a structured approach (to break the problem into parts), a rich set of library functions, data types, etc.

2) Abstract Data types

In C++, complex data types called Abstract Data Types (ADT) can be created using classe

3) Portable

C++ is a portable language and programs made in it can be run on different machines.

4) Mid-level / Intermediate programming language

C++ includes both low-level programming and high-level language so it is known as a mid-level and intermediate programming language. It is used to develop system applications such as kernel, driver, etc.

5) Structured programming language

C++ is a structured programming language. In this we can divide the program into several parts using functions.

6) Rich Library

C++ provides a lot of inbuilt functions that make the development fast. Following are the libraries used in C++ programming are:

- <iostream>
- <cmath>
- <cstdlib>
- <fstream>

7) Memory Management

C++ provides very efficient management techniques. The various memory management operators help save the memory and improve the program's efficiency. These operators allocate and deallocate memory at run time. Some common memory management operators available C++ are new, delete etc.

8) Quicker Compilation

C++ programs tend to be compact and run quickly. Hence the compilation and execution time of the C++ language is fast.

9) Pointer

C++ provides the feature of pointers. We can use pointers for memory, structures, functions, array, etc. We can directly interact with the memory by using the pointers.

10) Recursion

In C++, we can call the function within the function. It provides code reusability for every function.

11) Extensible

C++ programs can easily be extended as it is very easy to add new features into the existing program.

12) Object-Oriented

In C++, object-oriented concepts like data hiding, encapsulation, and data abstraction can easily be implemented using keyword class, private, public, and protected access specifiers. Object-oriented makes development and maintenance easier.

13) Compiler based

C++ is a compiler-based programming language, which means no C++ program can be executed without compilation. C++ compiler is easily available, and it requires very little space for storage. First, we need to compile our program using a compiler, and then we can execute our program.

14) Reusability

With the use of inheritance of functions programs written in C++ can be reused in any other program of C++.

18) Strongly typed language

The list of arguments of every function call is typed checked during compilation. If there is a type mismatch between actual and formal arguments, implicit conversion is applied if possible. A compile-time occurs if an implicit conversion is not possible or if the number of arguments is incorrect.

19) Redefine Existing Operators

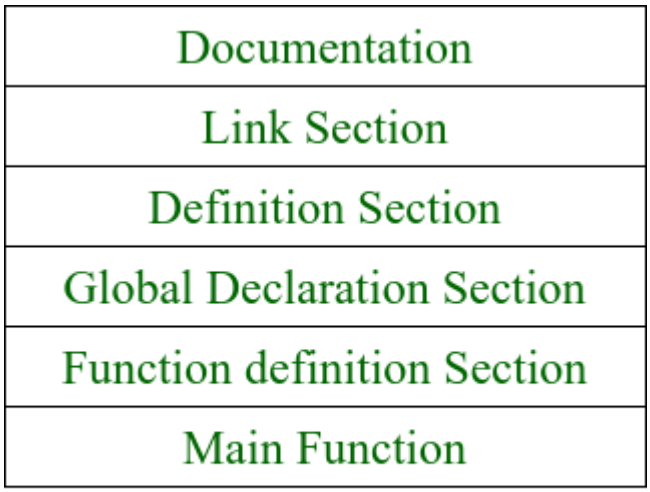
C++ allows the programmer to redefine the meaning of existing operators such as +, -. **For Example,** The "+" operator can be used for adding two numbers and concatenating two strings.

20) Modelling real-world problems

The programs written in C++ are well suited for real-world modeling problems as close as possible to the user perspective.

Structure of C++ Program

The C++ program is written using a specific template structure. The structure of the program written in C++ language is as follows:



Skeleton of C Program

Documentation Section:

- This section comes first and is used to document the logic of the program that the programmer going to code.
- It can be also used to write for purpose of the program.
- Whatever written in the documentation section is the comment and is not compiled by the compiler.
- Documentation Section is optional since the program can execute without them. Below is the snippet of the same:

```
/*  This is a C++ program to find the
    factorial of a number
```

The basic requirement for writing this
program is to have knowledge of loops

To find the factorial of number
iterate over range from number to one
*/

Linking Section:

The linking section contains two parts:

Header Files:

- Generally, a program includes various programming elements like built-in functions, classes, keywords, constants, operators, etc. that are already defined in the standard C++ library.
- In order to use such pre-defined elements in a program, an appropriate header must be included in the program.
- Standard headers are specified in a program through the preprocessor directive #include. In Figure, the iostream header is used. When the compiler processes the instruction #include<iostream>, it includes the contents of the stream in the program. This enables the programmer to use standard input, output, and error facilities that are provided only through the standard streams defined in <iostream>. These standard streams process data as a stream of characters, that is, data is read and displayed in a continuous flow. The standard streams defined in <iostream> are listed here.
- #include<iostream>

Namespaces:

- A namespace permits grouping of various entities like classes, objects, functions, and various C++ tokens, etc. under a single name.
- Any user can create separate namespaces of its own and can use them in any other program.
- In the below snippets, namespace std contains declarations for cout, cin, endl, etc. statements.
- using namespace std;
- Namespaces can be accessed in multiple ways:

- using namespace std;
- using std :: cout;

Definition Section:

- It is used to declare some constants and assign them some value.
- In this section, anyone can define your own datatype using primitive data types.
- In #define is a compiler directive which tells the compiler whenever the message is found to replace it with “Factorial\n”.
- typedef int INTEGER; this statement tells the compiler that whenever you will encounter INTEGER replace it by int and as you have declared INTEGER as datatype you cannot use it as an identifier.

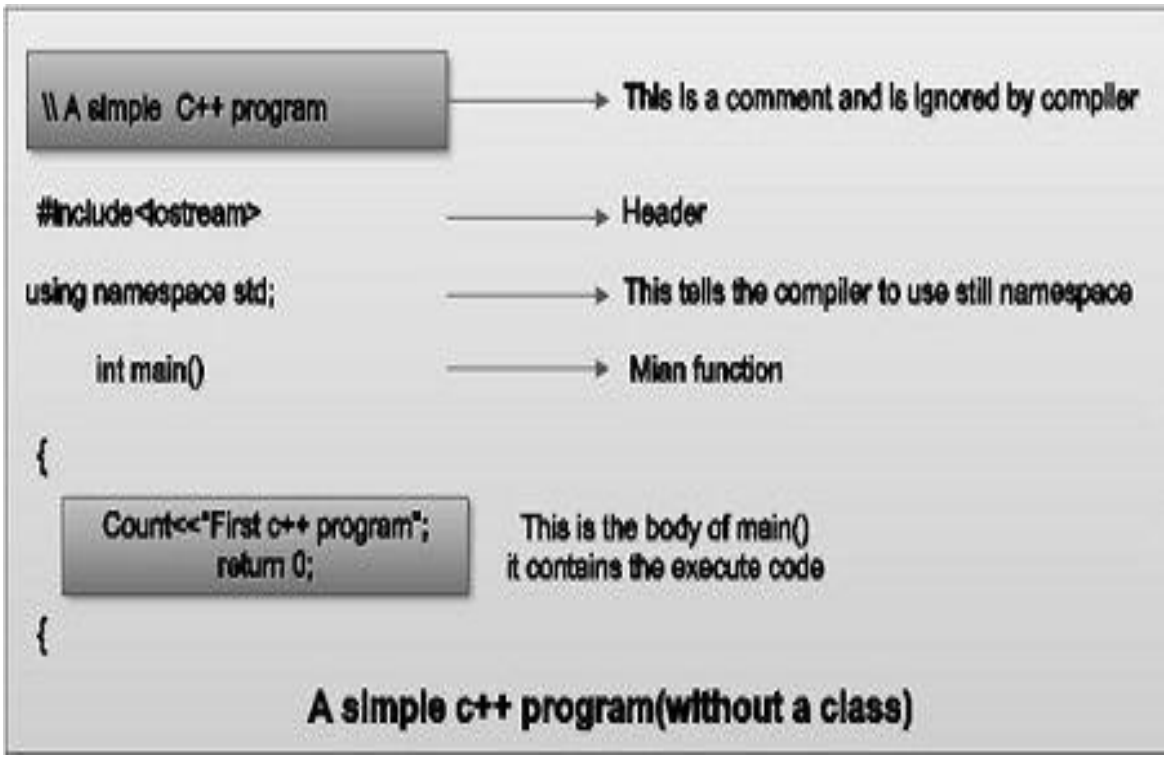
Global Declaration Section:

- Here, the variables and the class definitions which are going to be used in the program are declared to make them global.
- The scope of the variable declared in this section lasts until the entire program terminates.
- These variables are accessible within the user-defined functions also.
- Function Declaration Section:

- It contains all the functions which our main functions need.
- Usually, this section contains the User-defined functions.
- This part of the program can be written after the main function but for this, write the function prototype in this section for the function which for you are going to write code after the main function.

Main Function:

- The main function tells the compiler where to start the execution of the program. The execution of the program starts with the main function.
- All the statements that are to be executed are written in the main function.
- The compiler executes all the instructions which are written in the curly braces {} which encloses the body of the main function.
- Once all instructions from the main function are executed, control comes out of the main function and the program terminates and no further execution occur.



Token

- When the compiler is processing the source code of a C++ program, each group of characters separated by white space is called a token. Tokens are the smallest individual units in a program. A C++ program is written using tokens. It has the following tokens:
- Keywords
- Identifiers
- Constants
- Strings
- Operators

1.2 keywords in C++

Keywords

- Keywords(also known as **reserved words**) have special meanings to the C++ compiler and are always written or typed in short(lower) cases. Keywords are words that the language uses for a special purpose, such as **void**, **int**, **public**, etc. It can't be used for a variable name or function name or any other identifiers. The total count

of reserved keywords is 95. Below is the table for some commonly used C++ keywords.

C++ Keyword			
asm	double	new	switch
auto	else	operator	template
break	enum	private	this
case	extern	protected	throw
catch	float	public	try
char	for	register	typedef
class	friend	return	union
const	goto	short	unsigned
continue	if	signed	virtual
default	inline	sizeof	void
delete	int	static	volatile
do	long	struct	while