CPP Study Material
S.Y.B.Sc.(Computer Science)
Semester – II

Mr. Purushottam S. Dixit

Department of Computer Science
Tuljaram Chaturchand College,
Baramati.

Introduction to C++

- **C++** is a middle-level programming language developed by **Bjarne Stroustrup** starting in 1979 at Bell Labs. As an enhancement to the C language and originally named C with Classes but later it was renamed C++ in 1983.
- **C++** runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX.
- C++ is a super set of C programming with additional implementation of object-oriented concepts.
- C++ is a statically typed, compiled, general-purpose, case-sensitive, free-form programming language that supports procedural, object-oriented, and generic programming.
- C++ is regarded as a middle-level language, as it comprises a combination of both high-level and low-level language features.
- C++ is a superset of C, and that virtually any legal C program is a legal C++ program.

Why to Learn C++

1. C++ is very close to hardware, so you get a chance to work at a low level which gives you lot of control in terms of memory management, better performance and finally a robust software development.

2. C++ programming gives you a clear understanding about Object Oriented Programming. You will understand low level implementation of polymorphism when you will implement virtual tables and virtual table pointers, or dynamic type identification.

3. C++ is one of the every green programming languages and loved by millions of software developers. If you are a great C++ programmer then you will never sit without work and more importantly you will get highly paid for your work.

Applications of C++ Programming

As mentioned before, C++ is one of the most widely used programming languages. It has it's presence in almost every area of software development. I'm going to list few of them here:

- **Application Software Development** - C++ programming has been used in developing almost all the major Operating Systems like Windows, Mac OSX and Linux. Apart from the operating systems, the core part of many browsers like Mozilla Firefox and Chrome have been written using C++. C++ also has been used in developing the most popular database system called MySQL.

- **Programming Languages Development** - C++ has been used extensively in developing new programming languages like C#, Java, JavaScript, Perl, UNIX's C Shell, PHP and Python, and Verilog etc.

- **Computation Programming** - C++ is the best friends of scientists because of fast speed and computational efficiencies.

- **Games Development** - C++ is extremely fast which allows programmers to do procedural programming for CPU intensive functions and provides greater control over hardware, because of which it has been widely used in development of gaming engines.

- **Embedded System** - C++ is being heavily used in developing Medical and Engineering Applications like softwares for MRI machines, high-end CAD/CAM systems etc.

- This list goes on, there are various areas where software developers are happily using C++ to provide great softwares. I highly recommend you to learn C++ and contribute great softwares to the community.

C++ Program Structure

First Simple Program using C++

```
#include <iostream>
using namespace std;
int main()                          / / main() is where program execution begins.
{
    cout << "Hello World";          // prints Hello World
    return 0;
}
```

Let us look at the various parts of the above program −

- ✓ The C++ language defines several header files, which contain information that is either necessary or useful to your program. For this program, the header **<iostream>** is needed.
- ✓ The line **using namespace std;** tells the compiler to use the std namespace. Namespaces are a relatively recent addition to C++.
- ✓ The next line '**// main() is where program execution begins.**' is a single-line comment available in C++. Single-line comments begin with // and stop at the end of the line.
- ✓ The line **int main()** is the main function where program execution begins.
- ✓ The next line **cout << "Hello World";** causes the message "Hello World" to be displayed on the screen.
- ✓ The next line **return 0;** terminates main( )function and causes it to return the value 0 to the calling process.

Compile and Execute C++ Program

Follow the following steps –

- ✓ Save the file as: hello.cpp ,
- ✓ Open a command prompt and go to the directory where you saved the file.
- ✓ Type 'g++ hello.cpp' and press enter to compile your code.
- ✓ If there are no errors in your code the command prompt will take you to the next line and would generate a.out executable file.
- ✓ Now, type 'a.out' to run your program.
- ✓ You will be able to see ' Hello World ' printed on the window.

C++ Identifiers

- A C++ identifier is a name used to identify a variable, function, class, module, or any other user-defined item.
- An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores, and digits (0 to 9).
- C++ does not allow punctuation characters such as @, $, and % within identifiers.
- C++ is a case-sensitive programming language. Thus, **Manpower** and **manpower** are two different identifiers in C++.

Here are some examples of acceptable identifiers −

mohd	zara	abc	move_name	a_123
myname50	_temp	j	a23b9	retVal

Here are some examples of Invalid identifiers −

5abc	Amol@123	roll no

C++ Keywords

- Keywords are the words reserved by the language and having some specific meaning.
- The following list shows the reserved words in C++. These reserved words may not be used as constant or variable or any other identifier names.

asm	else	new	this	auto	enum	operator
throw	bool	explicit	private	true	break	export
protected	try	case	extern	public	typedef	catch
false	register	typeid	char	float	long	typename
class	for	return	union	const	friend	short
unsigned	static	goto	signed	static_cast	continue	if
sizeof	virtual	default	inline	const_cast	void	delete
int	using	volatile	do	reinterpret_cast	struct	wchar_t
double	mutable	switch	while	dynamic_cast	namespace	template

Comments in C++

- Program comments are explanatory statements that you can include in the C++ code. These comments help anyone reading the source code. All programming languages allow for some form of comments.
- C++ supports single-line and multi-line comments. All characters available inside any comment are ignored by C++ compiler.
- C++ comments start with /* and end with */. For example −

```
// This is single line comment

/* This is a comment */

/* C++ comments can also
        * span multiple lines
*/
```

C++ Data Types

While writing program in any language, you need to use various variables to store various information. Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in Memory. Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory.

Primitive Built-in Types

- C++ offers the programmer a rich assortment of built-in as well as user defined data types. Following table lists down seven basic C++ data types −

Type	Keyword
Boolean	bool
Character	char
Integer	int
Floating point	float
Double floating point	double
Valueless	void
Wide character	wchar_t

The following table shows the variable type, how much memory it takes to store the value in memory, and what is maximum and minimum value which can be stored in such type of variables.

Type	Typical Bit Width	Typical Range
char	1byte	-127 to 127 or 0 to 255
unsigned char	1byte	0 to 255
signed char	1byte	-127 to 127
int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295
signed int	4bytes	-2147483648 to 2147483647
short int	2bytes	-32768 to 32767
unsigned short int	2bytes	0 to 65,535
signed short int	2bytes	-32768 to 32767
long int	8bytes	-2,147,483,648 to 2,147,483,647
signed long int	8bytes	same as long int
unsigned long int	8bytes	0 to 4,294,967,295
long long int	8bytes	-(2^63) to (2^63)-1
unsigned long long int	8bytes	0 to 18,446,744,073,709,551,615
float	4bytes	
double	8bytes	
long double	12bytes	
wchar_t	2 or 4 bytes	1 wide character

Variable Definition in C++

- C++ also allows to define various other types of variables, which we will cover in subsequent chapters like **Enumeration, Pointer, Array, Reference, Data structures,** and **Classes**.

- A variable definition tells the compiler where and how much storage to create for the variable. A variable definition specifies a data type, and contains a list of one or more variables of that type as follows −

  type variable_list;

- Here, type must be a valid C++ data type including char, w_char, int, float, double, bool or any user-defined object, etc., and variable_list may consist of one or more identifier names separated by commas.

Some valid declarations are shown here −

int i, j, k;

char c, ch;

float f, salary;

double d;

Variable Scope in C++

- A scope is a region of the program and broadly speaking there are three places, where variables can be declared −
- Inside a function or a block which is called local variables,
- In the definition of function parameters which is called formal parameters.
- Outside of all functions which is called global variables.

Local Variables

Variables that are declared inside a function or block are local variables. They can be used only by statements that are inside that function or block of code. Local variables are not known to functions outside their own.

Global Variables

- Global variables are defined outside of all the functions, usually on top of the program. The global variables will hold their value throughout the life-time of your program.
- A global variable can be accessed by any function. That is, a global variable is available for use throughout your entire program after its declaration.

Operators in C++

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations.

C++ is rich in built-in operators and provide the following types of operators −

- Arithmetic Operators :- + , - , * , / , % , ++ , --
- Relational Operators :- == ,!= , < , > , <= , >=
- Logical Operators :- && , || , !
- Bitwise Operators :- &, | , ^
- Assignment Operators :- = , +=, -=,*=,/=,%=, <<=, >>= , &= , |= , ^=
- Misc Operators :- **sizeof , ?: , (,) , (.) , -> , cast**

C++ Loop Types

There may be a situation, when you need to execute a block of code several number of times. In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on.

Programming languages provide various control structures that allow for more complicated execution paths.

A loop statement allows us to execute a statement or group of statements multiple times and following is the general from of a loop statement in most of the programming languages −

- for

Syntax :

```
for(initilization ; condition ; modification)
{
   // body of for loop
}
```

- While

Syntax

```
while(condition)
{
    // body of the loop
}
```

- do-while

Syntax

```
do
{
}while(condition);
```

C++ decision making statements

- Decision making structures require that the programmer specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.
- Following is the general form of a typical decision making structure found in most of the programming languages −

➢ if

Syntax :

```
if(condition)
{
        // statements
}
```

➢ if-else

Syntax :

```
if(condition)
{
        // statements
}
else
{
        // statemetns
}
```

- Nested if

Syntax :

```
if(condition)
{
        if(condition)
        {
        }
        ….
}
```

- Switch

Syntax :

```
switch(condition)
{
        case Label:
                // statements
                break;
        case Label :
                // statement
                break;
        …
        …
}
```

C++ Functions

- A function is a group of statements that together perform a task. Every C++ program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.
- You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division usually is such that each function performs a specific task.
- A function **declaration** tells the compiler about a function's name, return type, and parameters. A function **definition** provides the actual body of the function.
- The C++ standard library provides numerous built-in functions that your program can call. For example, function **strcat()** to concatenate two strings, function **memcpy()** to copy one memory location to another location and many more functions.
- A function is known with various names like a method or a sub-routine or a procedure etc.

Function Declarations

A function declaration has the following parts −

return_type function_name(parameter list);

Defining a Function

The general form of a C++ function definition is as follows −

```
return_type function_name(parameter list )
{
    body of the function
}
```

Calling a Function

- While creating a C++ function, you give a definition of what the function has to do. To use a function, you will have to call or invoke that function.
- When a program calls a function, program control is transferred to the called function. A called function performs defined task and when it's return statement is executed or when its function-ending closing brace is reached, it returns program control back to the main program.
- To call a function, you simply need to pass the required parameters along with function name, and if function returns a value, then you can store returned value. For example −

Syntax :

Funcation_name(parameters);