1. OVERVIEW

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++ was developed by Bjarne Stroustrup starting in 1979 at Bell Labs in Murray Hill, New Jersey, as an enhancement to the C language and originally named C with Classes but later it was renamed C++ in 1983.
C++ is a superset of C, and that virtually any legal C program is a legal C++ program.
Note: A programming language is said to use static typing when type checking is performed during compile-time as opposed to run-time.

Object-Oriented Programming

C++ fully supports object-oriented programming, including the four pillars of object-oriented development:
☐ Encapsulation

☐ Data hiding

☐ Inheritance

☐ Polymorphism

Standard Libraries

Standard C++ consists of three important parts:
☐ The core language giving all the building blocks including variables, data types and literals, etc.

☐ The C++ Standard Library giving a rich set of functions manipulating files, strings, etc.

☐ The Standard Template Library (STL) giving a rich set of methods manipulating data structures, etc.

The ANSI Standard

The ANSI standard is an attempt to ensure that C++ is portable; that code you write for Microsoft's compiler will compile without errors, using a compiler on a Mac, UNIX, a Windows box, or an Alpha.

The ANSI standard has been stable for a while, and all the major C++ compiler manufacturers support the ANSI standard.

Learning C++

The most important thing while learning C++ is to focus on concepts.
The purpose of learning a programming language is to become a better programmer; that is, to become more effective at designing and implementing new systems and at maintaining old ones.
C++ supports a variety of programming styles. You can write in the style of Fortran, C, Smalltalk, etc., in any language. Each style can achieve its aims effectively while maintaining runtime and space efficiency.

Use of C++

C++ is used by hundreds of thousands of programmers in essentially every application domain.
C++ is being highly used to write device drivers and other software that rely on direct manipulation of hardware under real-time constraints.
C++ is widely used for teaching and research because it is clean enough for successful teaching of basic concepts.
Anyone who has used either an Apple Macintosh or a PC running Windows has indirectly used C++ because the primary user interfaces of these systems are written in C++. C++

2. ENVIORNMENT SETUP

Try it Option Online

You really do not need to set up your own environment to start learning C++ programming language. Reason is very simple, we have already set up C++ Programming environment online, so that you can compile and execute all the available examples online at the same time when you are doing your theory work. This gives you confidence in what you are reading and to check the result with different options. Feel free to modify any example and execute it online.

Try the following example using our online compiler option available at http://www.compileonline.com/

```
#include <iostream>
using namespace std;
int main()
{
cout << "Hello World";
return 0;
}
```

For most of the examples given in this tutorial, you will find `Try it` option in our website code sections at the top right corner that will take you to the online compiler. So just make use of it and enjoy your learning.

Local Environment Setup

If you are still willing to set up your environment for C++, you need to have the following two softwares on your computer.

Text Editor:

This will be used to type your program. Examples of few editors include Windows Notepad, OS Edit command, Brief, Epsilon, EMACS, and vim or vi.

Name and version of text editor can vary on different operating systems. For example, Notepad will be used on Windows and vim or vi can be used on windows as well as Linux, or UNIX.

The files you create with your editor are called source files and for C++ they typically are named with the extension .cpp, .cp, or .c.

A text editor should be in place to start your C++ programming.

C++ Compiler:

This is an actual C++ compiler, which will be used to compile your source code into final executable program.

Most C++ compilers don't care what extension you give to your source code, but if you don't specify otherwise, many will use .cpp by default.

Most frequently used and free available compiler is GNU C/C++ compiler, otherwise you can have compilers either from HP or Solaris if you have the respective Operating Systems.

Installing GNU C/C++ Compiler:

UNIX/Linux Installation:

If you are using `Linux or UNIX` then check whether GCC is installed on your system by entering the following command from the command line:

```
$ g++ -v
```

If you have installed GCC, then it should print a message such as the following:

```
Using built-in specs.
Target: i386-redhat-linux
Configured with: ../configure --prefix=/usr .......
Thread model: posix
```

```
gcc version 4.1.2 20080704 (Red Hat 4.1.2-46)
```
If GCC is not installed, then you will have to install it yourself using the detailed instructions available at http://gcc.gnu.org/install/ .

Mac OS X Installation:

If you use Mac OS X, the easiest way to obtain GCC is to download the Xcode development environment from Apple's website and follow the simple installation instructions.
Xcode is currently available at developer.apple.com/technologies/tools/.

Windows Installation:

To install GCC at Windows you need to install MinGW. To install MinGW, go to the MinGW homepage, www.mingw.org, and follow the link to the MinGW download page. Download the latest version of the MinGW installation program which should be named MinGW-<version>.exe. C++

While installing MinGW, at a minimum, you must install gcc-core, gcc-g++, binutils, and the MinGW runtime, but you may wish to install more.
Add the bin subdirectory of your MinGW installation to your PATH environment variable so that you can specify these tools on the command line by their simple names.
When the installation is complete, you will be able to run gcc, g++, ar, ranlib, dlltool, and several other GNU tools from the Windows command line. C++
6

3. BASIC SYNTAX

When we consider a C++ program, it can be defined as a collection of objects that communicate via invoking each other's methods. Let us now briefly look into what a class, object, methods, and instant variables mean.

☐ Object - Objects have states and behaviors. Example: A dog has states - color, name, breed as well as behaviors - wagging, barking, and eating. An object is an instance of a class.

☐ Class - A class can be defined as a template/blueprint that describes the behaviors/states that object of its type support.

☐ Methods - A method is basically a behavior. A class can contain many methods. It is in methods where the logics are written, data is manipulated and all the actions are executed.

☐ Instant Variables - Each object has its unique set of instant variables. An object's state is created by the values assigned to these instant variables.

C++ Program Structure:

Let us look at a simple code that would print the words *Hello World*.

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

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

1. The C++ language defines several headers, which contain information that is either necessary or useful to your program. For this program, the header `<iostream>` is needed.

2. The line `using namespace std;` tells the compiler to use the std namespace. Namespaces are a relatively recent addition to C++.

3. 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.

4. The line int main() is the main function where program execution begins.

5. The next line cout << "This is my first C++ program."; causes the message "This is my first C++ program" to be displayed on the screen.

6. The next line return 0; terminates main() function and causes it to return the value 0 to the calling process.

Compile & Execute C++ Program:

Let's look at how to save the file, compile and run the program. Please follow the steps given below:

1. Open a text editor and add the code as above.

2. Save the file as: hello.cpp

3. Open a command prompt and go to the directory where you saved the file.

4. 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.

5. Now, type 'a.out' to run your program.

6. You will be able to see ' Hello World ' printed on the window.

```
$ g++ hello.cpp
$ ./a.out
Hello World
```

Make sure that g++ is in your path and that you are running it in the directory containing file hello.cpp.

You can compile C/C++ programs using makefile. For more details, you can check our 'Makefile Tutorial'.

Semicolons & Blocks in C++

In C++, the semicolon is a statement terminator. That is, each individual statement must be ended with a semicolon. It indicates the end of one logical entity.

For example, following are three different statements:

```
x = y; C++
8
```

```
y = y+1;
add(x, y);
```
A block is a set of logically connected statements that are surrounded by opening and closing braces. For example:
```
{
cout << "Hello World"; // prints Hello World
return 0;
}
```
C++ does not recognize the end of the line as a terminator. For this reason, it does not matter where you put a statement in a line. For example:
```
x = y;
y = y+1;
add(x, y);
```
is the same as
```
x = y; y = y+1; add(x, y);
```
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
```
C++ Keywords
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	reinterpret_cast	typename
class	for	return	union
const	friend	short	unsigned
const_cast	goto	signed	using
continue	if	sizeof	virtual
default	inline	static	void
delete	int	static_cast	volatile
do	long	struct	wchar_t
double	mutable	switch	while
dynamic_cast	namespace	template	

Trigraphs

A few characters have an alternative representation, called a trigraph sequence. A trigraph is a three-character sequence that represents a single character and the sequence always starts with two question marks.
Trigraphs are expanded anywhere they appear, including within string literals and character literals, in comments, and in preprocessor directives.

Following are most frequently used trigraph sequences:

C++

Trigraph	Replacement
??=	#
??/	\
??'	^
??(	[
??)	]
??!	|
??<	{
??>	}
??-	~

All the compilers do not support trigraphs and they are not advised to be used because of their confusing nature.

Whitespace in C++

A line containing only whitespace, possibly with a comment, is known as a blank line, and C++ compiler totally ignores it.
Whitespace is the term used in C++ to describe blanks, tabs, newline characters and comments. Whitespace separates one part of a statement from another and enables the compiler to identify where one element in a statement, such as int, ends and the next element begins. Statement 1:

```
int age;
```

In the above statement there must be at least one whitespace character (usually a space) between int and age for the compiler to be able to distinguish them.
Statement 2:

```
fruit = apples + oranges; // Get the total fruit C++
11
```

In the above statement 2, no whitespace characters are necessary between fruit and =, or between = and apples, although you are free to include some if you wish for readability purpose.

4. 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 a comment */
/* C++ comments can also
* span multiple lines
*/
```

A comment can also start with //, extending to the end of the line. For example:

```
#include <iostream>
using namespace std;
main()
{
cout << "Hello World"; // prints Hello World
return 0;
}
```

When the above code is compiled, it will ignore `// prints Hello World` and final executable will produce the following result:

```
Hello World
```

Within a /* and */ comment, // characters have no special meaning. Within a // comment, /* and */ have no special meaning. Thus, you can "nest" one kind of comment within the other kind.