Week 1: Lecture 1

Introduction to Object Oriented Programming

The prime purpose of C++ programming is to add object orientation to the C programming language.

Object Oriented Programming in C++ Object Oriented programming is a programming style that is associated with the concept of Class, Objects and various other concepts revolving around these two, like Inheritance, Polymorphism, Abstraction, Encapsulation etc.

There are a few principle concepts that form the foundation of object-oriented programming −

Object

This is the basic unit of object oriented programming. That is both data and function that operate on data are bundled as a unit called as object.

Class

When you define a class, you define a blueprint for an object. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

Abstraction

Data abstraction refers to, providing only essential information to the outside world and hiding their background details, i.e., to represent the needed information in program without presenting the details.

For example, a database system hides certain details of how data is stored and created and maintained. Similar way, C++ classes provides different methods to the outside world without giving internal detail about those methods and data.

Encapsulation

Encapsulation is placing the data and the functions that work on that data in the same place. While working with procedural languages, it is not always clear which functions work on which variables but object-oriented programming provides you framework to place the data and the relevant functions together in the same object.

Inheritance

One of the most useful aspects of object-oriented programming is code reusability. As the name suggests Inheritance is the process of forming a new class from an existing class that is from the existing class called as base class, new class is formed called as derived class.

This is a very important concept of object-oriented programming since this feature helps to reduce the code size.

Polymorphism

The ability to use an operator or function in different ways in other words giving different meaning or functions to the operators or functions is called polymorphism. Poly refers to many. That is a single function or an operator functioning in many ways different upon the usage is called polymorphism.

Overloading

The concept of overloading is also a branch of polymorphism. When the exiting operator or function is made to operate on new data type, it is said to be overloaded.

First program is C++

```
// This is my first program is C++
/* this program will illustrate different components of a
simple program in C++ */

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

When the above program is compiled, linked and executed, the following output is displayed on the screen.

```
Hello World!
```

Various components of this program are discussed below:

Comments

First three lines of the above program are comments and are ignored by the compiler. Comments are included in a program to make it more readable. If a comment is short and can be accommodated in a single line, then it is started with double slash sequence in the first line of the program. However, if there are multiple lines in a comment, it is enclosed between the two symbols /* and */

#include <iostream>

The line in the above program that start with # symbol are called directives and are instructions to the compiler. The word include with '#' tells the compiler to include the file iostream into the file of the above program. File iostream is a header file needed for input/ output requirements of the program. Therefore, this file has been included at the top of the program.

using namespace std;

All the elements of the standard C++ library are declared within std. This line is very frequent in C++ programs that use the standard library.

int main ( )

The word main is a function name. The brackets ( ) with main tells that main ( ) is a function. The word int before main ( ) indicates that integer value is being returned by the function main (). When program is loaded in the memory, the control is handed over to function main ( ) and it is the first function to be executed.

Curly bracket and body of the function main ( )

A C++ program starts with function called main(). The body of the function is enclosed between curly braces. The program statements are written within the brackets. Each statement must end by a semicolon, without which an error message in generated.

cout<<"Hello World!";

This statement prints our "Hello World!" message on the screen. cout understands that anything sent to it via the << operator should be printed on the screen.

return 0;

This is a new type of statement, called a return statement. When a program finishes running, it sends a value to the operating system. This particular return statement returns the value of 0 to the operating system, which mean “everything went okay!”.

Printing Multiple Lines of Text with a Single Statement

```
/* This program illustrates how to print multiple lines of text with a single statement */

#include <iostream>
using namespace std;
int main()
{
cout << "Welcome\nto\nC++";
return 0;
}
```

Output:

```
Welcome
to
C++
```

The characters print exactly as they appear between the double quotes. However, if we type \n, the characters \n are not printed on the screen. The backslash (\) is called an **escape character**. It indicates that a "special" character is to be output. When a backslash is encountered in a string of characters, the next character is combined with the backslash to form an **escape sequence**. The escape sequence \n means **newline**. It causes the cursor to move to the beginning of the next line on the screen. The following table gives a listing of common escape sequences.

Escape Sequence	**Description**
\n	Newline
\t	Horizontal tab
\a	Bell (beep)
\\	Backslash
\'	Single quote
\"	Double quote