INTRODUCTION TO C++

Introduction to C++

C++, as we all know is an extension to C language and was developed by **Bjarne stroustrup** at bell labs. C++ is an intermediate level language, as it comprises a confirmation of both high level and low level language features. C++ is a statically typed, free form, multiparadigm, compiled general-purpose language.

C++ is an **Object Oriented Programming language** but is not purely Object Oriented. Its features like Friend and Virtual violate some of the very important OOPS features, rendering this language unworthy of being called completely Object Oriented. It's a middle level language.

Object Oriented Programming

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.

Now, let us discuss some of the main features of Object Oriented Programming which you will be using in C++ (technically).

- Objects
- Classes
- Abstraction
- Encapsulation
- Inheritance
- Overloading
- Exception Handling

Objects

An object is an identifiable entity with some characteristics and behavior. Objects are the basic unit of OOP. They are instances of class, which have data members and use various member functions to perform tasks.

Example: - 'Orange' is an object with some characteristic like it is spherical shaped, its color is orange etc. Its behavior is juicy/citrus and it testes sweet-sour.

Class

It is similar to structures in C language. Class can also be defined as user defined data type but it also contains functions in it. So, class is basically a blueprint for object. It declares & defines what data variables the object will have and what operations can be performed on the class's object.

Example:- Mango, apple & orange are member of the class 'Fruits'.

Abstraction

Abstraction refers to showing only the essential features of the application and hiding the details. In C++, classes can provide methods to the outside world to access & use the data variables, keeping the variables hidden from direct access, or classes can even declare everything accessible to everyone, or maybe just to the classes inheriting it. This can be done using access specifies.

Example:- A 'switch board'. You only press certain switches according to your requirement. What happening inside how it is happening etc. You needn't know. So this is abstraction. You know only the essential things to operate on switch board without knowing the background detail of switchboard.

Encapsulation

It can also be said data binding. Encapsulation is all about binding the data variables and functions together in class.

Example:- Medicine Capsule contain the different type of medicine in a single container, that means it encapsulate the medicine.

Inheritance

Inheritance is a way to reuse once written code again and again. The class which is inherited is called the **Base** class & the class which inherits is called the **Derived** class. They are also called parent and child class. So when, a derived class inherits a base class, the derived class can use all the functions which are defined in base class, hence making code reusable.

Example:- We are humans. We inherit from the class 'Humans' certain properties, such as ability to speak, breath, eat, drink etc. But these properties are not unique to humans. The class 'Human' inherit these properties from class 'mammal' which again inherits some of its properties from another class 'Animal'.

Polymorphism

It is a feature, which lets us create functions with same name but different arguments, which will perform different actions. That means, functions with same name, but functioning in different ways. Or, it also allows us to redefine a function to provide it with a completely new definition. You will learn how to do this in details soon in coming lessons.

Example:- Suppose if you are in class room that time you behave like a student, when you are in market at that time you behave like a customer, when you at your home at that time you behave like a son or daughter, Here one person present in different-different behaviors.

Exception Handling

Exception handling is a feature of OOP, to handle unresolved exceptions or errors produced at runtime.

Example:-

(a) you provide a web form for users to fill in and submit.but incase there are a lot of conditions to be handled and the conditions keeps changing periodically, you use exception handling to simplify the process

(b) Database connectivity uses exception handling (why???) this is because the reason for database connectivity failure cannot be predicted and handled as it can be caused by many variables such as power failure, unreachable server, and failure at client front/back end and so on.

C VS C++

No.	C	C++
1)	C follows the **procedural style programming.**	C++ is multi-paradigm. It supports both **procedural and object oriented.**
2)	Data is less secured in C.	In C++, you can use modifiers for class members to make it inaccessible for outside users.
3)	C follows the **top-down approach.**	C++ follows the **bottom-up approach.**
4)	C does not support function overloading.	C++ supports function overloading.
5)	In C, you can't use functions in structure.	In C++, you can use functions in structure.
6)	C does not support reference variables.	C++ supports reference variables.
7)	In C, **scanf() and printf()** are mainly used for input/output.	C++ mainly uses stream **cin and cout** to perform input and output operations.
8)	Operator overloading is not possible in C.	Operator overloading is possible in C++.
9)	C programs are divided into **procedures and modules**	C++ programs are divided into **functions and classes.**
10)	C does not provide the feature of namespace.	C++ supports the feature of namespace.
11)	Exception handling is not easy in C. It has to perform using other functions.	C++ provides exception handling using Try and Catch block.

Applications of C++

By the help of C++ programming language, we can develop different types of secured and robust applications:

- Window application
- Client-Server application
- Device drivers
- Embedded firmware etc

First C++ program

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
 system("cls");
 cout<<"\n Welcome to c++ Programming Language";
 cout<<endl<<"C++ Super set of C Language";
 cout<<endl<<"C is Sub Set of C++ Programming Language"<<endl;
 system("pause");
}
```

Header files are included at the beginning just like in C program. Here iostream is a header file which provides us with input & output streams. Header files contained predeclared function libraries, which can be used by users for their ease.

main(), is the function which holds the executing part of program its default return type is int.

cout <<, is used to print anything on screen, same as printf() in C language. **cin** and **cout** are same as scanf() and printf(), only difference is that you do not need to mention format specifiers like, %d for int etc, in cout & cin.

Comments in C++ Program

For single line comments, use // before mentioning comment, like.
cout<<"Single line"; // This is single line comment

For multiple line comment, enclose the comment between /* and */
/* This is a
Multiple line
Comment */

More C++ Statements

Let us consider a slightly more complex C++ program. Assume that we would like to read two numbers from the keyboard and display their average on the screen.

```
#include<iostream.h>
#include<conio.h>
int main()
{
 float num1,num2,sum,avg;
 clrscr();
 cout<<"\n Enter two number :- ";
 cin>>num1;
 cin>>num2;
 sum=num1+num2;
 avg=sum/2;
 cout<<"\n Sum :- "<<sum;
 cout<<"\n Average :- "<<avg;
 getch();
 return 0;
}
```

Input Operator

The statement
cin>>num1;
is an input statement and causes the program to wait for the user to type in a number. The number keyed in is placed in the variable1. The identifier cin (pronounced 'C' in) is a predefined object in C++ that corresponds to the standard input stream. Here, this stream represents the keyboard.
The operator >> is known as extraction or get from operator. It extracts (or takes) the value from the keyboard and assigns it to the variable on its right. This corresponds to the familiar scanf() operation. Like <<, the operator >> can also be overloaded.

Output Operator

The only statement in First C++ Program is an output statement. The statement

cout<<"Welcome to Ssi Indore";

causes the string in quotation marks to be displayed on the screen. This statement introduces two new C++ features, cout and <<. The identifier cout(pronounced as C out) is a predefined object that represents the standard output stream in C++. Here the standard output stream represents the screen. It is also possible to redirect the output to other output devices. The operator << is called the insertion or put to operator. It inserts (or sends) the contents of the variable on its right to the object on its left.

Creating Classes in C++

Classes name must start with capital letter, and they contain data variables and member functions. This is a mere introduction to classes; we will discuss classes in detail throughout the C++ tutorial.

Creating Class using struct keyword

```
#include<iostream.h>
#include<conio.h>
struct item
{
 int number;
 float cost;
void getdata()
 {
  cout<<"\n Enter value of number and cost :-";
  cin>>number>>cost;
 }
 void putdata()
 {
  cout<<"\n Number:-"<<number;
  cout<<"\n Cost  :-"<<cost;
 }
};
void main()
{

 struct item p1;
 clrscr();
 p1.getdata();
 p1.putdata();
 getch();
}
```

Creating Class using class keyword

```
#include<iostream.h>
#include<conio.h>
class Persion
{
 private:
```

```
    int pid;
    char pname;
    int age;
  public:
    void getdata()
    {
    cout<<"\n Enter Persion Id :- ";
    cin>>pid;
    cout<<"\n Enter Persion Name :- ";
    cin>>pname;
    cout<<"\n Enter Persion Age :- ";
    cin>>age;
    }
    void display()
    {
      cout<<"\n Persion Id :- "<<pid;
      cout<<"\n Persion pname :- "<<pname;
      cout<<"\n Persion age :- "<<age;
    }
}p1; //Global Object;
void main()
{
  clrscr();
  p1.getdata();
  p1.display();
  getch();
}
```

@ Contents Developed By

Mr. Nitesh Jain
(M.Phil. (CS), M.Sc. (IT))