Object-Oriented Programming in C++

(Summer Internship 2020)

Presented by: -

Nishikanta Ray

B.Tech (ECE), 2nd Semester

SiC#: 190410294

Silicon Institute of Technology

Bhubaneswar

1. Table of Contents

1 Introduction

- **C++** is a cross-platform language that can be used to create high-performance applications.
- **C++** was developed by Bjarne Stroustrup, as an extension to the C language.
- **C++** gives programmers a high level of control over system resources and memory.

Some of the ***features & key-points*** to note about the programming language are as follows:

- **Simple**: It is a simple language in the sense that programs can be broken down into logical units and parts, has a rich library support and a variety of data-types.
- **Machine Independent but Platform Dependent**: A C++ executable is not platform-independent (compiled programs on Linux won't run on Windows), however they are machine independent.
- **Mid-level language**: It is a mid-level language as we can do both systems-programming (drivers, kernels, networking etc.) and build large-scale user applications (Media Players, Photoshop, Game Engines etc.)
- **Rich library support**: Has a rich library support (Both standard ~ built-in data structures, algorithms etc.) as well 3rd party libraries (e.g. Boost libraries) for fast and rapid development.
- **Speed of execution**: C++ programs excel in execution speed. Since, it is a compiled language, and also hugely procedural. Newer languages have extra in-built default

features such as garbage-collection, dynamic typing etc. which slow the execution of the program overall. Since there is no additional processing overhead like this in C++, it is blazing fast.

- **Pointer and direct Memory-Access**: C++ provides pointer support which aids users to directly manipulate storage address. This helps in doing low-level programming (where one might need to have explicit control on the storage of variables).
- **Object-Oriented**: One of the strongest points of the language which sets it apart from C. Object-Oriented support helps C++ to make maintainable and extensible programs. i.e. Large-scale applications can be built. Procedural code becomes difficult to maintain as code-size grows.
- **Compiled Language**: C++ is a compiled language, contributing to its speed.
-

Object-oriented programming –

As the name suggests uses objects in programming .Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism, etc in programming. The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.

2 Objectives

- To understand how C++ improves C with object-oriented features.
- To learn how to write inline functions for efficiency and performance.
- To learn the syntax and semantics of the C++ programming language.
- To learn how to design C++ classes for code reuse.
- To learn how to implement copy constructors and class member functions.
- To understand the concept of data abstraction and encapsulation.
- To learn how to overload functions and operators in C++.
- To learn how containment and inheritance promote code reuse in C++.
- To learn how inheritance and virtual functions implement dynamic binding with polymorphism.
- To learn how to design and implement generic classes with C++ templates.
- To learn how to use exception handling in C++ programs.

3 Contents of the Program

Characteristics of an Object Oriented Programming language:

3.1 Class:

The building block of C++ that leads to Object-Oriented programming is a Class. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A class is like a blueprint for an object.

3.2 Object:

An Object is an identifiable entity with some characteristics and behavior . An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

3.3 Encapsulation:

In normal terms, Encapsulation is defined as wrapping up of data and information under a single unit. In Object-Oriented Programming, Encapsulation is defined as

binding together the data and the functions that manipulate them.

3.4 Abstraction:

Data abstraction is one of the most essential and important features of object-oriented programming in C++. Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation.

3.5 Polymorphism:

The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form.

3.6 Inheritance:

The capability of a class to derive properties and characteristics from another class is called Inheritance. Inheritance is one of the most important features of Object-Oriented Programming.

- **Sub Class**: The class that inherits properties from another class is called Sub class or Derived Class.
- **Super Class**: The class whose properties are inherited by sub class is called Base Class or Super class.

- **Reusability**: Inheritance supports the concept of “reusability”, i.e. when we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.

3.7 File Handling In C++

Files are used to store data in a storage device permanently. File handling provides a mechanism to store the output of a program in a file and to perform various operations on it.

A stream is an abstraction that represents a device on which operations of input and output are performed. A stream can be represented as a source or destination of characters of indefinite length depending on its usage.

In C++ we have a set of file handling methods. These include ifstream, ofstream, and fstream. These classes are derived from fstrembase and from the corresponding iostream class. These classes, designed to manage the disk files, are declared in fstream and therefore we must include fstream and therefore we must include this file in any program that uses files.

In C++, files are mainly dealt by using three classes fstream, ifstream, ofstream.

- ofstream: This Stream class signifies the output file stream and is applied to create files for writing information to files
- ifstream: This Stream class signifies the input file stream and is applied for reading information from files
- fstream: This Stream class can be used for both read and write from/to files.

All the above three classes are derived from fstreambase and from the corresponding iostream class and they are designed specifically to manage disk files.
C++ provides us with the following operations in File Handling:

- Creating a file: open()
- Reading data: read()
- Writing new data: write()
- Closing a file: close()

4 Methodology

Day 1: Properties such as Abstraction, Encapsulation, Inheritance and Polymorphism were discussed.

Also they discussed what is functional decomposition, what is class decomposition etc.

Day 2: How use Inline function. what call by value, call by reference and pointer were discussed.

Day 3: Discussed about the Different Types of specifier and how to use in the program.

Day 4: Various uses and statements for implementing Difference types of Constructor and Destructor.

Day 5: Difference between global object and local objects. Introduction to concepts of Anonymous object.

Day 6: Uses of constant, Static members function and 'this' pointer.

Day 7: Syntax for overloading and how to use operator overloading in function were discussed.

Day 8: Discussed about friend function and how to use in two class.

Day 9: Various rules and arguments for implementing different types of type conversion were discussed.

Day 10: Discussed about how to use virtual objects.

Day 11: what is inheritance and types, how to use in program were discussed.

Day 12: How access specifier were use in inheritance were discussed.

Day 13: Difference between various types inheritance were discussed.

Day 14: Techniques to initiate base class constructor of base class and how to use virtual function were discussed.

Day 15: File Handling and error handling using try catch throw method were discussed.

Day 16: Hands on practice how to use file in c++ and real life problems.

5 Details of Works done during the Internship (on daily/weekly basis)

Day 1: Assignment on array, pointer, structure was given and some output questions were given.

Day 2: On the spot quiz was conducted, and assignment on output question were given.

Day 3: Assignment on array of was given and output questions were given.

Day 4: Assignment on class and object using constructor and quiz on output was held.

Day 5: Test on the topics discussed before was conducted.

Day 6: Assignment was given.

Day 7: Assignment on friend function was given.

Day 8: Assignment on operator overloading was given.

Day 9: Assignment on various type conversion was given and quiz on output questions was taken.

Day 10: No work.

Day 11: Quiz was held on output question.

Day 12: Assignment on multiple and hierarchical inheritance was taken.

Day 13: Assignment was given on virtual function.

Day 14: Assignment was given on exception handling.

Day 15: Assignment on file handling was given.

6 Outcomes of the program

- Understood the difference between the top-down and bottom-up approach.
- Understood the object-oriented programming approach in connection with C++.
- Understood the features of C++ supporting object oriented programming.
- Understood the process of data file manipulations using C++.
- Understood how to produce object oriented Software using C++.
- Understood advance features of C++ Specifically I/O, operator overloading.

7 Conclusions

We learnt the concepts of the object oriented programming. In further classes we learned to implement programs in c++ and also got to know its uses. Further we got to know about some terms like Constructor and destructor and used them in programs. Global objects, anonymous object and local object were taught. Next we got to know about friend function which was very useful in writing programs. Learned about type conversion and how to address the problems related to it. Inheritance and data abstraction were some important concepts which helped in understanding the OOPS in much better way. Further we learnt about uses of virtual function, polymorphism, multiple inheritance programs and abstract class. At last class we got to know about The most important concept in C++ which is file handling. Overall it was a great program, Very informative and we got to know about many things related to C++. Regular assignments were given and teachers were very cooperative.

8 References:

- Geeks for geeks
- Tutorials point
- Java point