C++ PROGRAMMING

DURATION:

5 Days

COURSE DESCRIPTION:

Programming in C++ offers a total immersion approach to learning C++. The course starts with

Basic OO concepts, and then introduces their implementation in C++. The language is presented,

not as an extension of C, but as a demonstration of how object-oriented development promotes a new, productive way of thinking. A carefully selected set of features important for object-oriented development is presented first, while more advanced features are dealt with when the students have a solid grasp on the fundamentals. A large part of the course consists of exercises.

WHO SHOULD ATTEND:

This course is for programmers who have had experience in any programming language or have been tasked with a C++ programming project, and other technical types including managers and customer support engineers who need to know Object Oriented Programming and who wants to continue java,.net etc.

PREREQUISITES:

Familiarity with a structured (preferably C) or object-oriented language is a prerequisite for this course.

COURSE OUTLINE:

Day 1

sbmadake@gmail.com +91 9922292223

Session 1: OOPS Buzzword

• Introduction

• Why do we need object oriented programming?

• Procedural Languages

1. Division into functions

2. Problems with structured programming

• An object-oriented approach

• Characteristics of object-oriented programming

1. Abstraction

2. Objects

3. Classes

4. Encapsulation

5. Inheritance

6. Polymorphism

7. Dynamic Binding

8. Message Communication

• Benefits of OOP

• Application of OOP

Session 2: C++ Programming Basics

• What is C++ ?

• Application of C++

• A simple C++ program

sbmadake@gmail.com +91 9922292223

• Programming Features

1. Comments

2. Output Operator

3. The iostream.h File

4. Return statement

• More statements

1. Input Operator

2. Cascading of I/O operators

• An Example with class

• Structure of C++ program

• Tokens

Keywords

1. Identifiers

• Basic Data Types

• User Defined Data Types

1. Structure and Classes

2. Enumerated Data Type

• Derived Data Types

1. Arrays

2. Functions

3. Pointers

4. Symbolic Constants

5. Type compatibility

6. Reference variables

7. Passing by reference

8. Scope Resolution operator

9. Member Dereferencing operators

10. Memory management operators

11. Manipulators

Session 3: Programming Constructs

• Introduction

• Sequence construct

• Selection construct

1. The if statement

2. Nested ifs

3. The if-else-if statements

4. The switch statement

• Iteration construct

1. The while statement

2. The do..while statement

3. For statement

• Jump statement

1. The return statement

2. The goto statement

3. The break statement

4. The exit( ) function

5. The continue statement

Day 2

Session 4: Functions

• Introduction

• Function Declarations

• Function Definitions

• Inline Functions

• Argument Passing

• Value Return

• Array arguments

• Default arguments

• Function overloading

Session 5: Classes and Objects

• Introduction

• C structures

1. Limitations of C structures

2. Extensions to structures

• Specifying a class

1. A simple class example

2. Creating objects

3. Accessing class members

• Defining member functions

1. Outside the class definition

2. Inside the class definition

• A C++ program with class

1. Making an outside function inline

• Nesting of member functions

• Private member functions

• Arrays within a class

• Memory allocation for objects

• Static data members

• Static member functions

• Array of object

• Objects as function arguments

• Friend functions

• Pointer to members

• C++ class constructor

• C++ class destructor

• Dynamic memory allocation of classes

• Copy constructor

Day 3

Session 6: Operator Overloading and Type conversion

• Introduction

• Defining operator overloading

1. Overloading unary operators

2. Overloading binary operators

3. Overloading a comparison operators

4. Overloading the & operator

• Manipulation of strings using operators

• Rules for overloading operations

• Type conversion

1. Basic to class Type

2. Class to basic type

3. Once class to another class

• A data conversion example

Session 7: Inheritance

• Introduction

Advantages

1. Derived classes

2. Access Regions and inheritance

3. Instantiation

4. Purpose of derived class

5. Constructors and destructors

6. Sibling classes

• Accessing members when using inheritance

• Multilevel inheritance

• Multiple inheritance

• Hierarchical inheritance

• Virtual base classes

Session 8: Virtual functions and Polymorphism

• Introduction

• Virtual Functions

• Pure virtual functions

• Abstract classes

• Rules for virtual functions

Day 4

Session 9: Console Input/output

• Introduction

• C++ Streams

• C++ Stream Classes

Unformatted I/O operations

1. Overloaded operators >> and <<

2. put( ) and get( ) functions

3. getline( ) and write( ) functions

• Formatted console I/O operations

1. Defining field: width( )

2. Setting precision: precision( )

3. Filling and padding: fill( )

4. Formatting flags, Bit-fields and setf( )

5. Displaying trailing zeros and plus sign

• Managing output with manipulators

Session 10 : File Input/output

• Introduction

• Fstream.h and the file classes

• Opening and closing a file

• Reading and writing text files

• Binary I/O

1. get( ) and put( )

2. read( ) and write( )

3. More get( ) functions

4. getline( )

• Detecting EOF

• The ignore ( )

• peek( ) and putback( )

• Flush( )

• Random access

1. Obtaining the current file position

2. I/O status

• Customized I/O and files

Session 11: Templates

• Introduction

• Generic Functions

1. A function with two generic types

2. Explicitly overloading a generic function

3. Generic Function Restrictions

• Applying Generic Functions

• A generic sort

1. Compacting an array

• Generic classes

1. An example with two generic data types

• Creating a generic array class

Day 5

Session 12. Exception Handling

• Introduction

• Exception-Handling Fundamentals

Using Multiple catch statements

• Exception-Handling Options

1. Catching all exceptions

2. Restricting exceptions

3. Rethrowing an exception

• Applying Exception Handling

Session 13. Dynamic Casting and Runtime Type Identification

• Getting and using RTTI

• dynamic_cast operator

• Const_cast operator

• rreinterpret_cast operator

• Static_cast operator

Session 14. Standard Template Library

• Introducing STL

• Using Iterators and Containers

• Using Strings

• Working with Numerics

• Working with Files and Streams

• Examining Algorithms

sbmadake@gmail.com +91 9922292223