|  |
| --- |
| Logo  Description automatically generated with medium confidence  **Course Data Sheet** **C++ Programming and OOPS**    Document ID: **CDS/COE/CPP03** |
| |  |  |  |  | | --- | --- | --- | --- | | **Version** | **Owner** | **CDS Owner** | **Last Edited** | | **CEE v1.2** | **eAge Technologies India Pvt. Ltd.** | **Animesh, Vijay, Mubarak, and Naveed** | **22.04.2024** |   **CONFIDENTIALITY:**  This training content document is confidential and is intended, among other things, to present a general outline of eAge Technologies India Private Limited and its services. The contents are not to be reproduced or distributed to the public or press. Each person who has received a copy of this document (whether or not such person purchases any securities) is deemed to have agreed: (i) not to reproduce or distribute this document, in whole or in part, without the prior written consent of eAge Technologies India Private Limited, other than to legal, tax, financial and other advisors on a need to know basis, (ii) if such person has not purchased securities, to return this presentation to eAge Technologies India Private Limited upon its request, (iii) without the prior written consent of eAge Technologies India Private Limited, not to disclose any information contained in this presentation except to the extent that such information was (a) previously known by such person through a source (other than eAge Technologies India Private Limited) not bound by any obligation to keep such information confidential, (b) in the public domain through no fault of such person, or (c) lawfully obtained at a later date by such person from sources (other than eAge Technologies India Private Limited) not bound by any obligation to keep such information confidential, and (iv) to be responsible for any disclosure of this document, or the information contained herein, by such person or any of its employees, agents or representatives. All rights to this information is reserved. |
|  |

**Compiler: g++ on Linux**

**Standards: C++11 (Modern C++)**

**Course Overview**

A perfectly designed and custom course which, focuses on designing and developing solving software requirement using C++ to enhance the skills required in every day programming challenge. This makes it significantly different from most C++ courses by focusing on C++ internals above design patterns principles and idoms. This customized training is backed up by a comprehensive coverage demonstrating the best practices in C++ standards 98/+03/+11/+14.

**Duration: TBD**

**Key Takeaway:**

**Delegates will learn:**

1. **Will learn how to code from an architecture point of view.**
2. **Will understand how C++ is different from C.**
3. **Will learn the meaning and implementation of OOPs.**
4. **Will understand how to use templates, STL.**
5. **Will learn how to debug the c++ code and profile the code also.**

**Course Materials:**

* Soft and online references

**Lab/Environment**

1. **G++ compiler 9.X plus**
2. **Visual studio code for Linux**
3. **Ubuntu 20.x plus**

**Lab requirements:**

**Native environment on participant’s PC**

1. **G++ compiler 9.X plus**
2. **Visual studio code for Linux**
3. **Ubuntu 20.x plus**

**Course Material:**

Soft and Online reference

**Detail Course Outline**

**Day 1**

**An Overview of C++**

* What is Object Oriented Programming?
* Versions of C++ available in the market and compiler support
* Two versions of C++
* C++ Console I/O
* C++ Comments
* Classes: A First Look
* Some Differences between C and C++
* Introducing function overloading
* C++ keyboards

# ARRAYS, POINTERS AND REFERENCES

* Arrays of objects
* Using Pointers to objects
* The this Pointer
* Using new And delete
* More about new And delete
* References – Lvalue & Rvalue
* Passing references to objects
* Returning references
* Independent references and restrictions

# INTRODUCING CLASSES

* Constructor and Destructor Functions
* Constructors that take parameters
* Object Pointers
* Classes, Structures and Unions are related
* In – Line functions
* Automatic In-lining

**Day 2**

# FUNCTION OVERLOADING

* Overloading constructor functions
* Creating and using a copy constructor
* The overload anachronism
* Using default arguments
* Overloading and ambiguity
* Finding the address of an overloaded function

# A CLOSER LOOK AT CLASSES

* Assigning objects
* Passing objects to functions
* Returning objects from functions
* An introduction to friend functions
* Static Class members
* Const Member functions and mutable
* A final look at constructors

# OPERATOR OVERLOADING AND CONVERSION FUNCTIONS

* The basic operator overloading
* Overloading binary operators
* Overloading the relational and logical operators
* Overloading a unary operator
* Using friend operator functions
* A closer look at the assignment operator
* Overloading the [ ] subscript operator
* Creating a conversion function

# Day 3

# INHERITANCE

* Base class access control
* Using protected members
* Constructors, Destructors and Inheritance
* Multiple inheritance
* Virtual base classes

# VIRTUAL FUNCTIONS

* Pointers to Derived classes
* Introduction to virtual functions
* More about virtual functions
* Applying polymorphism

# RUN-TIME TYPE IDENTIFICATION

* Understanding run-time type identification
* Using dynamic\_cast

# NAMESPACES AND THE CASTING OPERATORS

* Namespace
* Using const\_cast, reinterpret\_cast And static\_cast

# TEMPLATES AND EXCEPTION HANDLING

* Generic functions
* Generic classes
* Exception Handling
* More about Exception handling
* Handling exceptions thrown by new

**Day 4**

# STL

* Introduction
* Containers
* Iterator
* Function Pointers
* Generic Functions
* Exercises