Semester: I/II			
Course Title	Introduction to C++ Programming	**CIE Marks**	50
Course Code	BCSPLC105D/205D	**SEE Marks**	50
Course Type	Integrated	**Total Marks**	100
Course Category	PLC	**SEE Duration**	03 hrs
Hours Per Week	2L: 0T: 2P	**SEE Type**	Written
Total Hours	40 hrs	**Credits**	03

Course Learning Objectives:
This course will enable the students to
• Understand the fundamental concepts of object-oriented programming.
• Comprehending how a class can depend on another class and its functions.
• Familiarizing with constructors and destructors.
• Explore the concepts of polymorphism and virtual functions.
• Demonstrate the generic programming functionalities of C++, encompassing exception handling.

Module-1	**(8 hrs)**
Introduction to Object Oriented Programming and C++: Need for C++, Object Oriented Programming Concepts, Classes and objects, methods and messages, abstraction and encapsulation, inheritance, abstract classes, polymorphism. **Beginning with C++:** Structure of C++ program, Simple C++ Program, Creating the source file, Compiling and Linking. **Text Book 1: Chapter 1(1.1 to 1.8) Text Book 2: Chapter 2(2.3, 2.6, 2.7, 2.8)**	
Module-2	**(8 hrs)**
Tokens, Expressions, and Control Structure: Tokens, Keywords, Identifiers and constants, Basic data types, User defined data types, Operators in C++, Scope resolution operator, Expressions and their types, Special assignment expressions. **Functions in C++:** Function prototyping, Call by reference, Return by reference, Inline functions, Default arguments, Function overloading. **Text Book 2: Chapter 3(3.2,3.3,3.4,3.5,3.6,3.14,3.15,3.20,3.21), Chapter 4(4.3,4.4,4.5,4.6,4.7,4.10)**	
Module-3	**(8 hrs)**
Classes and Objects: Specifying a class**,** Defining a member functions, C++ Program with class, Private Member Functions, Static data members, static member functions, Array of Objects, Objects as function arguments, friendly functions. **Constructors and Destructors:** Constructors, Parameterized Constructors, Multiple Constructors in a class, Constructors with Default Arguments, Copy Constructors, Destructors. **Text Book 2: Chapter 5 (5.3,5.4,5.5,5.8,5.9,5.11,5.12,5.13,5.14,5.15) Chapter 6 (6.2,6.3,6.4,6.5,6.7,6.11)**	
Module-4	**(8 hrs)**
Operator overloading and Type Conversions: Defining Operator Overloading, Overloading Unary Operators, Overloading Binary Operators, Overloading Binary Operators using Friend, Rules for Overloading Operators. **Inheritance:** Defining Derived classes, Single Inheritance, Multilevel Inheritance, Multiple Inheritance, Hierarchical Inheritance, Hybrid Inheritance, and Abstract Classes. **Text Book 2: Chapter 7(7.2, 7.3, 7.4, 7.5, 7.8) chapter 8 (8.2, 8.3, 8.5, 8.6, 8.7, 8.8, 8.10)**	

Module-5	(8 hrs)
Pointers, Virtual functions, Polymorphism: this Pointer, Polymorphism, Virtual Functions, Pure Virtual Functions. **Managing Console I/O Operations:** C++ Streams, C++ Streams Classes, Unformatted I/O Operations, formatted Console I/O Operations. **Exception Handling:** Basics of Exception handling, Exception handling Mechanisms, Throwing Mechanisms, Catching Mechanisms. **Textbook 2: Chapter 9(9.4, 9.5, 9.7, 9.8) Chapter 10 (10.2 to 10.5) Chapter 13 (13.2 to 13.5)**	

Practical Components

Sl. No.	Programs for Implementation:
1.	Write a C++ program to find the area of a circle. Take the integer radius value as input.
2.	Write a C++ program to find the sum of all the natural numbers from 1 to N.
3.	Write a C++ program to swap 2 values by writing a function that uses call by reference technique.
4.	Write a C++ program to demonstrate function overloading for the following prototypes. add(int a, int b) add(double a, double b)
5.	Create a class named Shape with a function that prints "This is a shape". Create another class named Polygon inheriting the Shape class with the same function that prints "Polygon is a shape". Create two other classes named Rectangle and Triangle having the same function which prints "Rectangle is a polygon" and "Triangle is a polygon" respectively. Again, make another class named Square having the same function which prints "Square is a rectangle". Now, try calling the function by the object of each of these classes.
6.	Suppose we have three classes: Vehicle, Four_wheeler, and Car. The class Vehicle is the base class, the class Four_wheeler is derived from it and the class Car is derived from the class FourWheeler. Class Vehicle has a method 'vehicle' that prints 'I am a vehicle', class Four_wheeler has a method 'four wheeler' that prints 'I have four wheels', and class Car has a method 'car' that prints 'I am a car'. So, as this is a multi-level inheritance; we can have access to all the other classes' methods from the object of the class Car. We invoke all the methods from a Car object and print the corresponding outputs of the methods. So, if we invoke the methods in this order, car(), fourWheeler(), and vehicle(), then the output will be I am a car I have four wheels I am a vehicle Write a C++ program to demonstrate multilevel inheritance using this.
7.	Write a C++ program to perform unary (increment and Decrement) operator overloading.
8.	Write a C++ program to add two complex numbers using the operator Overloading concept.
9.	Write a function which throws a division by zero exception and catch it in the catch block. Write a C++ program to demonstrate usage of try, catch and throw to handle exceptions.
10.	Write a C++ program function which handles an array of bounds exceptions using C++.

Text Books:	
1.	Bhushan Trivedi, “Programming with ANSI C++”, Oxford Press, Second Edition, 2014.
2.	Balaguruswamy E, Object Oriented Programming with C++, Tata McGraw Hill Education Pvt. Ltd, Fourth Edition,2010.
Reference Books:	
1.	Herbert Schildt, “C++: THE COMPLETE REFERENCE”, McGraw Hill Education, Fourth Edition, July 2017.
2.	Yashwanth Kanetkar, **“Let us C++”,** BPB Publications, July 2019.

Course Outcome Statements: After the successful completion of the course, the student will be able to	
CO1	Understand the fundamental features of object-oriented concepts.
CO2	Illustrate the usage of functions, classes and objects in C++ Programming.
CO3	Analyze the importance of inheritance to develop reusable and extensible code.
CO4	Demonstrate Polymorphism using function and operator overloading.
CO5	Implement C++ program for providing solutions to real world problems by incorporating exception handling.

Course Articulation Matrix

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	2	2	-	-		-	-	-	-	-	-	-
CO2	2	2	-	-	2	-	-	-	-	-	-	-
CO3	2	2	2	-	2	-	-	-	-	-	-	-
CO4	2	2	2	-	2	-	-	-	-	-	-	-
CO5	2	2	2	-	2	-	-	-	-	-	-	-
Avg.	**2**	**2**	**2**	**-**	**2**	**-**	**-**	**-**	**-**	**-**	**-**	**-**

3: High, 2: Medium, 1: Low