25CE102: Object Oriented Programming using C++

w.e.f. Academic Year:	**2025-26**
Semester:	**2**
Category of the Course:	**Engineering Science**
Prerequisite:	Students should have basic knowledge of procedural programming using C, including variables, control structures, functions, and fundamental problem-solving skills.
Rationale:	This course introduces object-oriented programming concepts using C++ to develop modular, reusable, and maintainable software, laying the foundation for advanced programming and real-world application development.

Course Outcomes:

After completion of the course, student will be able to:

	Course Outcome (CO)	RBT Level (Cognitive Domain)
CO1	Describe the key principles of OOP and their advantages over procedural programming.	Understand (U)
CO2	Implement C++ programs using functions, classes, constructors, and operator overloading.	Apply (Ap)
CO3	Demonstrate inheritance and polymorphism concepts for creating hierarchical and reusable programs.	Apply (Ap)
CO4	Develop file-handling programs and use C++ stream classes for data management.	Apply (Ap)
CO5	Evaluate and assess the effectiveness of templates and exception handling mechanisms in developing robust and reusable programs.	Evaluate (E)
CO6	Analyze complex programming scenarios and optimize solutions using OOP principles.	Analyze (An)

Teaching and Evaluation Scheme:

Teaching Scheme					Examination Scheme				
L	**T**	**P**	**C**	**Hrs/Week**	**IE**	**Theory**	**CIA**	**Practical**	**Total Marks**
03	-	02	04	05	40	60	30	20	150

IE: Internal Evaluation Theory: Theory Exam (End Semester)

CIA: Continuous Internal Assessment Practical: Practical Exam (End Semester)

Detailed Syllabus:

Topic		Hrs.	% of Weightage
UNIT: 1	**Concepts of OOP:**	**11**	**25**
Introduction OOP, Procedural Vs. Object Oriented Programming, Principles of OOP, Benefits and applications of OOP. C++ Basics: Overview, Program structure, namespace, identifiers, variables, constants, enum, operators, typecasting, control structures. C++ Functions: Simple functions, Call and Return by reference, Overloading of functions, Inline functions, default arguments, friend functions, and virtual functions.			
UNIT: 2	**Objects and classes:**	**11**	**25**
Basics of object and class in C++, Private, protected and public Members, static data and static function, Constructors and their types, Destructors. Operator Overloading: Overloading unary and binary operators, Operator Overloading with friend function, Data Conversion, type conversion, class to class, basic to class, class to basic			
UNIT: 3	**Inheritance & Polymorphism:**	**11**	**25**
Inheritance: Concept of Inheritance, Types of inheritance, Method overriding, virtual base class, Virtual and Pure virtual function, Late binding, Polymorphism implementation using examples.			
UNIT: 4	**File management, Templates, Exceptions:**	**12**	**25**
File stream, C++ File stream classes, File management functions, File modes, Binary and random files. Function templates, class templates, Introduction to exception, try-catch throw, multiple catch, rethrowing exception, implementing user defined exceptions.			
		45	**100**

List of Practical:

#	Aim	Teaching Hrs.
Lab 1	**Basics of C++ & Control Structures** 1. Write a simple "Hello, World!" program and demonstrate C++ program structure. 2. Implement a program to demonstrate data types, typecasting, and operators. 3. Write a program to check whether a number is prime or not using control structures.	**02**
Lab 2	**Functions in C++** 4. Implement call-by-value and call-by-reference in a function. 5. Write a program to demonstrate function overloading with different parameter lists. 6. Develop a program to implement an inline function for better	**02**

	performance.	
Lab 3	**Friend & Virtual Functions** 7. Write a program to demonstrate friend functions for accessing private members. 8. Implement a virtual function and demonstrate dynamic binding.	**02**
Lab 4	**Default Arguments & Namespaces** 9. Implement a function with default arguments and test different cases. 10. Demonstrate the use of namespace and scope resolution operators in C++.	**02**
Lab 5	**Classes & Static Members** 11. Create a class for a student (name, roll number, marks) and implement member functions. 12. Write a program to demonstrate static data members and static member functions.	**02**
Lab 6	**Constructors & Destructors** 13. Implement a program to demonstrate different types of constructors (default, parameterized, copy). 14. Write a program to show destructor execution in object destruction.	**02**
Lab 7	**Operator Overloading** 15. Overload the + operator to add two complex numbers. 16. Implement overloading of unary operators (++, --) using a member function.	**02**
Lab 8	**Type Conversion** 17. Implement class-to-class type conversion using constructors. 18. Develop a program to convert basic type to class type and vice versa.	**02**
Lab 9	**Inheritance Basics** 19. Implement single and multilevel inheritance with a real-world example. 20. Write a program to demonstrate multiple inheritance using two base classes.	**02**
Lab 10	**Method Overriding & Virtual Functions** 21. Demonstrate method overriding using base and derived class functions. 22. Implement virtual base classes in C++.	**02**
Lab 11	**Late Binding & Pure Virtual Functions** 23. Develop a program to demonstrate late binding using a virtual function. 24. Implement a pure virtual function to create an abstract class.	**02**
Lab 12	**Polymorphism** 25. Implement runtime polymorphism using virtual functions. 26. Create a program that demonstrates compile-time polymorphism using function overloading.	**02**

Lab 13	**File Handling Basics** 27. Write a C++ program to read and write data to a file. 28. Implement file modes (read, write, append, binary mode).	**02**
Lab 14	**Binary Files & Random Access** 29. Write a program to demonstrate binary file handling (read & write objects). 30. Implement a random-access file operation using seekg() and seekp().	**02**
Lab 15	**Templates in C++** 31. Implement a function template for finding the maximum of two numbers. 32. Write a class template for a stack and perform push/pop operations.	**02**
	Total Hours	**30**

TEXT/REFERENCE BOOKS:

Text Books:

1. Balagurusamy, E. (2020). Object-Oriented Programming with C++ (8th ed.). McGraw Hill.
2. Lafore, R. (2001). Object-Oriented Programming in C++ (4th ed.). Sams Publishing.
3. Kanetkar, Y. P. (2019). Let Us C++. BPB Publications.

Reference Books:

1. Schildt, H. (2002). C++: The Complete Reference (4th ed.). McGraw-Hill.
2. Holzner, S. (2002). C++ Black Book: A Comprehensive Guide to C++ Mastery. Paraglyph Press.

LIST OF OPEN-SOURCE SOFTWARE/LEARNING WEBSITE:

- Open-source software dev C++
- www.nptel.ac.in
- www.learncpp.com

Course Outcomes Mapping:

CO	Course Outcome (CO) Description	POs/ PSOs Mapping	Cognitive Level (RBT)	Knowledge Category	Lectures (Hours)	Lab (Hours)
CO1	Describe the key principles of OOP and their advantages over procedural programming	PO1, PO2, PO4, PSO1	Understand	Conceptual, Factual	8	2
CO2	Implement C++ programs using	PO1, PO2,	Apply	Conceptual, Procedural	8	10

	functions, classes, constructors, and operator overloading	PO3, PO4, PSO1				
CO3	Demonstrate inheritance and polymorphism concepts for creating hierarchical and reusable programs	PO1, PO3, PO5, PSO1	Apply, Analyze	Conceptual, Procedural	8	8
CO4	Develop file-handling programs and use C++ stream classes for data management	PO1, PO4, PSO1	Apply	Conceptual, Procedural	5	4
CO5	Evaluate and assess the effectiveness of templates and exception handling mechanisms in developing robust and reusable programs	PO1, PO2, PO3, PO5, PSO2	Apply, Evaluate, Create	Conceptual, Procedural, Metacognitive	8	4
CO6	Analyze complex programming scenarios and optimize solutions using OOP principles	PO1, PO3, PO5, PO10, PSO1, PSO2	Analyze, Evaluate	Procedural, Metacognitive	8	2

Mapping of COs with POs & PSOs:

COs	**POs**												**PSOs**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO1	2	2	1	1	1	–	–	–	–	1	–	2	2	1
CO2	3	2	3	2	3	–	–	–	1	1	–	2	3	2
CO3	3	2	3	2	2	–	–	–	2	1	–	2	3	2
CO4	2	2	2	3	3	–	–	–	–	–	–	1	3	1
CO5	2	3	2	2	3	–	–	–	1	–	–	3	3	3
CO6	3	3	3	3	2	–	–	–	2	2	–	3	3	3

3: High, 2: Medium, 1: Low