Course No.	Course Name	L-T-P - Credits	Year of Introduction
IE202	OBJECT ORIENTED PROGRAMMING & NUMERICAL METHODS	3-0-0-3	2016

Prerequisite : Nil

Course Objectives

- To bring together mathematical modeling and modern programming methodology.
- To provide an overview of OOP paradigm, its' methodology and its' applications in the field of software design for computer-based numerical analysis for natural science problems.
- To draw attention to polymorphism and typing, parallelism, stability and other significant concepts and features of OOP methodology.
- To understand methods, techniques; tricks and receipts for practical programming and program coding in C++.

Syllabus

Programming methodology, Concept of Procedure oriented and Object oriented Programming. Classes in C++. Concepts of Data Abstraction, Data binding, Polymorphism, Data Encapsulation. Class initialization techniques. Dynamic memory control. Inheritance principles in Object Oriented languages. Importance of Abstract classes. Virtual Inheritance. Exception handling systems . Templates in Object Oriented languages. Numerical methods to solve the Differential equation. Numerical solution to Interpolation and integration

Expected outcome.

After completing the course, the students are expected to be able to:

(i) compare the programming languages and their facilities for object-oriented programming;

(ii) define a set of abstract concepts as a knowledge domain environment;

(iii) design a hierarchical set of data types (classes) on the base of inheritance;

(iv) make out the classes hierarchy on the base of inclusion relations and on the base of private inheritance;

(v) choose and implement a suitable inheritance scheme: behavior and realization; only realization; only behavior;

(vi) understand and utilize dynamic dispatch (dynamic binding or late binding or run-time linking) and static calls (fixed implementation or name binding or early binding or compilation-time linking) and utilize both in computer program code;

(vii) understand the role of polymorphism and design polymorphic computer program code;

Text Book:

Booch, Grady (1997). Object-Oriented Analysis and Design with Applications.

Numerical Methods for Scientific and Engineering Computation by M.K.Jain, S.R.K.Iyengar, R.K.Jain, 4th Edition

S. B. Lippman, Josee Lajoie, Barbara E. Moo, “C++ Primer”, Fourth Edition,
Pearson Education, 2005.

References:

1. Stroustrup, Bjarne (1997). The C++ Programming Language (Third ed.).
2. Sutter, Herb (2004). Exceptional C++ Style. Addison-Wesley.
3. Becker, Pete (2006). The C++ Standard Library Extensions: A Tutorial and Reference.

Course Plan

Module	Contents	Hours	Sem. Exam Marks
I	Role of programming methodology, programming technology and programming languages facilities. Comparison the concepts of procedure-oriented programming (POP) and object-oriented programming (OOP). Fundamental concepts of object-oriented programming OOP): abstraction, encapsulation, modularity, data hierarchy through inheritance, polymorphism and typing, parallelism and stability.	7	15%
II	Classes in C++. Encapsulation and modularity – methods and properties in classes. Example – class "String". Polymorphism in C++. Five ways to implement polymorphic code. The functions overloading and default values for functions' parameters in C++	6	15%
FIRST INTERNAL EXAMINATION			
III	Interface as a data type definition. Operators overloading in C++. Class methods and class friends in C++. Inline functions. Pointer and reference data types in C++. Constants in C++: objects, pointers, references, methods. Techniques; tricks and receipts for practical programming and program coding in C++ with pointers, references and constants. Concept of encapsulation and its implementation in C++.	7	15%
IV	Initialization and assignment for objects. Constructors and destructor in C++ classes environmental. Type conversion (type corection and type casting) rules in C++. How to control typecasting by constructors and typecasting operation. Typecasting for variables, pointers and references	6	15%
SECOND INTERNAL EXAMINATION			
V	Heap-based (dynamic) memory control in C++ classes. Techniques; tricks and receipts for practical programming and program coding in C++ with heap-based memory control. Classes' hierarchy: inclusion relations and inheritance. Inheritance implementation for polymorphism and for strong typing. Dynamic (late) binding and name (early) binding. Three inheritance schemes. Abstract classes and abstract base classes in C++. Multiple inheritances in C++. Virtual inheritance and virtual base classes. Techniques; tricks and receipts for practical programming and program coding in C++ with multiple inheritances. Exception handling system in C++.	10	20%

	Techniques; tricks and receipts for practical programming and program coding in C++ with exception handling implementation. Templates in C++		
VI	*Errors:* Concepts, types of errors Finding roots of an equation by Bisection algorithm, Regular-falsi method, Secant and Newton-Raphson method, Problems and its graphical significances. *Solution of Differential Equation:* Euler Method, Taylor Method, Runge-Kutta second and fourth order method for solving differential equations. *Interpolation:* Newton Forward and Backward interpolation, Lagrange interpolation *Integration:* Mathematical Foundation for Trapezoidal and Simpson's 1/3rd Rules and its Composite forms	6	20%
END SEMESTER EXAM			

Question Paper Pattern

Total marks: 100, Time: 3 hrs

The question paper should consist of three parts

Part A

4 questions uniformly covering modules I and II. Each question carries 10 marks
Students will have to answer any three questions out of 4 (3X10 marks =30 marks)

Part B

4 questions uniformly covering modules III and IV. Each question carries 10 marks
Students will have to answer any three questions out of 4 (3X10 marks =30 marks)

Part C

6 questions uniformly covering modules V and VI. Each question carries 10 marks
Students will have to answer any four questions out of 6 (4X10 marks =40 marks)

Note: In all parts, each question can have a maximum of four sub questions, if needed.