Program	BS Data Science	
Course Code	CC-211N	
Course Title	Object Oriented Programming	
Credit Hours	**Theory**	**Lab**
	3	1
Lecture Duration	90 minutes (1.5 Hours), 2 lectures per week, 3 hours lab session per week	
Semester	2	
Pre-requisites	**Courses**	**Knowledge**
	Programming Fundamentals	Students should know how to program in C++, Structural programming in C++.
Follow Up Courses	Data Structures	

Course Learning Outcomes (CLOs)

CLO No	**Course Learning Outcome**	**Bloom Taxonomy**
CLO-1	Understand principles of object-oriented paradigm.	C2 (Understand)
CLO-2	Identify the objects & their relationships to build object-oriented solution.	C3 (Identify)
CLO-3	Model a solution for a given problem using object-oriented principles.	C3 (Apply)
CLO-4	Examine an object-oriented solution.	C4 (Examine)

Objectives	1. To equip the learner with the philosophy and necessary skills to formulate solutions of real world problems using object oriented paradigm. 2. Justify the philosophy of object-oriented design and the concepts of encapsulation, abstraction, inheritance, and polymorphism.
	3. Strong concepts of object manipulation and dynamic memory allocation within classes

Learning Outcomes	• Students can formulate solutions of real world problems using object oriented paradigm. • Students should be able to translate a real world problem to object oriented model. • Student can familiar with encapsulation, abstraction, inheritance, and polymorphism concepts.
Syllabus	Modular vs. Object-Oriented Paradigm, Abstraction, Encapsulation, Information hiding; Classes and Objects with C++: Data members, Member functions, Public/private access, Constructors, Destructors, Overloaded constructors, Constant member functions, Arrays of objects (both static and dynamic), Objects as arguments, Returning objects from functions, Copy constructor, Pointers as member variables, Shallow copy vs. Deep copy, Destructor, this pointer, Constant member variables, Constant objects, Static member variables, Static member functions; Operator Overloading: Simple binary operators, Overloading assignment operator for classes with dynamic memory allocation, Overloading logical and unary operators, Friend functions, Overloading operators as friend functions, Overloading stream insertion and extraction operators, Some other operators; I/O and File Processing: Text filing, Binary filing; Aggregation and Composition: Classes within classes, UML, Constructor, destructor calling sequence; Inheritance: Basics, Examples, UML, Public inheritance, Protected access specifier, Public inheritance vs. private and protected inheritance, Multiple inheritance, Diamond problem, Virtual inheritance, Writing copy constructors and overloading assignment operator for derived classes; Inheritance and Polymorphism: Virtual functions, Static vs. dynamic binding, Pure virtual functions, Abstract classes, Examples;
	Templates: Function templates, Class templates; Exceptions and Exception handling; Recursion: Basics, Examples

Contents	1. Introduction to Object Oriented Concepts 1.1. Real world examples Define the keyword 'class' 2.1. Access modifiers 2.2. Setter/Mutator and Getter/Accessor methods 2.3. Constructor & Destructor 3. Pointer/Reference to objects Preventing changes in data members from a method 4.1. Constant method 4.2. Constant data members Static functions Static data members constant and static objects 8. Calling sequence of Constructor & Destructor for constant and static objects Composition/Aggregation Cont... Nameless objects 11. Array of objects; 12. Operator Overloading 13. Friend functions Inheritance 14.1. Multilevel Inheritance 14.2. Private Inheritance 15. Polymorphism 15.1. Pure virtual functions and abstract class Diamond inheritance 17. Virtual inheritance
	18. Template 18.1. Templated Function 18.2. Templated Class 19. Exception handling

Teaching-learning Strategies	• Interactive class session • Hands on practices in class • Brainstorming and Group discussion sessions • Coding in LABS
Assignments	Coding Assignments 5
Textbooks	• **A.** Deitel, H. M., & Deitel, P. J. (2010). C++ How to Program 6 th Edition. Prentice Hall. • **B.** Gaddis, T., & Sengupta, P. (2012). Starting Out with C++: From Control Structures Through Objects. Pearson.
Reference Material/Suggested Readings	• **R1.** Handouts. • **R2.** Shtern, V. (2000). Core C++: A software engineering Approach. Prentice Hall., • **R3.** Prata, S. (2002). C++ primer plus. Sams Publishing**.** • **R4.** Stroustrup, B. (2013). The C++ Programming Language