NIRMA UNIVERSITY

Institute:	Institute of Technology, School of Technology
Name of Programme:	BTech CSE, Integrated BTech (CSE)-MBA
Course Code:	2CS511CC25
Course Title:	Java Programming
Course Type:	Core
Year of Introduction:	2025-26

L	T	Practical Component				C
		LPW	PW	W	S	
2	0	2	-	-	-	3

Course Learning Outcomes (CLO):

At the end of the course, the students will be able to –

1. explain procedural and object-oriented paradigms and principles (BL2)
2. relate the concepts of object-oriented design with principles of object-oriented programming (BL2)
3. apply exception handling, input-output operations, and multi-threading concepts for application development (BL3)
4. develop various applications using object-oriented concepts. (BL6)

Unit	Contents	Teaching Hours (Total 30)
Unit-I	**Introduction to Object-Oriented Concepts and Java**: Basic concepts of OOP, Features of Java, byte code, Java Development Kit (JDK), Java Virtual Machine (JVM), Introduction to three OOP principles (Inheritance, Polymorphism, Encapsulation), brief overview of classes, Java program compilation and execution	03
Unit-II	**Data Types, Variables, and Operators in Java:** The primitive types, Literals, Type Casting, Arrays: One-dimensional array, multi-dimensional array, alternative array declaration statements. Control Statements: Selection statements (i.e., if, switch, etc.), iteration statements (i.e., while, do while, the for-each version of the for Loop, Nested Loops, etc.), jump statements (i.e., break, continue)	06
Unit-III	**Classes and Methods:** class fundamentals, objects, assigning object reference variables, methods in a class, constructors, this keyword, garbage collection, finalize () method, overloading methods, argument passing, access control, static, final, nested, and inner classes, command line arguments, variable-length arguments. String Handling: Basics of String handling in Java, String class methods, Brief of StringBuffer and StringBuilder Classes, Inheritances: Basics, member access and inheritance, super class references, using super, multilevel hierarchy, constructor call sequence, method overriding, dynamic method dispatch, abstract classes, Object class. Packages and Interfaces: defining a package, finding packages and CLASSPATH, access protection, importing packages, interfaces (defining, implementation, nesting, applying), variables in interfaces, extending interfaces	12
Unit-IV	**Exception Handling**: fundamental, exception types, uncaught exceptions, try, catch, throw, throws, finally, multiple catch clauses,	04

	built-in exceptions, custom exceptions	
Unit-V	**Multithreading and I/O:** Multithreaded Programming: Java thread model, thread priorities, Thread class, Runnable interfaces, creating a thread(s), Thread class methods. Managing I/O: Basic of I/O, File handling.	05

Self-Study:

The self-study contents will be declared at the commencement of the semester. Around 10% of the questions will be asked from self-study content.

Suggested Readings/ References:

1. Herbert Schildt, Java – The Complete Reference, McGraw Hill
2. E. Balaguruswamy, Programming with Java – A primer, McGraw Hill
3. David Flanagan, Student Workbook Java in a Nutshell, O'Reilly
4. Cay S. Horstmann, Core Java (TM)-Fundamentals, Prentice Hall.

Suggested List of Experiments:

Sr. No.	**Name of Experiments/Exercises**	**Hours**
1	**(Preliminaries and Basic Java Programming)** a) Hands-on practice on C-Programming: i. Write a program in C to display and count duplicate elements in an array ii. Write a program in C to count the frequency of each element of an array b) Write a Java program to display a greeting message like: "First Java Program..." on the console c) Write a Java program to display all primitive type variables and display your name on the last line	04
2	**(Operators)** a) Write a Java Program that checks whether a user-entered number is a special number or not b) Write a Java program to demonstrate automatic and explicit type casting c) Write a Java program to: i. Check whether a number is odd or even using an if-else statement ii. Check the category of a given character using the if-else-if ladder iii. Check whether a number is prime using a for loop iv. Display the reverse of a number and check whether it is a palindrome using while/do-while loops. v. Print the pattern using nested loops	03
3	**(Operators and Array)** a) Design a calculator with arithmetic and bitwise operators. The user must scan the operand (s) and operator. b) Given an array of positive and negative numbers, find if there is a subarray with a sum of 0	03
4	**(Class and Objects)** a) Create a class called Complex to perform arithmetic operations with complex numbers. b) Implement addition, subtraction, and display methods	03

5	**(String Handling)** a) Write a Java Program to ask a user to enter a paragraph and perform operations like sentence and word counting, character frequency, and word search b) Implement a program that arranges the words of a string in descending order of length and alphabetically for words of the same length. Store the output in a text file	04
6	**(Inheritance)** a) Create a class CovidParameters inheriting from a Patient class. Implement constructors and member variables like CTScore, D-dimer, and platelet count	03
7	**(Abstract Class)** a) Create an abstract class Instrument with an abstract play method. Create Piano, Flute, and Guitar subclasses, overriding the play method. Demonstrate polymorphism	02
8	**(Package and Interface)** a) Create an interface Polygon with methods for calculating area, perimeter, and displaying values. Implement Square and Rectangle classes	02
9	**(Exception Handling)** a) Create a MathFunctions class with methods for calculating mean and division. Handle custom exceptions for invalid numbers and division by zero. b) Create a BankAccount class with methods for deposit, withdrawal, and balance checking. Handle custom exceptions for negative amounts, insufficient funds, and low balances	03
10	**(Multithreading and I/O)** a) Write a Java program that creates two threads to print odd and even numbers sequentially. Synchronize the threads if needed b) Write a stream-based program to accept and validate user details, handle exceptions, and save data to a file. Display saved records.	03