Programme: B. Tech. (CSE)	Course Title: Advance Programming			Course Code: CSE226
Type of Course: Program Core	Prerequisites: Computer Programming (CP), Data Structures and Algorithms (DSA)			Total Contact Hours: 42 (Theory)+22 (Lab)
Year/Semester: 2/Odd	Lecture Hrs/Week: 3	Tutorial Hrs/Week: 0	Practical Hrs/Week: 2	Credits: 4

Learning Objective:

This course teaches and trains object-oriented programming skills necessary for software development using Java programming language as a vehicle. The students learn the fundamental concepts of object-oriented software engineering and development, such as Objects and classes, abstraction, inheritance, and polymorphism. This course offers a balanced treatment of OOP theory and practices for developing secure and robust codes (using exception handling) with debugging strategies. Advanced topics like Graphic-User Interface, Multithreading, and JDBC are also covered.

Course outcomes (COs):

On completion of this course, the students will have the ability to:		Bloom's Level
CO-1	**Model** and **solve** problems using object-oriented modelling and programming constructs (UML, Classes, Inheritance, Polymorphism, Interfaces)	3
CO-2	**Demonstrate** advanced programming skills by employing generics, error and exception handling and testing.	3
CO-3	**Understand** the basics of multi-threaded programs	2
CO-4	**Demonstrate** skills to construct GUI based program interfaces.	3
CO-5	**Analyze**, **design**, and **implement** software solution for a given problem using Object Oriented Programming language.	3
CO-6	**Understand** a method for data exchange between a program and a permanent database	2

Course Topics	Lecture/Lab Hours	
Unit I: A. Software Development Lifecycle		
Waterfall Model of Software Development 1. Software product stakeholders and product developers. 2. System development Requirements 3. Feasibility Study 4. System Design (Low Level and High Level) 5. Coding and Integration (Top-down and Bottom-up approach)	3	3

6. Debugging and Testing 7. Implementation and post-implementation reviews Discuss a case study to explain the software development process and the benefits of systematically following a developmental process.		
B. Programming Paradigms		
1. Functional Programming 2. Object-Oriented Programming	1	1
C. Principles of Object-Oriented Programming and problem solving		
1. Unified Modeling Language (UML) 2. Use Case Diagrams 3. Classes and Objects. 4. Collaborations and Interactions: Links and Associations. 5. Abstraction and Data hiding 6. Class hierarchy (Inheritance): is-a, generalization-specialization, Liskov substitution Principle 7. Polymorphism (Function binding and overriding)	5	5
Unit-II A. OOP Language Basics		
Review: Data types, variables, scope and lifetime of variables, operators, and expressions.	1	5
Control statements, type conversion and casting, simple java program	1	
Introduce basics concepts of classes, objects, constructors, methods, access control	1	
this keyword, message passing, parameter passing, Java input-output and string handling, garbage collection,	2	
B. Inheritance		
Inheritance Basics (types and relationships), Member access specifiers, Member access and Inheritance (with a practical example), Superclass Variable (reference a subclass object), Use of super keyword (call a member of super class) (using super to call super-class constructors),	2	5
Multilevel Hierarchy construction, Method overriding, run time polymorphism, Application of Method overriding	1	
Abstract classes, Final keyword, and its uses. Use of static and non-static initializer blocks with inheritance concept	1	
Case study to explain inheritance and polymorphism	1	
Unit- III A. Package and Interfaces		
Package Basics (requirements and defining a package (example)), Access protection with example, import packages.	1	4
Interfaces, basics (requirements and defining/implementing an interface (example)), Solution to multiple inheritance.	1	
Accessing implementations via interface references, partial implementations	1	

Nested interfaces, variables in interfaces, and extending interfaces	1	
B. Exception handling (Secure and Robust coding)		
Concepts of exception handling, benefits of exception handling, exception hierarchy.	1	4
Usage of try, catch, throw, throws and finally, built in exceptions	1	
User defined exception classes, nested try	1	
A case study to explain the Exception handling Concepts	1	
Unit –IV A. Multithreading		
Java Threads: multi-threading and multitasking, thread life cycle.	**1**	4
Creating threads, daemon threads.	**1**	
Thread synchronization, threads Priority, thread termination	**1**	
Example of a programs with threads.	**1**	
Unit –IV B. Graphic User Interface Programming		
Programming model for GUI based programs	**1**	4
Java Swing components and their geometric and other properties	**1**	
Writing call back functions	**1**	
A case study example	**1**	
Unit –IV C. Database Connectivity		
Introduction to JDBC, creating connection, registering DBMS driver.	**2**	4
Types of JDBC drivers, executing queries.	**2**	
Unit –IV D. Design Patterns		
Advantages of design patterns and discuss some simple design patterns.	**3**	3

Lab Experiments and Exercises

0	Importing and installing IDE and UML tools on students' personal computers
1	Using UML tools to understand and analyze problem statements
2	Java programming: Input-Output, Files, Primitive and Object types
3	Java programming: Simple class, Data and Method members, Abstraction.
4	Java Programming: Class hierarchy, Introduction to Java platform package classes
5	Java Programming: Implementing 1-1, 0-* associations. Encapsulation: Interfaces and Packages.
6	Java programming: Polymorphism, run-time determination of method invocation.
7	Java programming: Robust and fault tolerant programs. Exception handling.
8	Concurrent programming, Java threads, Synchronization
9	Graphic User Interfaces and reactive programs
10	Interfacing programs with databases.

Textbook References:

Textbook:

1. Herbert Schildt, The Complete Reference JAVA2, Fifth Edition, McGraw-Hill.

Reference books:

1. B. Bruegge and A.H. Dutoit, Object-Oriented Software Engineering using UML, Patterns, and Java, Pearsons, 2010

2. Timothy C Lethbridge and Robert Laganiere, Object-Oriented Software Engineering, TATA McGRAW-Hill Edition.

Additional Resources:
Class notes and lecture recordings when available.

Evaluation Method	
Item	**Weightage (%)**
Theory	
Quiz 1 (after 10 lessons)	10
Midterm	25
Quiz 2 (after 30 lessons)	10
Final Examination	30
Lab	
Weekly lab works	10
3 MCQ	3*5

CO and PO Correlation Matrix

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	3	2	3	3	3	-	-	3	2	3	1	2	3	3	-
CO2	3	2	3	3	3	-	-	3	2	3	1	2	3	3	-
CO3	2	-	-	-	3	-	-	3	2	-	-	-	3	3	-
CO4	3	-	-	-	3	3	-	3	2	-	-	-	3	3	-
CO5	3	2	3	3	3	-	-	3	2	3	2	-	3	3	-
CO6	2	2	3	3	3	-	-	3	3	3	3	3	3	3	-

Last Updated on: 15th August 2021
Updated by: Vishv Malhotra, Nirmal Kumar S, Varun K Sharma
Approved by: