Year: B. Tech III (Semester V)

Subject Name: Object Oriented Modelling and Programming **Subject Code:** BTCO15503
Type of course: Open Elective -1
Prerequisite (if any): Working experience of any one programming language like C

Rationale: Object oriented programming has become a fundamental part of software development. Object oriented programming facilitates Reuse of code, flexibility, and effective problem solving. It provides a modular structure for programs and implementation details are hidden. Reuse of code lowers the cost of development. Open-source object oriented programming languages such as Java plays an important role in developing highly efficient, scalable and secure software.

Teaching and Examination Scheme:

Teaching Scheme				Theory Marks			Practical Marks		Total
L	T	P	C	TEE	CA1	CA2	TEP	CA3	
3	0	2	4	60	25	15	30	20	150

CA1: Continuous Assessment (assignments/projects/open book tests/closed book tests CA2: Sincerity in attending classes/class tests/ timely submissions of assignments/self-learning attitude/solving advanced problems TEE: Term End Examination TEP: Term End Practical Exam (Performance and viva on practical skills learned in course) CA3: Regular submission of Lab work/Quality of work submitted/Active participation in lab sessions/viva on practical skills learned in course

Content:

Sr. No.	Content	Total Hrs
1	**Class Modeling using UML** Object and class concepts, link and association, Generalization and Inheritance, Advanced Object and class concepts, Association Ends, N-ary associations, aggregation, abstract classes, multiple inheritance, Metadata, Constraints, Derived data, Packages.	4
2	**Basics of Java Programming :** Features of Java, Concepts of JDK, bytecode, JVM and platform independency, Creating, compiling and executing a simple java program, variables, constants, naming conventions, data types, operators, operator precedence and associativity, expressions evaluation, type conversion, reading input from console, Flow control in Java.	4
3	**Class Fundamentals :** General form of class, creating class and object, overloading methods, constructor, constructor overloading, passing and returning object to and from methods, assigning object reference variables, introducing access control, understanding static, final, this keyword, Garbage collection and the finalize () method, wrapper classes	5

4	**Array and String Handling :** Array basics, String class, Array of Strings, StringBuffer and StringBuilder class, String Tokenizer Class, Command line arguments	4
5	**Inheritance, Interfaces and Packages** **Inheritance :** Concept of Inheritance, super class, sub class, Inheriting data members and methods, access modifiers in inheritance, multilevel inheritance, constructors in inheritance, method overriding, dynamic method dispatch and runtime polymorphism, instanceof operator, super keyword, using final with inheritance, Object class - super class of all the classes, abstract classes **Interfaces :** Defining Interface, Default Methods, Implementing Interface, Variables in Interface **Package :** Concepts of Package, creating own package, CLASSPATH, Importing package-using import statement, access modifiers	10
6	**Exceptions Handling :** Exception handling fundamentals, Exception and Error, Use of try, catch, throw, throws and finally, Built-in exceptions, Creating custom exceptions	5
7	**Multithreading in Java :** Concepts of Multithreading, Main thread, Life cycle of thread, Thread class and Runnable interface, Creating thread by extending Thread class, creating thread by implementing Runnable interface, Creating multiple threads, Thread priorities, Thread synchronization, Thread communication, Deadlock	7
8	**Input/output Programming : I/O :** File Class, Introduction to Stream, Byte Stream, Character stream, Readers and Writers, File InputStream, File Output Stream, InputStreamReader, OutputStreamWriter, FileReader, FileWriter, Buffered Reader	6

Suggested Specification table with Marks (Theory): (For B.Tech only)

Distribution of Theory Marks					
R Level	**U Level**	**A Level**	**N Level**	**E Level**	**C Level**
15	15	15	05	05	05

Legends: R: Remembrance; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create and above Levels (Revised Bloom's Taxonomy)

Note: This specification table shall be treated as a general guideline for students and teachers. The actual distribution of marks in the question paper may vary slightly from above table.

Reference Books:

Sr No	Title of book /article	Author(s)	Publisher and details like ISBN	Year of publication / Publication Edition

1	Java : The Complete Reference	Herbert Schild	Oracle Press	
2	Java Fundamentals - A comprehensive introduction	Herbert Schildt, Dale Skrien	McGraw Hill Education	
3	Programming with Java A Primer	E. Balaguruswamy	McGraw Hill Education	
4	Core Java Volume-I Fundamentals	Cay S. Horstmann, Gray Cornell	Pearson Education	
5	Introduction to Java Programming Comprehensive Version	Y. Daniel Liang	Pearson	
6	Object Oriented Modeling and Design with UML –second edition	Michael Blaha and James Rambaugh	Pearson	

Course Outcomes:

Sr. No.	CO statement	Marks % weightage
CO-1	Describe and design object oriented programming concepts in Java.	25%
CO-2	Develop applications using Java building blocks like inheritance, package, collection and interfaces.	25%
CO-3	Solve concurrency and reliability issues using multithreading and exception handling respectively.	25%
CO-4	Implement I/O based applications considering byte as well as character streams.	25%

List of Open learning website:

1. https://www.tutorialspoint.com/java/index.htm
2. https://www.w3schools.com/java/
3. https://www.javatpoint.com/java-tutorial
4. https://docs.oracle.com/javase/tutorial/
5. https://www.programiz.com/java-programming
6. https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm

List of Open Source Software:

1. Eclipse
2. Netbeans IDE
3. IntelliJ IDEA
4. VSCodium

List of Experiments:

Sr.No	Practical
1	Write a program which takes two numbers and an operator from the user and performs mathematical operation on entered two numbers.
2	Write a program that sorts the integer numbers taken from the user as an input.
3	Write a program to validate the entered password. The password must have one capital letter, one digit and one special character from {$, #, %} set. The length of password must be at least 8 characters. The program should display "Password is Valid" If the password satisfies above criteria, otherwise it should display "Password is Invalid".
4	Write a program to find the factorial of a given number. Take the number through the command line argument.
5	Design a class named Triangle to represent a Triangle. The class contains: • Three double data fields named a, b and c specifying three sides of the triangle. The default values are 1. • A no-arg constructor that creates a default triangle. • A constructor that creates a triangle with the specified values. • A method named getArea() that returns the area of this triangle. • A method named getPerimeter() that returns the perimeter.
6	Design a class named StopWatch. The class contains: • Private data fields startTime and endTime with getter methods. • A no-arg constructor that initializes startTime with the current time. • A method named start() that resets the startTime to the current time. • A method named stop() that sets the endTime to the current time. • A method named getElapsedTime() that returns the elapsed time for the stopwatch in milliseconds.
7	Create a class called Time containing followings: • Two private data fields Hour and Minute • No-argument constructor and parameterized constructor • A method getTime() that asks the user to enter the values of both fields

	• A method ahead() which displays which object is ahead from other • A method add() that performs addition of two objects and returns a third object • A method showTime() to display time object
8	Write a simple java application that defines a class Complex with real(int) and img(int) as data fields, no-argument constructor and parameterized constructor. The class should have overloaded methods to perform addition of two Complex numbers by passing objects as arguments. Demonstrate this keyword in a parameterized constructor.
9	Write a simple java application that defines a class Student with roll_no(int), name(String), address(String) & branch(String) as data fields. The class should have getData() & showData() methods. The program should create an array of Student objects, get the details and display it. Create branchDisplay(student[] s) static method to display all objects having computer branch.
10	Demonstrate the static data field and static method by creating an appropriate class. Demonstrate this class in the main method by creating different objects.
11	Write a simple java application that creates a Player class. Inherit CricketPlayer class from Player class. Inherit Batsman & Bowler classes from CricketPlayer class. Assume suitable data and member methods.
12	Write a simple java application that creates a Shape class with two double data members. The class should have an area method to calculate the area of shape. Inherit two classes Rectangle and Triangle from Shape class. Demonstrate method overriding & super keyword.
13	Write a simple java application that creates a Shape class with two double data members. The class should have one abstract method area to calculate the area of shape. Inherit two classes Rectangle and Triangle from Shape class. Demonstrate runtime polymorphism.
14	Write a simple java application that creates an interface Shape. The interface declares read() and show() methods and PI as constant data member. Create classes Rectangle and Circle that implement a Shape interface. Assume suitable data and member methods.
15	Create two packages, CE_Dept & IT_Dept with Machine_Detail_CE and Machine_Detail_IT classes respectively. The class should have a method to display machine information (No_of_PC(int), configuration(String)) for both departments. Write a java application that imports both defined packages and call their methods.

16	Write a simple java application that reads marks of five subjects through command line arguments and displays the average. The application should generate an exception if marks are not in integer format and out of 0-100.
17	Write a simple java application that declares Employee class. The program should generate and handle custom exceptions such as a. InvalidEmailAddressException if the address does not contain . and @ b. InvalidTelephoneNumberException if total no of digits > 10.
18	Write a simple java application that creates two threads: One thread creates even numbers and another thread creates odd numbers.
19	Implement producer consumer IPC problem using multi-threading.

Major Equipment Needed:

- Latest PCs with related software.