

## DEPARTMENT OF COMPUTER SCIENCE & DESIGN

### COURSE MODULE: OBJECT ORIENTED PROGRAMMING WITH JAVA

Course Coordinator: Yeshashwini Bhandari K R				Academic Year:2024-25	
Department: COMPUTER SCIENCE & DESIGN					
Course Code	Course Title	Core/Elective	Prerequisite	Contact Hours L:T: P:S	Total Hrs/ Sessions
BCS306A	OBJECT ORIENTED PROGRAMMING WITH JAVA	Elective	-	2:0:2	48
<b>Course Learning Objective:</b> The course will enable the students to: <div><div>1.</div><div>To learn primitive constructs JAVA programming language.</div></div> <div><div>2.</div><div>To understand Object Oriented Programming Features of JAVA.</div></div> <div><div>3.</div><div>To gain knowledge on: packages, multithreaded programming and exceptions.</div></div>					
<b>Teaching-Learning Process (General Instruction):</b> These are sample Strategies, which teachers can use to accelerate the attainment of the various course outcomes and make Teaching –Learning more effective <div><div>1.</div><div>Use Online Java Compiler IDE: <a href="https://www.jdoodle.com/online-java-compiler/">https://www.jdoodle.com/online-java-compiler/</a> or any other.</div></div> <div><div>2.</div><div>Demonstration of programing examples.</div></div> <div><div>3.</div><div>Chalk and board, power point presentations</div></div> <div><div>4.</div><div>Online material (Tutorials) and video lectures.</div></div>					
MODULE-1					
An Overview of Java: Object-Oriented Programming (Two Paradigms, Abstraction, The Three OOP Principles), Using Blocks of Code, Lexical Issues (Whitespace, Identifiers, Literals, Comments, Separators, The Java Keywords). Data Types, Variables, and Arrays: The Primitive Types (Integers, Floating-Point Types, Characters, Booleans), Variables, Type Conversion and Casting, Automatic Type Promotion in Expressions, Arrays, Introducing Type Inference with Local Variables. Operators: Arithmetic Operators, Relational Operators, Boolean Logical Operators, The Assignment Operator, The ? Operator, Operator Precedence, Using Parentheses. Control Statements: Java’s Selection Statements (if, The Traditional switch), Iteration Statements (while, do-while, for, The For-Each Version of the for Loop, Local Variable Type Inference in a for Loop, Nested Loops), Jump Statements (Using break, Using continue, return). Chapter 2, 3, 4, 5					
MODULE-2					
Introducing Classes: Class Fundamentals, Declaring Objects, Assigning Object Reference Variables, Introducing Methods, Constructors, The this Keyword, Garbage Collection. Methods and Classes: Overloading Methods, Objects as Parameters, Argument Passing, Returning Objects, Recursion, Access Control, Understanding static, Introducing final, Introducing Nested and Inner Classes. Chapter 6, 7					
MODULE-3					
Inheritance: Inheritance Basics, Using super, Creating a Multilevel Hierarchy, When Constructors Are Executed, Method Overriding, Dynamic Method Dispatch, Using Abstract Classes, Using final with Inheritance, Local Variable Type Inference and Inheritance, The Object Class. Interfaces: Interfaces, Default Interface Methods, Use static Methods in an Interface, Private Interface Methods. Chapter 8, 9					
MODULE-4					
Packages: Packages, Packages and Member Access, Importing Packages. Exceptions: Exception-Handling Fundamentals, Exception Types, Uncaught Exceptions, Using try and catch, Multiple catch Clauses, Nested try Statements, throw, throws, finally, Java’s Built-in Exceptions, Creating Your Own Exception Subclasses, Chained Exceptions. Chapter 9, 10					
MODULE-5					
Multithreaded Programming: The Java Thread Model, The Main Thread, Creating a Thread, Creating Multiple Threads, Using isAlive() and join(), Thread Priorities, Synchronization, Interthread Communication, Suspending, Resuming, and Stopping Threads, Obtaining a Thread’s State. Enumerations, Type Wrappers and Autoboxing: Enumerations (Enumeration Fundamentals, The values() and valueOf() Methods), Type Wrappers (Character, Boolean, The Numeric Type Wrappers), Autoboxing (Autoboxing and Methods, Autoboxing/Unboxing Occurs in Expressions, Autoboxing/Unboxing Boolean and Character Values). Chapter 11, 12					

## DEPARTMENT OF COMPUTER SCIENCE & DESIGN

**Course outcome** (Course Skill Set) At the end of the course, the student will be able to:

- CO 1. Demonstrate proficiency in writing simple programs involving branching and looping structures.
- CO 2. Design a class involving data members and methods for the given scenario.
- CO 3. Apply the concepts of inheritance and interfaces in solving real world problems.
- CO 4. Use the concept of packages and exception handling in solving complex problem
- CO 5. Apply concepts of multithreading, autoboxing and enumerations in program development

### Programming Experiments (Suggested and are not limited to)

1. Develop a JAVA program to add TWO matrices of suitable order N (The value of N should be read from command line arguments).
2. Develop a stack class to hold a maximum of 10 integers with suitable methods. Develop a JAVA main method to illustrate Stack operations.
3. A class called Employee, which models an employee with an ID, name and salary, is designed as shown in the following class diagram. The method raiseSalary (percent) increases the salary by the given percentage. Develop the Employee class and suitable main method for demonstration.
4. A class called MyPoint, which models a 2D point with x and y coordinates, is designed as follows:
  - a. Two instance variables x (int) and y (int).
  - b. A default (or "no-arg") constructor that construct a point at the default location of (0, 0).
  - c. A overloaded constructor that constructs a point with the given x and y coordinates.
  - d. A method setXY() to set both x and y.
  - e. A method getXY() which returns the x and y in a 2-element int array.
  - f. A toString() method that returns a string description of the instance in the format "(x, y)".
  - g. A method called distance(int x, int y) that returns the distance from this point to another point at the given (x, y) coordinates
  - h. An overloaded distance(MyPoint another) that returns the distance from this point to the given MyPoint instance (called another)
  - i. Another overloaded distance() method that returns the distance from this point to the origin (0,0) Develop the code for the class MyPoint. Also develop a JAVA program (called TestMyPoint) to test all the methods defined in the class.
5. Develop a JAVA program to create a class named shape. Create three sub classes namely: circle, triangle and square, each class has two member functions named draw () and erase (). Demonstrate
6. Develop a JAVA program to create an abstract class Shape with abstract methods calculateArea() and calculatePerimeter(). Create subclasses Circle and Triangle that extend the Shape class and implement the respective methods to calculate the area and perimeter of each shape.
7. Develop a JAVA program to create an interface Resizable with methods resizeWidth(int width) and resizeHeight(int height) that allow an object to be resized. Create a class Rectangle that implements the Resizable interface and implements the resize methods
8. Develop a JAVA program to create an outer class with a function display. Create another class inside the outer class named inner with a function called display and call the two functions in the main class.
9. Develop a JAVA program to raise a custom exception (user defined exception) for DivisionByZero using try, catch, throw and finally.
10. Develop a JAVA program to create a package named mypack and import & implement it in a suitable class.
11. Write a program to illustrate creation of threads using runnable class. (start method start each of the newly created thread. Inside the run method there is sleep() for suspend the thread for 500 milliseconds).
12. Develop a program to create a class MyThread in this class a constructor, call the base class constructor, using super and start the thread. The run method of the class starts after this. It can be observed that both main thread and created child thread are executed concurrently.

**Assessment Details (both CIE and SEE)** The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks out of 50) and for the SEE minimum passing mark is 35% of the maximum marks (18 out of 50 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/ course if the student secures a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together.

### CIE for the theory component of the IPCC (maximum marks 50)

- IPCC means practical portion integrated with the theory of the course.
- CIE marks for the theory component are 25 marks and that for the practical component is 25 marks.
- 25 marks for the theory component are split into 15 marks for two Internal Assessment Tests (Two Tests, each of 15 Marks with 01-hour duration, are to be conducted) and 10 marks for other assessment methods mentioned in 22OB4.2. The first test at the end of 40-50% coverage of the syllabus and the second test after covering 85-

## DEPARTMENT OF COMPUTER SCIENCE & DESIGN

90% of the syllabus.

- Scaled-down marks of the sum of two tests and other assessment methods will be CIE marks for the theory component of IPCC (that is for 25 marks).
- The student has to secure 40% of 25 marks to qualify in the CIE of the theory component of IPCC.

### CIE for the practical component of the IPCC

- 15 marks for the conduction of the experiment and preparation of laboratory record, and 10 marks for the test to be conducted after the completion of all the laboratory sessions.
- On completion of every experiment/program in the laboratory, the students shall be evaluated including viva-voce and marks shall be awarded on the same day.
- The CIE marks awarded in the case of the Practical component shall be based on the continuous evaluation of the laboratory report. Each experiment report can be evaluated for 10 marks. Marks of all experiments' write-ups are added and scaled down to 15 marks.
- The laboratory test (duration 02/03 hours) after completion of all the experiments shall be conducted for 50 marks and scaled down to 10 marks.
- Scaled-down marks of write-up evaluations and tests added will be CIE marks for the laboratory component of IPCC for 25 marks.
- The student has to secure 40% of 25 marks to qualify in the CIE of the practical component of the IPCC.

**SEE for IPCC Theory** SEE will be conducted by University as per the scheduled timetable, with common question papers for the course (duration 03 hours)

1. The question paper will have ten questions. Each question is set for 20 marks.
2. There will be 2 questions from each module. Each of the two questions under a module (with a maximum of 3 sub-questions), should have a mix of topics under that module.
3. The students have to answer 5 full questions, selecting one full question from each module.
4. Marks scored by the student shall be proportionally scaled down to 50 Marks

**The theory portion of the IPCC shall be for both CIE and SEE, whereas the practical portion will have a CIE component only. Questions mentioned in the SEE paper may include questions from the practical component.**

### Suggested Learning Resources: Textbook

1. Java: The Complete Reference, Twelfth Edition, by Herbert Schildt, November 2021, McGraw-Hill, ISBN: 9781260463422

### Reference Books

1. Programming with Java, 6th Edition, by E Balagurusamy, Mar-2019, McGraw Hill Education, ISBN: 9789353162337.
2. Thinking in Java, Fourth Edition, by Bruce Eckel, Prentice Hall, 2006 ([https://sd.blackball.lv/library/thinking\\_in\\_java\\_4th\\_edition.pdf](https://sd.blackball.lv/library/thinking_in_java_4th_edition.pdf))

### Web links and Video Lectures (e-Resources):

- Java Tutorial: <https://www.geeksforgeeks.org/java/>
- Introduction To Programming In Java (by Evan Jones, Adam Marcus and Eugene Wu): <https://ocw.mit.edu/courses/6-092-introduction-to-programming-in-java-january-iap-2010/>
- Java Tutorial: <https://www.w3schools.com/java/>
- Java Tutorial: [https://www.javatpoint.com/java-tutorial`](https://www.javatpoint.com/java-tutorial)

### Activity Based Learning (Suggested Activities)/ Practical Based learning

1. Installation of Java (Refer: [https://www.java.com/en/download/help/index\\_installing.html](https://www.java.com/en/download/help/index_installing.html))
2. Demonstration of online IDEs like geeksforgeeks, jdoodle or any other Tools
3. Demonstration of class diagrams for the class abstraction, type visibility, composition and inheritance

### Assessment Method

1. Programming Assignment / Course Project

## DEPARTMENT OF COMPUTER SCIENCE & DESIGN

The Correlation of Course Outcomes (CO's) and Program Outcomes (PO's)													
Subject Code:	BCS306 A			TITLE: OBJECT ORIENTED PROGRAMMING WITH JAVA									
List of Course Outcomes	Program Outcomes												Total
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	
C01	2	2	2	2	2	0	0	0	0	0	0	2	12
C02	3	2	2	2	2	0	0	0	0	0	0	2	13
C03	2	2	2	2	3	0	0	0	0	0	0	2	13
C04	2	2	2	2	3	0	0	0	0	0	0	2	13
C05	2	2	2	2	2	0	0	0	0	0	0	2	11
Total	11	10	10	10	12	0	0	0	0	0	0	10	63
Note: 3 = Strong Contribution 2 = Average Contribution 1 = Weak Contribution 0 = No Contribution													

The Correlation of Program Specific Outcome's (PSO's) and Program Outcomes (PO's)			
Subject Code:	BCS306 A		TITLE: OBJECT ORIENTED PROGRAMMING WITH JAVA
List of Course Outcomes	Program Specific Outcomes		Total
	PSO-1	PSO-2	
C01	2	2	4
C02	2	2	4
C03	2	2	4
C04	2	2	4
C05	2	2	4
Total	10	10	20
Note: 3 = Strong Contribution 2 = Average Contribution 1 = Weak Contribution 0 = No Contribution			