Kadi Sarva Vishwavidyalaya, Gandhinagar

25CAMJ206: Object Oriented Programming with Java

Program: B.E. Computer Engineering / Computer Science and Engineering / Information Technology (Year - 2)
Teaching Scheme: 2L - 0T - 4P (4 Credits)
Prerequisite: Basic knowledge of procedural programming (C/C++), control structures, functions, and basic OOP concepts

Rationale: This course builds a strong programming foundation by integrating theoretical and practical aspects of object-oriented programming using Java. It enables students to design and develop real-world applications while preparing them for advanced studies and professional careers in modern software development.

1. **Course Objectives:**

The Purpose of learning this course is

- To understand the fundamentals of Java programming including syntax, data types, and control structures
- To apply object-oriented programming principles such as inheritance, polymorphism, and abstraction
- To develop modular and reusable applications using classes, interfaces, and packages
- To utilize collections and exception handling for efficient and robust application development
- To implement advanced concepts like file handling and multithreading
- To apply integrated Java concepts to solve real-world problems

2. Teaching and Evaluation Scheme:

Teaching Scheme					Examination Scheme				
L	**T**	**P**	**C**	**Hrs / Week**	**IE**	**EE**	**CIA**	**Practical**	**Total Marks**
2	-	4	4	6	40	60	30	20	150

IE: Internal Exam
EE: External Exam (End Semester)
CIA: Continuous Internal Assessment
Practical: Practical Exam (End Semester)

3. Detailed Syllabus

Unit	Topic	Lectures	% weigh	Outcomes
1	**Introduction to Java, Classes, Objects and Methods:** Basics of JAVA, History, Features of Java, Java Development Kit (JDK), Java Virtual Machine (JVM), byte code, structure of a Java program, Java program compilation and execution. Identifiers, Variables, Data types, Literals, constants, operators, expressions, Type conversions, Control structure, Loops, Arrays, Input/output using Scanner Class and System classes. Defining classes, Creating objects and assigning to Reference Variables, constructors and object initialization, this keyword, Garbage Collection, finalize() Method, Method overloading, argument passing, Access modifiers, Static Members and Methods, final keyword, nested and inner classes, command-line arguments. Inbuilt classes: String Class,	08	25%	CO1

	MATH Class, Wrapper Classes. **Focus on strengthening computational thinking and programming literacy contributing to Quality Education.**			
2	**OOPS concepts, Interfaces and Packages:** Concept of inheritance, Types of Inheritance, Method Overriding, Dynamic Method Dispatch, Super Keyword, abstract classes, Final classes and methods. Polymorphism, Run Time Binding. Defining, implementing, nesting and applying interfaces, variable in interfaces, extending interfaces. Defining package, finding packages and CLASSPATH, importing packages, User-defined packages, Access Modifiers in Packages, Built-in packages, UTIL Package. **Emphasis on modular, reusable design and structured programming for improved learning outcomes.**	08	25%	CO2
3	**Collections and Exception Handling:** Introduction to Java Collection Framework, Collection classes: ArrayList, LinkList and TreeSet. Collection Interfaces: List, Set, Map, Queue, Collection. Exception-Handling Fundamentals, Exception Types, Uncaught Exceptions, Try and catch, Multiple catch Clauses, Throw, Throws, finally, Built-in Exceptions, Creating your own Exception subclasses. **Development of efficient, reliable, and optimized software systems aligned with Industry, Innovation and Infrastructure.**	07	25%	CO3
4	**I/O Programming, Multithreading:** File handling using File, File Reader, FileWriter, BufferReader, Serialization and Deserialization. Thread life cycle, Thread class, Runnable interfaces, Main Thread, Create a Thread, Creating a multiple Threads, Thread priorities, Thread synchronization. **Focus on scalable, high-performance and concurrent applications supporting modern digital infrastructure.**	07	25%	CO4,CO5

4. List of Experiments:

No	Details	Hours	CO
1	Java compilation and Execution, Basic Programming Concepts 1) Write a Java Program to display a greeting message like: "Welcome to Java, Learning Java Programming is fun" on the console. 2) Write a Java program that reads a number in meters, convert it to feet, and displays the result. 3) Write a Java Program to stimulate a simple calculator. Provide 2 numbers and operator from console to perform Addition, Subtraction, Modulo, Power and Square root. If the operator is not valid Give appropriate Message	6	CO1
2	Operator, Loops, Conditions, Math Class 4) Write a Java program that takes an array of 4-bit binary to visually represent numbers 0–15 in binary. Take any two integers, perform operators and print the result. 5) Write a program that prompts the user to enter a letter and check whether a letter is a vowel or constant.	6	CO1

No	Details	Hours	CO
	6) Write a Java program to test the command line argument. Display Total Number of arguments and display one argument in one line.		
3	Arrays, Class, Object, Constructor 7) Write a Java program that finds if there is a subarray with a sum of 0 from given an array of positive and negative numbers. 8) Write a Java program to display area of circle by creating Area Class and areaCircle Method. 9) Write a Java program that displays a person's name and age using a person class and parameterized constructor. 10) Write a Java program that displays book title and price using this keyword. 11) Write a Java Program that compares two students' marks and return the object with the higher marks. 12) Write a Java program to print fibonacci series using a recursion function.	8	CO2
4	String Handling 13) Write a Java Program to ask a user to enter a paragraph and perform operations like sentences and word counting, character frequency and word search. 14) Write a Java Program that arranges the words of string in descending order of length and alphabetically for words of the same length. Store the output in a text file.	4	CO2
5	Static Keyword, Inner Class, Inheritance, Polymorphism 15) Write a Java Program to display the total number of cars created and model name of cars using static variables and methods. 16) Write a Java program demonstrating inner class in the library system. 17) Write a Java program that creates a class CocidParameters inheriting from Patient class. Implement constructor and member variables like CTScore, D-dimer and platelet count. 18) Write a Java program that demonstrates method overloading and overriding using Restaurant billing system.	8	CO2
6	Interfaces, package, UTIL Package 19) Write a Java program to create interface Polygon with methods for calculating area, perimeter and displaying values. Implement Square and Rectangle classes. 20) Write a Java application to demonstrate a package using an ecommerce system that contains customer, product and order classes. 21) Write a Java Program to browse history Using Stack.	6	CO2
7	Collections 22) Write a Java Program to manage stock of Fruits Using ArrayList. 23) Write a Java program to print a queue Using Queue. 24) Write a Java program to create a Music Playlist Using LinkedList. 25) Write a Java Program to manage city names in alphabetic order Using TreeSet.	6	CO3
8	Exception Handling 26) Write a Java application to create a MathFunctions class with methods for calculating mean and division. Handle custom exceptions for invalid numbers and divided by zero. 27) Write a Java application to create a BankAccount class with methods for deposit, withdrawal and balance checking. Handle custom exceptions for negative amounts, insufficient funds and low balances.	4	CO3
9	I/O	4	CO4,CO5

No	Details	Hours	CO
	28) Write a stream-based program to accept and validate user details, handle exceptions and save data to a file. Display saved records. 29) Write a Java Program to create a backup file utility.		
10	Multithreading 30) Write a Java program that creates two threads to print odd and even numbers sequentially. Synchronizes the threads if needed. 31) Write a Java program to set thread priority with CPU-Intesive Task. 32) Write a Java Program in that Main Thread acts as server, spawning child threads to handle requests concurrently.	8	CO4,CO5

5. Course Outcomes (COs)

After completion of the course, the student will be able to:

CO No.	Course Outcome (CO)	RBT Level (Cognitive Domain)
CO1	**Understand** the fundamentals of Java programming, including basic syntax, data types, operators, control structures, arrays, and input/output operations, and write simple Java programs.	Understand
CO2	**Apply** object-oriented programming concepts such as classes, objects, constructors, method overloading, inheritance, polymorphism, interfaces, and packages to design modular and reusable Java applications.	Analyze ,Apply
CO3	**Utilize** Java Collection Framework and exception handling mechanisms to efficiently manage data, handle runtime errors, and develop robust Java applications.	Analyze , Evaluate
CO4	**Implement** file I/O operations, serialization, and multithreading in Java programs to build concurrent and persistent applications.	Analyze , Evaluate , Apply
CO5	**Apply** integrated Java concepts (OOP, collections, exception handling, file I/O, multithreading) to solve real-world programming problems with focus on innovation, scalability, and efficient software solutions aligned with Industry, Innovation and Infrastructure and skill development under Quality Education.	Apply

6. Mapping of COs with POs & PSOs:

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	3	–	–	3	–	–	–	–	3	3	3	-
CO2	3	3	3	–	3	–	–	–	3	3	3	3	-
CO3	3	3	-	3	3	–	–	–	3	–	3	3	3
CO4	3	3	3	3	3	–	–	–	3	–	3	3	3
CO5	3	3	3	3	3	–	–	–	3	2	2	3	3
Avg.	**3.00**	**3.00**	**2.25**	**2.25**	**3.00**	**0.00**	**0.00**	**0.00**	**2.40**	**2.00**	**2.40**	**3.00**	**2.40**

7. Reference Reading:

7.1 Text Books

1. Schildt, H.(2017). Java: The Complete Reference (9th ed.), McGraw Hill.
2. Balagurusamy, E. (2023). Programming with Java (7th ed.). McGraw Hill.

7.2 Reference Books

3. Horstmann, C. Cornell, G. (2017). Core Java Volume I – Fundamentals (8th ed.), Pearson Education.
4. Sierra, K. Bates, B. (2009). Head First Java (2nd ed.), O'Reilly
5. Nageswara Rao, R. (2016) Core JAVA: An Integrated Approach. Dreamtech Publication.

7.3 Online References

6. JDK toolkit, Notepad++, VS Code with JAVA Extension
7. www.nptel.ac.in
8. www.coursera.org
9. https://docs.oracle.com/javase/tutorial/
10. https://www.tutorialspoint.com/java/index.htm

8. Course Outcomes Mapping Table

CO	Course Outcome (CO)	POs/PSOs Map
CO1	**Understand** the fundamentals of Java programming, including basic syntax, data types, operators, control structures, arrays, and input/output operations, and write simple Java programs.	PO1, PO2, PO5, PO10, PO11, PO12, PSO1
CO2	**Apply** object-oriented programming concepts such as classes, objects, constructors, method overloading, inheritance, polymorphism, interfaces, and packages to design modular and reusable Java applications.	PO1, PO2, PO3, PO5, PO9, PO10, PO11, PO12, PSO1
CO3	**Utilize** Java Collection Framework and exception handling mechanisms to efficiently manage data, handle runtime errors, and develop robust Java applications.	PO1, PO2, PO4, PO5, PO9, PO11, PO12, PSO1, PSO2
CO4	**Implement** file I/O operations, serialization, and multithreading in Java programs to build concurrent and persistent applications.	PO1, PO2, PO3, PO4, PO5, PO9, PO11, PO12, PSO1, PSO2
CO5	**Apply** integrated Java concepts (OOP, collections, exception handling, file I/O, multithreading) to solve real-world programming problems with focus on innovation, scalability, and efficient software solutions aligned with Industry, Innovation and Infrastructure and skill development under Quality Education.	PO1, PO2, PO3, PO4, PO5, PO9, PO10, PO11, PO12, PSO1, PSO2

9. Evaluation Scheme:

- The course evaluation shall comprise Internal Examination (IE) and End Semester Examination (EE) for theory, along with Continuous Internal Assessment (CIA) and External Practical Examination.
- The Internal Examination (40 marks) will be conducted through one or more mid-semester tests covering the prescribed syllabus.
- The End Semester Examination (60 marks) will assess the overall understanding of the course with emphasis on problem-solving and analytical ability.

- The Practical Examination (20 marks) shall be conducted externally and will include implementation, performance, and viva-voce.
- Students are required to secure minimum qualifying marks in each component separately as per university rules.

10. Pedagogical Framework:

- The course adopts a blended teaching approach combining lectures, coding demonstrations, and interactive problem-solving sessions.
- Emphasis is placed on learning by doing through extensive laboratory work, assignments, and mini-projects.
- Conceptual understanding is reinforced with real-world examples and algorithm visualization techniques.
- Students are encouraged to engage in collaborative learning, discussions, and peer problem-solving.
- Continuous feedback through assessments and lab performance supports progressive skill development and mastery.

11. Innovative Practices:

- Integration of visualization tools and coding platforms to enhance conceptual understanding and engagement.
- Use of problem-based learning (PBL) with real-world scenarios to strengthen analytical and design skills.
- Incorporation of mini-projects and case studies to connect data structures with practical applications.
- Encouragement of competitive programming and coding challenges to improve speed and accuracy.
- Adoption of peer learning, code reviews, and flipped classroom techniques to promote active learning and collaboration.
- Incorporation of SDG-oriented case studies and sustainable software design practices aligned with SDG 4 (Quality Education) and SDG 9 (Industry, Innovation and Infrastructure).

12. Self-Learning Component:

- Students are encouraged to engage in self-paced learning through online coding platforms (NPTEL/SWAYAM), tutorials, and technical resources to strengthen their understanding of Java programming and object-oriented concepts. (E.g. *Programming in Java* courses by IITs available on NPTEL – https://nptel.ac.in)
- Regular practice of coding problems beyond the syllabus helps in improving problem-solving skills and logical thinking.
- Learners can explore advanced Java topics such as multithreading, collections framework internals, file handling, JDBC, and design patterns independently.
- Participation in coding competitions, hackathons, and open-source contributions promotes continuous self-improvement.
- This approach fosters lifelong learning habits, enabling students to keep pace with evolving technologies and industry demands.

Sustainable Development Goals (SDG) Integration

This course integrates sustainable software development principles through efficient coding, modular design, and scalable application development using Java. It aligns with Quality Education by enhancing programming skills, problem-solving ability, and lifelong learning. The course supports Industry, Innovation and Infrastructure by enabling development of robust and innovative software systems. It also incorporates software engineering practices like reusability, modularity, and efficient resource utilization for sustainable digital solutions.

SDG Mapping with Course Outcomes

CO	Course Outcome	Relevant SDGs	Justification
CO1	Understand the fundamentals of Java programming, including basic syntax, data types, operators, control structures, arrays, and input/output operations, and write simple Java programs.	SDG4	Builds strong foundational programming knowledge and learning skills
CO2	Apply object-oriented programming concepts such as classes, objects, constructors, method overloading, inheritance, polymorphism, interfaces, and packages to design modular and reusable Java applications.	SDG4	Develops structured thinking and core software design skills
CO3	Utilize Java Collection Framework and exception handling mechanisms to efficiently manage data, handle runtime errors, and develop robust Java applications.	SDG9	Improves efficiency, reliability, and scalability of software systems
CO4	Implement file I/O operations, serialization, and multithreading in Java programs to build concurrent and persistent applications.	SDG9	Enables development of high-performance and real-world computing applications
CO5	Apply integrated Java concepts (OOP, collections, exception handling, file I/O, multithreading) to solve real-world programming problems with focus on innovation, scalability, and efficient software solutions aligned with Industry, Innovation and Infrastructure and skill development under Quality Education.	SDG4, SDG9	Promotes innovation, industry readiness, and holistic programming competency