Gyanmanjari Innovative University

Syllabus
Gyanmanjari Institute of Technology Semester-1
(B.Tech.)

Subject: Java: Building Blocks of Programming –BET1CE11301

Type of course: Engineering Science Courses

Prerequisite: Basic Knowledge of C and C++ Language

Rationale:

This course is designed to introduce and develop object-oriented programming (OOP) concepts, techniques, and applications using the Java programming language. The focus is on fundamental principles of structured design through the use of classes, encompassing development, testing, implementation, and documentation. Key OOP topics covered include the creation and use of classes and objects. Java, as the primary language for this course, is a simple, portable, distributed, robust, secure, dynamic, architecture-neutral, and object-oriented programming language. It is specifically designed to support the development of reliable, high-performance, and cross-platform applications that can operate on a wide variety of computing platforms, including real-time and embedded systems. By enabling applications to run across diverse and heterogeneous environments, Java facilitates broader service delivery, enhances user productivity, and improves communication and collaboration across both enterprise and consumer applications. Originally developed as part of a research project aimed at creating advanced software for network devices and embedded systems, Java serves as the instructional foundation for this course.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks		Total Marks
CI	T	P	C	SEE	CCE	
4	0	2	5	100	50	150

Legends: CI-Class Room Instructions; T – Tutorial; P - Practical; C – Credit; SEE - Semester End Evaluation; CCE-Continuous and Comprehensive Evaluation.

Course Content:

Sr. No	Course Content	Hrs.	% Weightage
1	**Introduction to Java Programming Language:** ***Theory Topics:*** Introduction to Java and brief history, java features, java Applications, Java components: Java Virtual Machine (JVM), Java Runtime Environment (JRE), JDK (Java Development Kit), Importance of byte code and Garbage collection, Java environment setup: structure of java program, compilation and execution of java program, comment, Syntax, Primitive data types: byte, short, int, long, float, double, char, Boolean, Identifiers, declarations of constants & variables, type conversion and type casting, scope of variables, Array, Types of Arrays: one dimensional and two-dimensional array. Different Operators: Arithmetic, Bitwise, Relational (Comparison), Logical, Assignment, Conditional, Ternary, Increment and Decrement, Decision & Control Statements: Selection Statement (if, if...else, switch), Loops (while, do-while, for), break, continue, return. ***Practical:*** 1. Write a Java program to display "Hello, World!" and Include comments to explain each part of the program. 2. Write a Java program to declare and display variables of all primitive data types (byte, short, int, long, float, double, char, boolean). 3. Write a program that converts data types using type casting and type conversion (e.g., int to float, double to int). 4. Write a Java program that accepts two numbers as input and performs all basic arithmetic operations: addition, subtraction, multiplication, division, and modulus. 5. Write a Java program to input 5 integers into a one-dimensional array, then display the array elements and calculate the sum and average. 6. Write a Java program that determines whether a given number is even or odd using an if-else statement 7. Write a Java program that displays the corresponding day of the week based on user input (1 to 7) using a switch statement. 8. Write a Java program to print numbers from 1 to 10 using all three loop types (for, while, and do-while) and use break and continue to skip number 5 and stop at 8. 9. Write a Java program that prints the Fibonacci series up to a specified number using a for loop.	18	20%

10. Create a Java program to calculate the factorial of a given number using a while loop.
11. Develop a Java program to store 10 integers in a one-dimensional array, display them, and sort the array in ascending order.
12. Write a Java program that performs the addition of two matrices using two-dimensional arrays.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Student Info Analyzer & Report Generator" – Java Mini Project:** Create a console-based Java application that stores and processes student details, uses arrays, operators, data types, control structures, and generates a performance report based on input.	20	
2	**Active Learning Assignment Java Greeting Card Generator:** To use basic Java concepts like data types, variables, arrays, loops, and conditionals **to create a personalized greeting card** generator in the console.		10
	Total	20	10

Examination Style:

Student Info Analyzer & Report Generator (20 Marks)
Create a console-based Java application that stores and processes student details, uses arrays, operators, data types, control structures, generates a performance report based on input. By applying Unit-1 Java fundamentals, the student will input student details, process the data, and present the output in a formatted report.

Java Greeting Card Generator (10 Marks)
To use basic Java concepts like data types, variables, arrays, loops, and conditionals to create a personalized greeting card generator in the console. Take User Input as Their name, Occasion (Birthday, Festival, Congratulations, etc.), Favorite symbol or emoji (like *, #, @, 🎉, etc.) and Generate Creative Output using loops and conditionals to display a decorated border using the chosen symbol, print a creative message using the occasion and their name, also show a simple ASCII design using a 2D array or loop pattern.

2	**Object Oriented Programming Concepts:** ***Theory Topics:*** Procedure-Oriented vs. Object Oriented Programming concept, Basics of OOP: Class, Object, Encapsulation, Polymorphism, Abstraction, Inheritance, Defining classes, fields and methods, creating object, Accessing rules: public, private, protected, default, this keyword, static keyword, final keyword, Constructors: Default constructors, Parameterized constructors, Copy constructors, Passing object as a parameter, method overloading, constructor overloading, Wrapper class and String class and its methods: charAt(), contains(), format(), length(), split(), User Input: Scanner class and Command Line Arguments. ***Practical:*** 1. Write a Java program that illustrates encapsulation by declaring class fields as private and accessing them through getter and setter methods. 2. Write a Java program to demonstrate inheritance by creating a base (parent) class and a derived (child) class, and invoking inherited methods. 3. Write a Java program that showcases method overloading by defining multiple methods with the same name but different parameter lists. 4. Write a Java program that demonstrates constructor overloading by writing multiple constructors with different sets of parameters. 5. Write a Java program to illustrate polymorphism through method overriding using a parent class and a child class. 6. Write a Java program to demonstrate the use of the "this" keyword to distinguish between instance variables and method parameters with the same names. 7. Write a Java program that uses the "static" keyword to declare static variables and methods within a class. 8. Write a Java program that demonstrates the use of the "final" keyword by declaring a final variable, creating a final method, and defining a final class. 9. Write a Java program that uses a copy constructor to create an object by copying another object and passing it as a parameter. 10. Write a Java program to illustrate the use of different access specifiers (public, private, protected, and default) across multiple classes. 11. Write a Java program to demonstrate the use of wrapper classes (Integer, Double, Character) and perform both boxing and unboxing operations.	18	20%

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Student Management System: From Procedural to Object-Oriented Transformation:** To help students understand the difference between Procedure-Oriented Programming (POP) and Object-Oriented Programming (OOP) by building a Student Management System using both approaches and exploring core OOP concepts and Java features.	20	
2	**Concept Check Challenge – OOP Knowledge Snap Quiz:** The quiz is designed to assess understanding through **scenario-based MCQs** that apply real-life logic, code interpretation, and conceptual clarity.		10
	Total	20	10

Examination Style:

Student Management System: From Procedural to Object-Oriented Transformation (20 Marks)
Implement a simple program using POP to take input for student name, ID, marks in 3 subjects using Scanner, calculate total, average, and grade and print the student report using functions without classes. Re-implement the same system using OOP Class Student with: Fields: id, name, marks[], grade, Methods: inputDetails(), calculate(), displayReport(), Constructor overloading (default, parameterized), this keyword for field access, Static field collegeName (shared among all students), Final variable for total subjects. Additionally use StudentHelper class,passObject method, String and wrapper class with command line input.

Concept Check Challenge – OOP Knowledge Snap Quiz (10 Marks)
The quiz is designed to assess understanding through **scenario-based MCQs** that apply real-life logic, code interpretation, and conceptual clarity.Five multiple-choice questions should be included from this unit for evaluation purposes. Each question carries **1 mark**, totaling **10 marks**.

3	**Inheritance, Packages & Interfaces:** ***Theory Topics:*** Basics of Inheritance, Types of inheritance: single, multiple, multilevel, hierarchical and hybrid inheritance, method overriding, Object class and overriding its methods: equals (), toString(), finalize(), hashCode(), Defining interface, implementing interface, multiple inheritance using interface, Abstract class and final class, Creating packages, importing packages, and access rules for packages. ***Practical:*** 1. Write a Java program that demonstrates single inheritance by defining a parent class and a child class, and calling the inherited methods from the child class. 2. Write a Java program that implements multilevel inheritance to show how methods are passed through multiple levels of class hierarchy. 3. Write a Java program that illustrates hierarchical inheritance, where multiple child classes inherit from one common parent class. 4. Write a Java program to override a method in a child class and call both the parent and child class methods using an object. 5. Write a Java program that overrides the equals(), toString(), finalize(), and hashCode() methods from the Object class, and demonstrates their functionality. 6. Write a Java program with an abstract class that includes an abstract method, then extend it in a subclass and provide an implementation for the abstract method. 7. Write a Java program that defines an interface, implements it in a class, and demonstrates multiple inheritance using interfaces. 8. Write a Java program to show how the final keyword is used with a variable, a method, and a class to restrict modification or inheritance. 9. Write a Java program that creates a custom package, imports it into another class, and accesses its members. 10. Write a Java program that demonstrates the use of different access specifiers (public, private, protected, and default) within packages and explains how each affects class member accessibility.	18	20%

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Identify and Fix the Error:** Students will analyze and correct a faulty code snippet to develop debugging skills and reinforce programming concepts.	10	
2	**Tiny Project** Students will build a small Java project applying OOP principles like inheritance, interfaces, and abstraction to solve a real-world scenario.	10	
3	**Active Learning Assignment Poster Presentation:** Poster Topic will be provided by the subject faculty. Students need to prepare posters and upload files on GMIU Web Portal individually.		10
	Total	20	10

Examination Style:

Identify and Fix the Error (10 marks)

In this activity, the faculty will provide students with a code snippet that contains logical or syntactical errors. Students are required to carefully analyze the code, identify the mistakes, and correct them to ensure the program compiles and runs successfully. This exercise is designed to sharpen students' debugging skills, enhance problem-solving abilities, and strengthen their understanding of programming concepts.

Tiny Project (10 Marks)

This Tiny Project is designed to give students hands-on experience with core OOP principles, specifically focusing on inheritance, method overriding, abstract classes, interfaces, and packages in Java. Students will develop a small-scale application that demonstrates the practical implementation of these concepts in a real-world scenario.

Poster Presentation of Unit 3 (10 marks)

Prepared poster and give presentation on the given topics.

4	**Exception Handling & Multithreading:** ***Theory Topics:*** Types of errors, exceptions, try...catch statements, multiple catch blocks, throw and throws keywords, finally clause, uses of exceptions, user defined exceptions, Concept of Multithreading, creating thread, extending Thread class, implementing Runnable interface, life cycle of a thread, Thread priority, Thread exception handling in threads. ***Practical:*** 1. Write a Java program to handle an arithmetic exception (e.g., divide by zero) using a try-catch block. 2. Write a Java program to demonstrate multiple catch blocks for handling exceptions such as ArithmeticException, ArrayIndexOutOfBoundsException, and NullPointerException. 3. Write a Java program to show the use of the finally block in exception handling. 4. Write a Java program to use the throw keyword to manually throw a custom exception and handle it with a try-catch block. 5. Write a Java program to create and handle a user-defined exception for validating age input for voting eligibility. 6. Write a Java program to create a thread by extending the Thread class and executing its run() method. 7. Write a Java program to implement a thread using the Runnable interface and start it using the Thread class. 8. Write a Java program to demonstrate the life cycle of a thread using methods like start(), sleep(), and join(). 9. Write a Java program to set and demonstrate thread priority and observe its effect on the thread execution order. 10. Write a Java program to handle exceptions inside a thread using a try-catch block within the run() method.	18	20%

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Problem solving using Programming:** Problem statements will be provided to the student in the Lab session. students need to perform its allocated Problem statement and upload it on the GMIU Web Portal. (Group of two students)	20	
2	**Active Learning Assignment Game in Java Using OOP Concepts and Concurrency:** This project involves creating a game application in Java that tests players' knowledge by asking questions in multiple categories.		10
	Total	20	10

Examination Style:

Problem solving using Programming (20 marks)

Students, in groups of two, will receive problem statements during the session, complete their assigned task, and upload it to the GMIU Web Portal with Statement, Code and Output.

Game in Java Using OOP Concepts and Concurrency (20 marks)

This project involves creating a game application in Java that tests players' knowledge by asking questions in multiple categories. The game should be designed using object-oriented principles, organized into packages, and include features like inheritance for question types, concurrency for timed quizzes, and robust exception handling for input validation. Arrays will store questions and player responses, and loops and conditionals will control the game flow.

5	**File Handling and Collections Framework:** ***Theory Topics:*** Stream classes, class hierarchy, Useful I/O classes: FileInputStream, FileOutputStream, Creation of text file, reading and writing text files, Collections Framework overview, Collection classes: ArrayList, LinkedList, HashSet, The For-Each loop, Map class: HashMap.	18	20%

Practical:

1. Write a Java program to create a text file and write data using FileOutputStream.
2. Write a Java program to read data from a text file using FileInputStream.
3. Write a Java program to read a text file line by line using BufferedReader.
4. Write a Java program to copy content from one file to another using FileInputStream and FileOutputStream.
5. Write a Java program to use an ArrayList for adding, removing, and iterating elements with a for-each loop.
6. Write a Java program to perform insert, delete, and traverse operations using a LinkedList.
7. Write a Java program to demonstrate HashSet by adding elements, avoiding duplicates, and iterating the set.
8. Write a Java program to use a HashMap for storing key-value pairs, retrieving data, and iteration.
9. Write a Java program to use a for-each loop to iterate through an ArrayList, HashSet, and HashMap.
10. Write a Java program to count words in a text file and store unique words in a HashSet.

Evaluation Method:

Sr. No.	Evaluation Methods	SEE	CCE
1	**Mini Library Management System using Java I/O and Collections:** To give students hands-on experience with Java Stream Classes, File Handling, and Collections Framework by building a simple text-based library system that allows storing, retrieving, and displaying book records using files and collections.	20	
2	**Reverse Engineering.** Output of the program will be given to the student in the session; students need to perform its program and upload it on GMIU Web Portal. (Group of two students)		10
	Total	20	10

	Examination Style: **Mini Library Management System using Java I/O and Collections (20 marks)** To give students hands-on experience with **Java Stream Classes**, **File Handling**, and **Collections Framework** by building a simple text-based library system that allows storing, retrieving, and displaying book records using files and collections. Students must use Stream class hierarchy (FileInputStream, FileOutputStream, FileReader, FileWriter), Creating, writing, and reading text files, Using ArrayList, HashSet, LinkedList, HashMap, Enhanced for-each loop, Real-world use of Map for key-value storage in their mini library project. **Reverse Engineering Problem Statement of Unit 5 (10 marks)** Students, in groups of two, will receive problem statements during the lab session, complete their assigned task, and upload it to the GMIU Web Portal with Statement, Code and Output.		

Suggested Specification table:

Distribution of Marks (Revised Bloom's Taxonomy)						
Level	Remembrance (R)	Understanding (U)	Application (A)	Analyze (N)	Evaluate (E)	Create (C)
Weightage %	10%	30%	20%	10%	-	30%

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 the above table.

Course Outcome:

After learning the course, the students should be able to:	
CO1	Apply basic Java concepts like data types, operators, control structures, arrays, and environment setup to build structured programs.
CO2	Understand core OOP concepts in Java, including classes, constructors, access modifiers, and user input handling.
CO3	Demonstrate the use of inheritance, interfaces, abstract/final classes, and package management in Java to create structured and reusable code.
CO4	Develop robust and concurrent Java applications using exception handling and multithreading techniques.
CO5	Explore Java's I/O and Collections APIs to manage files and structure data in applications.

Instructional Method:

The course delivery method will depend upon the requirement of content and needs of students. The teacher, in addition to conventional teaching methods by black board, may also use any tools such as demonstration, role play, Quiz, brainstorming, MOOCs etc.

From the content 10% topics are suggested for flipped mode instruction.

Students will use supplementary resources such as online videos, NPTEL/SWAYAM videos, e-courses, Virtual Laboratory.

The internal evaluation will be done on the basis of the Active Learning Assignment.

Practical/Viva examination will be conducted at the end of semester for evaluation of performance of students in the laboratory.

Students will use Notepad or Notepad++ editor for writing practical programs.

Reference Books:

[1] Java The Complete Reference, Ninth Edition, Comprehensive Coverage of the Java Language, Herbert Schildt, Oracle Press.

[2] Object oriented programming with Java, Rajkumar Buyya,S Thamarai Selvi, Xingchen Chu, McGrawHill.

[3] Programming with JAVA, E Balagurusamy, McGrawHill.

[4] Intro to Java Programming, 10th edition, Y.Daniel Liang, Pearson.

[5] Programming in Java, Sachin Malhotra, Saurabh Choudhary, Oxford.