SHREE SWAMI ATMANAND SARASWATI INSTITUTE OF TECHNOLOGY

COMPUTER ENGINEERING DEPARTMENT

Problem Solving Exercise (PBL)

Subject Name & Code: Object Oriented Programming (BE04000231)

Year (Semester): 2nd Year (4th Semester)

Academic Year: Jan-2026. **Term Duration:** 01/01/2026 to 06/05/2026

Sr. No.	List of Experiment
Practical Set-1 (Basics of JAVA) : CO 1	
1	Develop a Java program that prompts the user to enter a distance in meters. Your program should then convert this distance to feet (1 meter = 3.28084 feet) and display the result formatted to two decimal places.
2	Write a Java program to solve a system of two linear equations with two variables (e.g., ax + by = e and cx + dy = f). Prompt the user to enter the coefficients a, b, c, d, e, f. Calculate and display the values of x and y using Cramer's rule. Include error handling for cases where the denominator is zero Cramer's rule : D=ad−bc, D_x=ed−bf, D_y=af−ce, x=D_x/D, y=D_y/D
3	Write a Java program that prompts the user to enter a single letter (character). Determine whether the entered character is a vowel (a, e, i, o, u, case-insensitive) or a consonant, and display the result.
4	Develop a Java application that calculates a person's Body Mass Index (BMI). The program should ask the user for their weight in pounds and height in inches. Convert these values to kilograms and meters respectively (1 pound = 0.45359237 kg, 1 inch = 0.0254 meters) and then calculate BMI (weight in kg / (height in meters)^2). Display the calculated BMI.
5	Write a program that takes the lengths of three sides of a triangle as input. Calculate and print the area of the triangle. Ensure that the program validates if the given side lengths can actually form a triangle (sum of any two sides must be greater than the third side) before calculating the area (use Heron's formula: Area = sqrt(s*(s-a)*(s-b)*(s-c)) where s = (a+b+c)/2).
Practical Set-2 (Basic OOP concepts-1) : CO 2	
6	Define a Java class named Rectangle. It should have two double data fields: width and height, both with a default value of 1. Implement a no-argument constructor and a constructor that takes width and height as parameters. Include methods getArea() and getPerimeter() that return the calculated area and perimeter respectively.
7	Create an Employee class with private instance variables for employeeName (String) and employeeSalary (double). Implement public methods readEmployeeData() (to take input from the user) and displayEmployeeData() (to print the employee's name and salary). Demonstrate object creation and method invocation in a main method.
8	Create a Point class representing a 2D point (x, y). Implement a default constructor that initializes both x and y to 5. Provide a parameterized constructor to initialize x and y with user-supplied values. Also, implement a copy constructor to create a new Point object as a copy of an existing Point object. Include a display() method to show the point's coordinates and write a main method to test all constructors and the display functionality.
9	Define a Java class named Rectangle. It should have two double data fields: width and height. In your main method, create two Rectangle objects: one with width 4 and height 40, and another with width 3.5 and height 35.9. For each rectangle, display its width, height, calculated area, and perimeter. Then, compare the two rectangles based on their areas and print which one has a larger area.
Practical Set-3 (Basic OOP concepts-1) : CO 2	
10	Design a class BankAccount with account_holder_name and balance. Use a static variable interest_rate (same for all accounts). Include methods to calculate and display the interest earned.

	Update interest rate using a static method.
11	Write a Java program to model a college admission system using the concept of inner classes. Create an outer class named College that stores the collegeName as a data member and initializes it through a constructor. Within the College class, define a non-static inner class named Admission that contains student-specific details such as the studentName and the course they are enrolling in. The inner class should have methods to accept and display student information, and it should also be able to access and display the collegeName stored in the outer class. In the main method, create an object of the College class by passing the collegeName, and then use this object to create an instance of the inner Admission class. Invoke appropriate methods to input the student's name and course, and then display the complete admission details, including the college name.
12	Write a Java program that demonstrates method overloading to calculate the volume of different 3D shapes. Implement overloaded methods named calculateVolume() for a Cube (takes one side length), a RectangularCube (takes length, width, height), and a Sphere (takes radius).
Practical Set-4 (Inheritance & Polymorphism) : CO 2	
13	Design a base class Shape with two double data members d1 and d2 to store dimensions. Include a method getData(double d1, double d2) to initialize these dimensions. Create two derived classes, Triangle and Rectangle, which inherit from Shape. Each derived class should have its own method to calculate its specific area.
14	Define a base class BankAccount with common attributes like accountNumber, accountHolderName, and balance. Then, define two subclasses: SavingAccount and FixedDepositAccount, which inherit from BankAccount. Implement basic operations like openAccount(), deposit(), checkBalance(), and withdraw() in BankAccount. The SavingAccount should include a calculateInterest() method specific to savings accounts, and FixedDepositAccount should have a maturityAmount() method considering fixed deposit terms.
15	Create a base class named Employee that contains a method displayDetails() which prints general employee details such as name and department. Now create a subclass Manager that inherits from Employee and overrides the displayDetails() method to include additional information such as the manager's team size or project name. In the main method, create objects of both Employee and Manager classes and call the displayDetails() method using each object to show how Java determines which version of the method to execute at runtime.
Practical Set-5 (Interface, Abstract Class and Package) : CO 2	
16	Given an interface Classify with a method String getDivision(double average). Implement this getDivision method in a class Result such that it returns "First Division" if the average is 60 or more.
17	Write the Java code for an interface named Exam which declares a single abstract method boolean isPassed(int mark). This method should take an integer mark as an argument. Write the Java code for another interface named Classify which declares a single abstract method String getDivision(double average). This method should take a double average as an argument. Create a class named Result that implements both the Exam and Classify interfaces. Provide concrete implementations for isPassed() and getDivision() methods. In your main method, create an instance of Result, set some marks and average, and demonstrate the use of both interface methods.
18	Write a Java program to create an abstract class Vehicle with: ● An abstract method fuelType() that returns the type of fuel used. ● An abstract method noOfWheels() that returns the number of wheels. Create two subclasses: ● Car that uses Petrol/Diesel and has 4 wheels. ● Bike that uses Petrol and has 2 wheels. In the main method, create objects of Car and Bike, and display their fuel type and number of

	wheels.
19	Write a Java program using packages to generate a mark sheet for students. Create a package student that contains a class Student with the following: ● Data members: rollNo, name. ● A constructor to initialize student details. ● A method displayStudent() to display student information. Create another package exam that contains a class Result which: ● Extends the Student class. ● Has data members: marks1, marks2, marks3. ● A method displayResult() that prints the student's mark sheet including total and average marks. In the main method (inside the exam package), create a student with marks and display the mark sheet.
Practical Set-6 (Exception Handling) : CO 3	
20	Take the value of denominator and numerator from user using command-line argument. Implement the concept of exception handling to manage all possible run-time error.
21	Write a Java program to create a class VotingApp where: ● The method checkEligibility(int age) checks if a person is eligible to vote. ● If age < 18, explicitly throw the predefined exception IllegalArgumentException with the message "Age must be 18 or above to vote". In the main method, test the method with different age inputs. ● Use a try-catch-finally block to handle exceptions. ● The finally block should always print "Validation process completed"
22	Define a custom exception class BookNotAvailableException that extends Exception. ● Create a class Library with: ● An instance variable availableBooks (integer). ● A method issueBook(int count) that: o If count <= availableBooks, reduce the number of books and display "Book issued successfully". o Otherwise, throw BookNotAvailableException with the message "Requested books not available". In the main() method: ● Initialize the library with 3 available books. ● Try issuing 2 books (valid). ● Then try issuing 2 more books (should throw the custom exception).
Practical Set – 7 (Concurrency Control – Threading) : CO 3	
23	Write a Java program that creates two threads: ● First thread prints numbers from 1 to 10 at the interval of 1 second. ● Second thread prints numbers from 11 to 20 at the interval of 500 ms. Run both threads and display the output.
24	Write a Java program where Thread T1 prints 1 to 100, T2 prints 101 to 200 and T3 prints 201 to 300. Initiate execution of all the three threads but ensure that numbers should be printed sequentially
25	Write a Java program where two threads print multiplication tables (e.g., Table of 5 and Table of 7). Use a synchronized method so that table outputs do not mix and remain consistent.
Practical Set – 8 (File I/O) : CO 3	
26	Write a program that will count the number of characters, words, and lines in a file. Words are

	separated by whitespace characters. The file name should be passed as a command-line argument.
27	Create a file named students.txt. Write at least three student records (roll number, name, marks) into the file. Read the file content and display all student records on the console. Handle exceptions like IOException properly using try-catch-finally.
28	Write a Java program that reads a text file named data.txt. The program should count and display: The total number of lines, The total number of words, The total number of characters (excluding spaces and newline characters), Use FileReader / BufferedReader for reading the file. Handle exceptions like FileNotFoundException and IOException.
Practical Set – 9 (Collection Framework and Generics) : CO 5	
29	Write a Java program that uses an ArrayList<Integer> to store marks of students. Perform the following operations: ● Add at least 5 marks. ● Display all marks. ● Find and display the highest and lowest marks using Collections.max() and Collections.min().
30	Write a program that accepts a sentence from the user and counts the frequency of each word using a HashMap<String, Integer>. Display the results in the format: Input: "Java is fun and Java is powerful" Output: Java -> 2 is -> 2 fun -> 1 and -> 1 powerful -> 1
31	Write a Java program to simulate a Music Playlist using LinkedList<String>. Perform the following operations: 1. Add songs to the playlist. 2. Display the full playlist. 3. Play the first song (remove from front). 4. Skip the last song (remove from end). 5. Display the updated playlist after each operation.
32	Write a generic method searchElement that accepts a LinkedList<T> and an element T to search. Return true if the element exists, otherwise false. ● Test with LinkedList<Integer> for roll numbers. ● Test with LinkedList<String> for names.
Practical Set – 10 (JAVA FX) : CO 4	
33	Write a JavaFX program that displays five Text nodes vertically (stacked). For each Text: ● Use font Times New Roman, bold + italic, size 22 px. ● Assign a random color and random opacity (between 0.3 and 1.0) to each text. ● Center the texts horizontally in the window and add spacing between them. UI / Classes to use: VBox, Text, Font, Color, Random. On launch the window shows five vertically arranged lines (e.g., “Text 1”, … “Text 5”) each with different color & opacity and same font style.
34	Design a registration form UI with fields: Roll_NO (numeric), Name, Age (numeric), Email and a Submit button. Requirements: ● Validate inputs on submit: o RollNo and Age must be integers. o Email must contain @ and . (basic check). ● On success show a confirmation Alert with entered data. ● On validation failure show an error alert describing the issue. UI / Classes to use: GridPane, TextField, Button, Alert, FileChooser, FileWriter/BufferedWriter.
35	Write a JavaFX program that displays a bar chart to represent the percentage distribution of overall

	grades using Rectangle shapes. ● Projects: 20%, displayed in Red ● Quizzes: 10%, displayed in Blue ● Midterm Exams: 30%, displayed in Green ● Final Exam: 40%, displayed in Orange Requirements: 1. Each category should be displayed with a labeled bar. 2. Bars should be proportional in height to the percentage. 3. Use the Rectangle class to create the bars. 4. Display the labels (e.g., "Projects — 20%") under each bar. 5. Arrange the bars horizontally in the scene using an HBox or Pane.

Prof. Vipul S. Kania	Prof. Vipul S. Kania	Prof. Chirag Patel
Subject Faculty	**Subject Coordinator**	**Head of Department**