**Session 2025-2026**

|  |  |
| --- | --- |
| **Vision:**  *To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration* | **Mission:** *To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies****.*** |

**Program Educational Objectives of the program (PEO):** (broad statements that describe the professional and career accomplishments)

|  |  |  |  |
| --- | --- | --- | --- |
| PEO1 | **Preparation** | **P: Preparation** | **Pep-CL abbreviation**  **pronounce as Pep-si-lL easy to recall** |
| PEO2 | **Core Competence** | **E: Environment (Learning Environment)** |
| PEO3 | **Breadth** | **P: Professionalism** |
| PEO4 | **Professionalism** | **C: Core Competence** |
| PEO5 | **Learning Environment** | **L: Breadth (Learning in diverse areas)** |

**Program Outcomes (PO):** (statements that describe what a student should be able to do and know by the end of a program)

**Keywords of POs:**

Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

**PSO Keywords:** Cutting edge technologies, Research

“I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life.” *to contribute to the development of cutting-edge technologies and Research*.

**Integrity:** I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

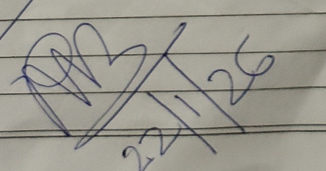
**Name and Signature of Student and Date**

(Signature and Date in Handwritten)



KARTIK GAJANAN PATIL 31/01/2026

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Session** | **2025-26 (EVEN)** | | **Course Name** | **JAVA FSD Lab** | |
| **Semester** | **4th** | | **Course Code** | **23ADS1407** | |
| **Roll No** | **144\_AIDS\_B** | | **Name of Student** | **KARTIK GAJANAN PATIL** | |
|  |  | |  |  |  |
| Practical Number | | **02** | | | |
| Course Outcome | | After completing this practical, we will be able to understand and apply the fundamental concepts of **Object-Oriented Programming** using Java. we will gain knowledge of defining **classes**, creating **objects**, and implementing **methods** to perform operations on data members. This practical enhances the learner’s ability to design simple, structured, and modular programs, thereby improving problem-solving skills and forming a strong foundation for advanced Java programming concepts. | | | |
| Aim | | **To write a Java program to demonstrate the concept of Class, Object, and Methods.** | | | |
| Problem Definition | | **Problem Statement 1**  **Design and implement a Java program to create a class that represents a Student.**  **The class should contain data members such as student name and marks, and methods to**  **accept input and display student details.**  **Create an object of the class and invoke its methods to demonstrate the use of class, object,**  **and methods in Java.**  **Problem Statement (Bank Related) 2 :**  **Develop a Java program to demonstrate the concept of class, object, and methods by**  **creating a class named BankAccount.**  **The class should contain data members such as account holder name, account number,**  **and balance.**  **Implement methods to deposit money, withdraw money, and display account details.**  **Create an object of the class and use it to perform various banking operations.** | | | |
| Theory  (100 words) | | This practical demonstrates the fundamental concepts of **Object-Oriented Programming (OOP)** in Java, namely **class, object, and methods**, using real-world examples such as *Student* and *bankaccount*. A class acts as a blueprint that defines data members and methods. In the Student example, the class contains data members like student name and marks, along with methods to accept and display student details. Similarly, the BankAccount class models a bank account with attributes such as account holder name, account number, and balance, and methods to perform deposit, withdrawal, and display operations. Objects of these classes are created to access and invoke the methods, enabling interaction with the data. This practical helps in understanding how Java programs can be structured using classes and objects to represent real-life entities, thereby improving code modularity, reusability, and clarity. | | | |
| Procedure and Execution  (100 Words) | | Algorithm:  FOR PROBLEM STATEMENT 1:  1. Start the program.  2. Define a class named Student.  3. Declare data members (variables) such as name and marks.  4. Define a method getData() to accept student details.  5. Define a method displayData() to display the entered details.  6. Create the main() method.  7. Inside the main() method, create an object of the Student class.  8. Call the getData() method using the object.  9. Call the displayData() method using the object.  10. Stop the program.  FOR PROBLEM STATEMENT 2:  1. Start the program.  2. Define a class named BankAccount.  3. Declare data members such as accountHolderName, accountNumber, and balance.  4. Create a method getAccountDetails() to accept account information.  5. Create a method deposit() to add money to the account balance.  6. Create a method withdraw() to deduct money from the account balance.  7. Create a method displayDetails() to display the account information and current  balance.  8. Define the main() method.  9. Create an object of the BankAccount class inside the main() method.  10. Call the methods using the object to perform banking operations.  11. Stop the program. | | | |
| Code: PROBLEM STATEMENT 1 :  class Student {  String name;  int marks;  void setData(String n, int a) {  name = n;  age = a;  }  void displayData() {  System.out.println("Student Name: " + name);  System.out.println("Student Age: " + age);  }  }  public class Main {  public static void main(String[] args) {  Student s1 = new Student();  s1.setData("Kartik", 20);  s1.displayData();  }  }  PROBLEM STATEMENT 2 :  class BankAccount {  String accountHolderName;  int accountNumber;  double balance;  void getAccountDetails(String name, int accNo, double bal) {  accountHolderName = name;  accountNumber = accNo;  balance = bal;  }  void deposit(double amount) {  balance = balance + amount;  }  void withdraw(double amount) {  if (amount <= balance) {  balance = balance - amount;  } else {  System.out.println("Insufficient Balance");  }  }  void displayDetails() {  System.out.println("Account Holder Name: " + accountHolderName);  System.out.println("Account Number: " + accountNumber);  System.out.println("Current Balance: " + balance);  }  }  public class Main {  public static void main(String[] args) {  BankAccount b1 = new BankAccount();  b1.getAccountDetails("Kartik", 101, 5000);  b1.deposit(2000);  b1.withdraw(1000);  b1.displayDetails();  }  } | | | |
| Output:  FOR PROBLEM STATEMENT 1:  FOR PROBLEM STATEMENT 2: | | | |
| Output Analysis | | **FOR PROBLEM STATEMENT 1 - Student Class Program**  **When the program is executed, it asks the user to enter student details such as name and marks. These values are stored in the data members of the Student class through an input method. After the details are entered, the display method is called using the student object, and the stored information is printed on the screen. The output confirms that the object successfully holds the data and that the methods of the class are working properly. This demonstrates how a class serves as a blueprint, how an object stores real-world data, and how methods are used to access and display that data. The correct display of student details verifies the proper interaction between class, object, and methods.**  **FOR PROBLEM STATEMENT 2 –**  **When the BankAccount program runs, it creates an object containing the account holder’s name, account number, and initial balance. After that, the deposit method increases the account balance when a deposit amount is entered, and the updated balance is shown in the output. The withdraw method checks whether the account has sufficient balance before deducting money. If enough balance is available, the amount is withdrawn and the new balance is displayed; otherwise, an insufficient balance message appears. Finally, the display method prints all account details along with the final balance. The output proves that the object maintains its own data and that methods can change and display this data correctly. This clearly illustrates the practical use of class, object, and methods in Java.** | | | |
| Link of student Github profile where lab assignment has been uploaded | | **https://github.com/Kartikpatil1905/JAVA-PRACT-2/blob/main/README.md** | | | |
| Conclusion | | **In the Student class program, the correct input and display of student details show how a class defines the structure of data and how an object stores and manages that data through method calls. In the BankAccount program, the deposit and withdrawal operations confirm that methods can modify object data while maintaining logical conditions such as balance checking. The display of updated information in both cases proves that objects retain their state and respond correctly to method execution. Overall, the outputs validate the practical implementation of object-oriented programming principles and highlight how classes and objects are used to model real-world entities in Java programs.** | | | |
| Plag Report (Similarity index < 12%) | |  | | | |
| Date | | **31-01-2026** | | | |

****