C-DAC Mumbai
OOPJ Lab Assignment-6

Answers

Problem 1: Counter for Cups
Scenario: You are keeping track of how many cups of tea are prepared in your home.
Requirements:
1. Create a class TeaCup with instance variable: teaType (String).
2. Create a static variable totalCups to count all cups created.
3. Constructor should initialize teaType and increment totalCups.
4. Create getter for teaType.
5. Create a static method showTotalCups() to print total cups.
Input Example:
Cup1: teaType = "Masala Tea"
Cup2: teaType = "Green Tea"
Cup3: teaType = "Ginger Tea"
Expected Output:
Cup1 type: Masala Tea
Cup2 type: Green Tea
Cup3 type: Ginger Tea
Total cups made: 3

```java
public class TeaCup {
    private String teaType;
    private static int totalCups = 0;

    public TeaCup(String teaType) {
        this.teaType = teaType;
        totalCups++;
    }

    public String getTeaType() {
        return teaType;
    }

    public static void showTotalCups() {
        System.out.println("Total cups made: " + totalCups);
    }

    public static void main(String[] args) {
        TeaCup cup1 = new TeaCup("Masala Tea");
        TeaCup cup2 = new TeaCup("Green Tea");
        TeaCup cup3 = new TeaCup("Ginger Tea");

        System.out.println("Cup1 type: " + cup1.getTeaType());
        System.out.println("Cup2 type: " + cup2.getTeaType());
        System.out.println("Cup3 type: " + cup3.getTeaType());
        TeaCup.showTotalCups();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac TeaCup.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java TeaCup
Cup1 type: Masala Tea
Cup2 type: Green Tea
Cup3 type: Ginger Tea
Total cups made: 3
```

Problem 2: Simple Mobile Tracker
Scenario: A shop wants to count how many mobiles are added to their inventory.
Requirements:
1. Create a class Mobile with instance variable: model (String).
2. Create a static variable totalMobiles to count total mobiles added.
3. Constructor should initialize model and increment totalMobiles.
4. Create a getter for model.
5. Create a static method showTotalMobiles() to print total mobiles.
Input Example:
Mobile1: model = "Samsung Galaxy M32"
Mobile2: model = "Redmi Note 12"
Expected Output:
Mobile1 model: Samsung Galaxy M32
Mobile2 model: Redmi Note 12
Total mobiles in stock: 2

```java
public class Mobile {
    private String model;
    private static int totalMobiles = 0;

    public Mobile(String model) {
        this.model = model;
        totalMobiles++;
    }

    public String getModel() {
        return model;
    }

    public static void showTotalMobiles() {
        System.out.println("Total mobiles in stock: " + totalMobiles);
    }

    public static void main(String[] args) {
        Mobile m1 = new Mobile("Samsung Galaxy M32");
        Mobile m2 = new Mobile("Redmi Note 12");

        System.out.println("Mobile1 model: " + m1.getModel());
        System.out.println("Mobile2 model: " + m2.getModel());
        Mobile.showTotalMobiles();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac Mobile.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java Mobile
Mobile1 model: Samsung Galaxy M32
Mobile2 model: Redmi Note 12
Total mobiles in stock: 2
```

Problem 3: Library Book Tracker
Scenario: A library in Delhi wants to track how many books are issued in total and details of each book.
Requirements:
1. Create a Book class with instance variables: title (String), author (String), issued (boolean).
2. Create static variable totalIssuedBooks to keep track of the total number of books issued.
3. Create a constructor to initialize the book details.
4. Create getters and setters for all instance variables.
5. Create a static method showTotalIssued() to print total issued books.
6. Write a main class to create **3 books**, issue some of them (issued = true), and show total issued books.

Input Example:
Book1: "Harry Potter", Author: "J.K. Rowling", Issued: true
Book2: "Five Point Someone", Author: "Chetan Bhagat", Issued: false
Book3: "Rich Dad Poor Dad", Author: "Robert Kiyosaki", Issued: true
Expected Output:
Book1 issued? true
Book2 issued? false
Book3 issued? true
Total books issued: 2

```java
public class Book {
    private String title;
    private String author;
    private boolean issued;
    private static int totalIssuedBooks = 0;

    public Book(String title, String author, boolean issued) {
        this.title = title;
        this.author = author;
        this.issued = issued;
        if (issued) totalIssuedBooks++;
    }

    public boolean isIssued() {
        return issued;
    }

    public static void showTotalIssued() {
        System.out.println("Total books issued: " + totalIssuedBooks);
    }

    public static void main(String[] args) {
        Book b1 = new Book("Harry Potter", "J.K. Rowling", true);
        Book b2 = new Book("Five Point Someone", "Chetan Bhagat", false);
        Book b3 = new Book("Rich Dad Poor Dad", "Robert Kiyosaki", true);

        System.out.println("Book1 issued? " + b1.isIssued());
        System.out.println("Book2 issued? " + b2.isIssued());
        System.out.println("Book3 issued? " + b3.isIssued());
        Book.showTotalIssued();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac Book.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java Book
Book1 issued? true
Book2 issued? false
Book3 issued? true
Total books issued: 2
```

Problem 4: Employee Salary Manager

Scenario: A company in Bengaluru wants to maintain employee details and give a bonus to employees who have worked more than 5 years.

Requirements:

1. Create a class Employee with instance variables: name (String), salary (double), yearsOfService (int).
2. Create static variable totalEmployees to store the number of employees created.
3. Constructor should initialize all instance variables and increment totalEmployees.
4. Create getters and setters for all instance variables.
5. Create a method calculateBonus() that returns 5% of salary if yearsOfService > 5, otherwise 0.
6. Create a static method showTotalEmployees() to print total employees created.
7. Write a main class to create **3 employees**, print their bonuses, and print total employees.

Input Example:

Employee1: Name: "Ravi", Salary: 150000, Years of Service: 6
Employee2: Name: "Anita", Salary: 120000, Years of Service: 3
Employee3: Name: "Suresh", Salary: 100000, Years of Service: 5

Expected Output:

Employee Ravi Bonus: 7500.0
Employee Anita Bonus: 0.0
Employee Suresh Bonus: 0.0
Total employees: 3

```java
public class Employee {
    private String name;
    private double salary;
    private int yearsOfService;
    private static int totalEmployees = 0;

    public Employee(String name, double salary, int yearsOfService) {
        this.name = name;
        this.salary = salary;
        this.yearsOfService = yearsOfService;
        totalEmployees++;
    }

    public double calculateBonus() {
        return yearsOfService > 5 ? salary * 0.05 : 0.0;
    }

    public static void showTotalEmployees() {
        System.out.println("Total employees: " + totalEmployees);
    }

    public static void main(String[] args) {
        Employee e1 = new Employee("Ravi", 150000, 6);
        Employee e2 = new Employee("Anita", 120000, 3);
        Employee e3 = new Employee("Suresh", 100000, 5);

        System.out.println("Employee Ravi Bonus: " + e1.calculateBonus());
        System.out.println("Employee Anita Bonus: " + e2.calculateBonus());
        System.out.println("Employee Suresh Bonus: " + e3.calculateBonus());
        Employee.showTotalEmployees();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac Employee.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java Employee
Employee Ravi Bonus: 7500.0
Employee Anita Bonus: 0.0
Employee Suresh Bonus: 0.0
Total employees: 3
```

Problem 5: Student Marks Calculator
Scenario: A school in Mumbai wants to calculate marks of students and also maintain total students in the class.
Requirements:
1. Create a class Student with instance variables: name (String), marks (int).
2. Create static variable totalStudents to count total number of students.
3. Constructor to initialize student details and increment totalStudents.
4. Getter and Setter for marks.
5. Method isPassed() returns true if marks >= 35, false otherwise.
6. Static method showTotalStudents() prints total students.
7. In main class, create **3 students**, check if they passed, and show total students.

Input Example:
Student1: Name: "Rahul", Marks: 78
Student2: Name: "Pooja", Marks: 34
Student3: Name: "Amit", Marks: 65
Expected Output:
Student Rahul Passed? true
Student Pooja Passed? false
Student Amit Passed? true
Total students: 3

```java
public class Student {
    private String name;
    private int marks;
    private static int totalStudents = 0;

    public Student(String name, int marks) {
        this.name = name;
        this.marks = marks;
        totalStudents++;
    }

    public boolean isPassed() {
        return marks >= 35;
    }

    public static void showTotalStudents() {
        System.out.println("Total students: " + totalStudents);
    }

    public static void main(String[] args) {
        Student s1 = new Student("Rahul", 78);
        Student s2 = new Student("Pooja", 34);
        Student s3 = new Student("Amit", 65);

        System.out.println("Student Rahul Passed? " + s1.isPassed());
        System.out.println("Student Pooja Passed? " + s2.isPassed());
        System.out.println("Student Amit Passed? " + s3.isPassed());
        Student.showTotalStudents();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac Student.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java Student
Student Rahul Passed? true
Student Pooja Passed? false
Student Amit Passed? true
Total students: 3
```

Problem 6: Indian Railway Ticket Booking
Scenario: You are building a mini ticket booking system. A passenger can book a ticket either by giving **name and age** or **name, age, and seat type**. The system should also count the **total tickets booked** using a static counter.
Tasks:
1. Create a Passenger class.
2. Implement **two constructors** (constructor overloading):

Constructor 1 → Passenger(String name, int age)
Constructor 2 → Passenger(String name, int age, String seatType)
3. Use a **static counter** to keep track of **total passengers booked**.
4. Print passenger details and total passengers.

Input Example:
Passenger1: "Ravi", 25
Passenger2: "Anita", 30, "AC Sleeper"
Passenger3: "Suresh", 40
Expected Output:
Passenger1: Name: Ravi, Age: 25, Seat: General
Passenger2: Name: Anita, Age: 30, Seat: AC Sleeper
Passenger3: Name: Suresh, Age: 40, Seat: General
Total Passengers Booked: 3

```java
public class Passenger {
    private String name;
    private int age;
    private String seatType;
    private static int totalPassengers = 0;

    public Passenger(String name, int age) {
        this.name = name;
        this.age = age;
        this.seatType = "General";
        totalPassengers++;
    }

    public Passenger(String name, int age, String seatType) {
        this.name = name;
        this.age = age;
        this.seatType = seatType;
        totalPassengers++;
    }

    public static void showTotalPassengers() {
        System.out.println("Total Passengers Booked: " + totalPassengers);
    }

    public static void main(String[] args) {
        Passenger p1 = new Passenger("Ravi", 25);
        Passenger p2 = new Passenger("Anita", 30, "AC Sleeper");
        Passenger p3 = new Passenger("Suresh", 40);

        System.out.println("Passenger1: Name: " + p1.name + ", Age: " + p1.age + ", Seat: " + p1.seatType);
        System.out.println("Passenger2: Name: " + p2.name + ", Age: " + p2.age + ", Seat: " + p2.seatType);
        System.out.println("Passenger3: Name: " + p3.name + ", Age: " + p3.age + ", Seat: " + p3.seatType);
        Passenger.showTotalPassengers();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac Passenger.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java Passenger
Passenger1: Name: Ravi, Age: 25, Seat: General
Passenger2: Name: Anita, Age: 30, Seat: AC Sleeper
Passenger3: Name: Suresh, Age: 40, Seat: General
Total Passengers Booked: 3
```

Problem 7: Indian Movie Ticket Booking
Scenario: A cinema hall offers **Normal** and **Premium** tickets. A customer can book **just name** or **name with ticket type**. Keep track of **total tickets sold** using a static counter.
Tasks:
1. Create a Customer class.
2. Implement **two constructors**:

Constructor 1 → Customer(String name)
Constructor 2 → Customer(String name, String ticketType)
3. **Static counter** to track tickets sold.
4. Print customer details and total tickets sold.

Input Example:
Customer1: "Rahul"
Customer2: "Pooja", "Premium"
Customer3: "Amit"
Expected Output:
Customer1: Name: Rahul, Ticket: Normal
Customer2: Name: Pooja, Ticket: Premium
Customer3: Name: Amit, Ticket: Normal
Total Tickets Sold: 3

```java
public class Customer {
    private String name;
    private String ticketType;
    private static int totalTickets = 0;

    public Customer(String name) {
        this.name = name;
        this.ticketType = "Normal";
        totalTickets++;
    }

    public Customer(String name, String ticketType) {
        this.name = name;
        this.ticketType = ticketType;
        totalTickets++;
    }

    public static void showTotalTickets() {
        System.out.println("Total Tickets Sold: " + totalTickets);
    }

    public static void main(String[] args) {
        Customer c1 = new Customer("Rahul");
        Customer c2 = new Customer("Pooja", "Premium");
        Customer c3 = new Customer("Amit");

        System.out.println("Customer1: Name: " + c1.name + ", Ticket: " + c1.ticketType);
        System.out.println("Customer2: Name: " + c2.name + ", Ticket: " + c2.ticketType);
        System.out.println("Customer3: Name: " + c3.name + ", Ticket: " + c3.ticketType);
        Customer.showTotalTickets();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac Customer.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java Customer
Customer1: Name: Rahul, Ticket: Normal
Customer2: Name: Pooja, Ticket: Premium
Customer3: Name: Amit, Ticket: Normal
Total Tickets Sold: 3
```

Problem 8: Bank Account Initialization

Scenario: A bank wants to **initialize the interest rate** for all accounts **once** when the system starts. Each account has **account holder name**, **balance**, and **interest rate**. Students should practice **static blocks** for initialization and **setters/getters** to modify and access account details.

Tasks:

1. Create a BankAccount class.
2. Use a **static block** to initialize **interest rate** to 4%.
3. Create instance variables: name (String) and balance (double).
4. Create **setters and getters** for name and balance.
5. Print account details including interest rate.

Input Example:

Account1: Name="Rohit", Balance=5000
Account2: Name="Priya", Balance=15000

Expected Output:

Bank Interest Rate Initialized: 4.0%
Account1: Name=Rohit, Balance=5000.0, Interest Rate=4.0%
Account2: Name=Priya, Balance=15000.0, Interest Rate=4.0%

```java
public class BankAccount {
    private String name;
    private double balance;
    private static double interestRate;

    static {
        interestRate = 4.0;
        System.out.println("Bank Interest Rate Initialized: " + interestRate + "%");
    }

    public BankAccount(String name, double balance) {
        this.name = name;
        this.balance = balance;
    }

    public static void main(String[] args) {
        BankAccount a1 = new BankAccount("Rohit", 5000);
        BankAccount a2 = new BankAccount("Priya", 15000);

        System.out.println("Account1: Name=" + a1.name + ", Balance=" + a1.balance + ", Interest Rate=" + interestRate + "%");
        System.out.println("Account2: Name=" + a2.name + ", Balance=" + a2.balance + ", Interest Rate=" + interestRate + "%");
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac BankAccount.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java BankAccount
Bank Interest Rate Initialized: 4.0%
Account1: Name=Rohit, Balance=5000.0, Interest Rate=4.0%
Account2: Name=Priya, Balance=15000.0, Interest Rate=4.0%
```

Problem 9: School Fee System

Scenario: A school wants to **initialize the tuition fee** for all students once at program start. Each student has **name** and **class**. Use **static blocks** to set the fee and **setters/getters** to update/access student information.

Tasks:

1. Create a Student class.
2. Use a **static block** to initialize **tuitionFee** to 30000.
3. Create instance variables: name (String) and className (String).
4. Create **setters and getters** for name and className.
5. Print student details including tuition fee.

Input Example:

Student1: Name="Anjali", Class="10th"
Student2: Name="Vikram", Class="12th"

Expected Output:

School Tuition Fee Initialized: 30000
Student1: Name=Anjali, Class=10th, Tuition Fee=30000
Student2: Name=Vikram, Class=12th, Tuition Fee=30000

```java
public class SchoolStudent {
    private String name;
    private String className;
    private static int tuitionFee;

    static {
        tuitionFee = 30000;
        System.out.println("School Tuition Fee Initialized: " + tuitionFee);
    }

    public SchoolStudent(String name, String className) {
        this.name = name;
        this.className = className;
    }

    public static void main(String[] args) {
        SchoolStudent s1 = new SchoolStudent("Anjali", "10th");
        SchoolStudent s2 = new SchoolStudent("Vikram", "12th");

        System.out.println("Student1: Name=" + s1.name + ", Class=" + s1.className + ", Tuition Fee=" + tuitionFee);
        System.out.println("Student2: Name=" + s2.name + ", Class=" + s2.className + ", Tuition Fee=" + tuitionFee);
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac SchoolStudent.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java SchoolStudent
School Tuition Fee Initialized: 30000
Student1: Name=Anjali, Class=10th, Tuition Fee=30000
Student2: Name=Vikram, Class=12th, Tuition Fee=30000
```

Problem 10: Student Marks Checker
Scenario: Create a Student class with rollNo, name, and marks.
- Use a **parameterized constructor** to initialize all fields.
- Create a **getter and setter** for marks.
- In main, create **one student**, update marks using setter, and print student details.

```java
public class StudentMarksChecker {
    private int rollNo;
    private String name;
    private int marks;

    public StudentMarksChecker(int rollNo, String name, int marks) {
        this.rollNo = rollNo;
        this.name = name;
        this.marks = marks;
    }

    public int getMarks() {
        return marks;
    }

    public void setMarks(int marks) {
        this.marks = marks;
    }

    public static void main(String[] args) {
        StudentMarksChecker s = new StudentMarksChecker(1, "Rahul", 45);
        s.setMarks(55);
        System.out.println("RollNo: " + s.rollNo + ", Name: " + s.name + ", Marks: " + s.getMarks());
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac StudentMarksChecker.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java StudentMarksChecker
RollNo: 1, Name: Rahul, Marks: 55
```

Problem 11: Student Grade Calculator
Scenario: Extend previous problem. Add method calculateGrade() which returns:
- "A" if marks ≥ 80
- "B" if marks ≥ 60
- "C" if marks ≥ 40
- "Fail" otherwise
- Create **2 students**, print marks and grades.

```java
public class StudentGradeCalculator {
    private int rollNo;
    private String name;
    private int marks;

    public StudentGradeCalculator(int rollNo, String name, int marks) {
        this.rollNo = rollNo;
        this.name = name;
        this.marks = marks;
    }

    public String calculateGrade() {
        if (marks >= 80) return "A";
        else if (marks >= 60) return "B";
        else if (marks >= 40) return "C";
        else return "Fail";
    }

    public static void main(String[] args) {
        StudentGradeCalculator s1 = new StudentGradeCalculator(1, "Rahul", 85);
        StudentGradeCalculator s2 = new StudentGradeCalculator(2, "Priya", 55);

        System.out.println("Student: " + s1.name + ", Marks: " + s1.marks + ", Grade: " + s1.calculateGrade());
        System.out.println("Student: " + s2.name + ", Marks: " + s2.marks + ", Grade: " + s2.calculateGrade());
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac StudentGradeCalculator.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java StudentGradeCalculator
Student: Rahul, Marks: 85, Grade: A
Student: Priya, Marks: 55, Grade: C
```

Problem 12: Bank Account Basic Info

Scenario: Create BankAccount class with accountHolder and balance.

- Use **parameterized constructor** to initialize account.
- Create **getter and setter** for balance.
- In main, create **one account** and display details.

```java
public class BankAccountBasicInfo {
    private String accountHolder;
    private double balance;

    public BankAccountBasicInfo(String accountHolder, double balance) {
        this.accountHolder = accountHolder;
        this.balance = balance;
    }

    public double getBalance() {
        return balance;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }

    public static void main(String[] args) {
        BankAccountBasicInfo acc = new BankAccountBasicInfo("Ravi", 10000);
        System.out.println("Account Holder: " + acc.accountHolder + ", Balance: " + acc.getBalance());
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac BankAccountBasicInfo.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java BankAccountBasicInfo
Account Holder: Ravi, Balance: 10000.0
```

Problem 13: Bank Deposit & Withdrawal
Scenario: Extend previous problem. Add methods:
- deposit(double amount) → adds to balance
- withdraw(double amount) → subtracts from balance
- Create **two accounts**, perform deposit/withdraw, display updated balance.

```java
public class BankDepositWithdrawal {
    private String accountHolder;
    private double balance;

    public BankDepositWithdrawal(String accountHolder, double balance) {
        this.accountHolder = accountHolder;
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance += amount;
    }

    public void withdraw(double amount) {
        balance -= amount;
    }

    public double getBalance() {
        return balance;
    }

    public static void main(String[] args) {
        BankDepositWithdrawal acc1 = new BankDepositWithdrawal("Ravi", 10000);
        BankDepositWithdrawal acc2 = new BankDepositWithdrawal("Priya", 5000);

        acc1.deposit(2000);
        acc2.withdraw(1000);

        System.out.println("Account1: " + acc1.accountHolder + ", Balance: " + acc1.getBalance());
        System.out.println("Account2: " + acc2.accountHolder + ", Balance: " + acc2.getBalance());
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac BankDepositWithdrawal.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java BankDepositWithdrawal
Account1: Ravi, Balance: 12000.0
Account2: Priya, Balance: 4000.0
```

Problem 14: Bank Name Display
Scenario: Add a **static variable** bankName = "CDAC Bank" and **static method** displayBankName() to BankAccount.
- Call displayBankName() from main.
- Create **one account** to verify instance creation.

```java
public class BankNameDisplay {
    private String accountHolder;
    private double balance;
    private static String bankName = "CDAC Bank";

    public BankNameDisplay(String accountHolder, double balance) {
        this.accountHolder = accountHolder;
        this.balance = balance;
    }

    public static void displayBankName() {
        System.out.println("Bank Name: " + bankName);
    }

    public static void main(String[] args) {
        displayBankName();
        BankNameDisplay acc = new BankNameDisplay("Ravi", 12000);
        System.out.println("Account Holder: " + acc.accountHolder + ", Balance: " + acc.balance);
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac BankNameDisplay.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java BankNameDisplay
Bank Name: CDAC Bank
```

Problem 15: Employee Auto-ID Generator

Scenario: Create Employee class with id, name, basicSalary.

- Add **static counter** starting from 1001 for IDs.
- Default constructor → name = "Unknown", salary = 20000
- Parameterized constructor → accept name and salary
- Getter for all variables
- Create **2 employees** and display their IDs, names, salary.

```java
public class TicketBookingSystem {
    private String passengerName;
    private int ticketNo;
    private static int counter = 5001;

    public TicketBookingSystem(String passengerName) {
        this.passengerName = passengerName;
        this.ticketNo = counter++;
    }

    public void displayTicket() {
        System.out.println("Ticket No: " + ticketNo + ", Passenger: " + passengerName);
    }

    public static void main(String[] args) {
        TicketBookingSystem t1 = new TicketBookingSystem("Rahul");
        TicketBookingSystem t2 = new TicketBookingSystem("Priya");
        TicketBookingSystem t3 = new TicketBookingSystem("Amit");

        t1.displayTicket();
        t2.displayTicket();
        t3.displayTicket();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac EmployeeAutoIDGenerator.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java EmployeeAutoIDGenerator
ID: 1001, Name: Ravi, Salary: 30000.0
ID: 1002, Name: Unknown, Salary: 20000.0
```

Problem 16: Employee Net Salary
Scenario: Extend previous problem. Add method calculateNetSalary():
- Add 10% HRA, 5% DA, deduct 2% PF from basicSalary
- Print net salary for **2 employees**

```java
public class EmployeeNetSalary {
    private int id;
    private String name;
    private double basicSalary;
    private static int counter = 2001;

    public EmployeeNetSalary(String name, double basicSalary) {
        this.id = counter++;
        this.name = name;
        this.basicSalary = basicSalary;
    }

    public double calculateNetSalary() {
        double hra = basicSalary * 0.10;
        double da = basicSalary * 0.05;
        double pf = basicSalary * 0.02;
        return basicSalary + hra + da - pf;
    }

    public static void main(String[] args) {
        EmployeeNetSalary e1 = new EmployeeNetSalary("Ravi", 30000);
        EmployeeNetSalary e2 = new EmployeeNetSalary("Anita", 40000);

        System.out.println("Employee: " + e1.name + ", Net Salary: " + e1.calculateNetSalary());
        System.out.println("Employee: " + e2.name + ", Net Salary: " + e2.calculateNetSalary());
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac EmployeeNetSalary.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java EmployeeNetSalary
Employee: Ravi, Net Salary: 33900.0
Employee: Anita, Net Salary: 45200.0
```

Problem 17: Library Book Addition
Scenario: Create Book class with bookId, title, author.
- Constructor + Getters/Setters
- Create Library class with libraryName and **static totalBooks**
- Method addBook(Book b) → increments totalBooks
- Method displayTotalBooks() → prints totalBooks
- Add **2 books** to library and display total books

```java
class Book {
    private int bookId;
    private String title;
    private String author;

    public Book(int bookId, String title, String author) {
        this.bookId = bookId;
        this.title = title;
        this.author = author;
    }
}

public class LibraryBookAddition {
    private String libraryName;
    private static int totalBooks = 0;

    public LibraryBookAddition(String libraryName) {
        this.libraryName = libraryName;
    }

    public void addBook(Book b) {
        totalBooks++;
    }

    public void displayTotalBooks() {
        System.out.println("Total Books: " + totalBooks);
    }

    public static void main(String[] args) {
        LibraryBookAddition lib = new LibraryBookAddition("City Library");
        Book b1 = new Book(1, "Harry Potter", "J.K. Rowling");
        Book b2 = new Book(2, "Wings of Fire", "A.P.J. Abdul Kalam");

        lib.addBook(b1);
        lib.addBook(b2);
        lib.displayTotalBooks();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac LibraryBookAddition.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java LibraryBookAddition
Total Books: 2
```

Problem 18: Vehicle Registration – Static Counter
Scenario: Create Vehicle class with regNo, ownerName, vehicleType.
- Static variable: vehicleCount
- Constructor → auto-generate regNo as "MH-2025-" + vehicleCount
- Getter methods for all fields
- Create **2 vehicles**, display registration details

```java
public class VehicleRegistrationCounter {
    private String regNo;
    private String ownerName;
    private String vehicleType;
    private static int vehicleCount = 0;

    public VehicleRegistrationCounter(String ownerName, String vehicleType) {
        this.ownerName = ownerName;
        this.vehicleType = vehicleType;
        vehicleCount++;
        this.regNo = "MH-2025-" + vehicleCount;
    }

    public static void main(String[] args) {
        VehicleRegistrationCounter v1 = new VehicleRegistrationCounter("Ravi", "Car");
        VehicleRegistrationCounter v2 = new VehicleRegistrationCounter("Anita", "Bike");

        System.out.println("RegNo: " + v1.regNo + ", Owner: " + v1.ownerName + ", Type: " + v1.vehicleType);
        System.out.println("RegNo: " + v2.regNo + ", Owner: " + v2.ownerName + ", Type: " + v2.vehicleType);
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac VehicleRegistrationCounter.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java VehicleRegistrationCounter
RegNo: MH-2025-1, Owner: Ravi, Type: Car
RegNo: MH-2025-2, Owner: Anita, Type: Bike
```

Problem 19: Vehicle Registration – Static Block
Scenario: Add a **static block** to Vehicle class:
- Print "Welcome to CDAC Vehicle Registration Portal" when class loads
- Verify that the message prints **only once** when multiple vehicles are created

```java
public class VehicleRegistrationBlock {
    private String regNo;
    private String ownerName;
    private String vehicleType;
    private static int vehicleCount = 0;

    static {
        System.out.println("Welcome to CDAC Vehicle Registration Portal");
    }

    public VehicleRegistrationBlock(String ownerName, String vehicleType) {
        this.ownerName = ownerName;
        this.vehicleType = vehicleType;
        vehicleCount++;
        this.regNo = "MH-2025-" + vehicleCount;
    }

    public static void main(String[] args) {
        VehicleRegistrationBlock v1 = new VehicleRegistrationBlock("Ravi", "Car");
        VehicleRegistrationBlock v2 = new VehicleRegistrationBlock("Priya", "Scooter");

        System.out.println("RegNo: " + v1.regNo + ", Owner: " + v1.ownerName + ", Type: " + v1.vehicleType);
        System.out.println("RegNo: " + v2.regNo + ", Owner: " + v2.ownerName + ", Type: " + v2.vehicleType);
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac VehicleRegistrationBlock.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java VehicleRegistrationBlock
Welcome to CDAC Vehicle Registration Portal
RegNo: MH-2025-1, Owner: Ravi, Type: Car
RegNo: MH-2025-2, Owner: Priya, Type: Scooter
```

Problem 20: Ticket Booking System
Question: Create a class Ticket with:
- passengerName (instance)
- ticketNo (instance, auto-generated using a static counter starting from 5001)
- Constructor to accept passengerName
- Method displayTicket() to show ticket details

Task: Create 3 tickets and display their details.
Sample Input:
Passenger 1: Rahul
Passenger 2: Priya
Passenger 3: Amit
Sample Output:
Ticket No: 5001, Passenger: Rahul
Ticket No: 5002, Passenger: Priya
Ticket No: 5003, Passenger: Amit

```java
public class TicketBookingSystem {
    private String passengerName;
    private int ticketNo;
    private static int counter = 5001;

    public TicketBookingSystem(String passengerName) {
        this.passengerName = passengerName;
        this.ticketNo = counter++;
    }

    public void displayTicket() {
        System.out.println("Ticket No: " + ticketNo + ", Passenger: " + passengerName);
    }

    public static void main(String[] args) {
        TicketBookingSystem t1 = new TicketBookingSystem("Rahul");
        TicketBookingSystem t2 = new TicketBookingSystem("Priya");
        TicketBookingSystem t3 = new TicketBookingSystem("Amit");

        t1.displayTicket();
        t2.displayTicket();
        t3.displayTicket();
    }
}
```

```
PS C:\Users\baenu\Test\OOPJ Assignment 6> javac TicketBookingSystem.java
PS C:\Users\baenu\Test\OOPJ Assignment 6> java TicketBookingSystem
Ticket No: 5001, Passenger: Rahul
Ticket No: 5002, Passenger: Priya
Ticket No: 5003, Passenger: Amit
```