CodInClub
Powered By BridgeLabz

Best Programming Practice

1. Use **static** for shared values and utility methods to reduce memory usage and avoid redundancy.
2. Leverage **this** to avoid ambiguity when initializing attributes.
3. Declare **final** variables for identifiers or constants that should remain unchanged.
4. Use **instanceof** for safe type-checking and to prevent runtime errors during typecasting.

Sample Program 1: Bank Account System

Create a BankAccount class with the following features:

1. **Static**:
    - A static variable bankName is shared across all accounts.
    - A static method getTotalAccounts() to display the total number of accounts.
2. **This**:
    - Use this to resolve ambiguity in the constructor when initializing accountHolderName and accountNumber.
3. **Final**:
    - Use a final variable accountNumber to ensure it cannot be changed once assigned.
4. **Instanceof**:
    - Check if an account object is an instance of the BankAccount class before displaying its details.

Sample Program 2: Library Management System

Create a Book class to manage library books with the following features:

1. **Static**:
    - A static variable libraryName shared across all books.
    - A static method displayLibraryName() to print the library name.
2. **This**:
    - Use this to initialize title, author, and isbn in the constructor.
3. **Final**:
    - Use a final variable isbn to ensure the unique identifier of a book cannot be changed.
4. **Instanceof**:
    - Verify if an object is an instance of the Book class before displaying its details.

Sample Program 3: Employee Management System

Design an Employee class with the following features:

1. **Static**:
    - A static variable companyName shared by all employees.
    - A static method displayTotalEmployees() to show the total number of employees.
2. **This**:
    - Use this to initialize name, id, and designation in the constructor.

3. **Final**:
    - Use a final variable id for the employee ID, which cannot be modified after assignment.
4. **Instanceof**
    - Check if a given object is an instance of the Employee class before printing the employee details.

Sample Program 4: Shopping Cart System

Create a Product class to manage shopping cart items with the following features:

1. **Static**:
    - A static variable discount shared by all products.
    - A static method updateDiscount() to modify the discount percentage.
2. **This**:
    - Use this to initialize productName, price, and quantity in the constructor.
3. **Final**:
    - Use a final variable productID to ensure each product has a unique identifier that cannot be changed.
4. **Instanceof**:
    - Validate whether an object is an instance of the Product class before processing its details.

Sample Program 5: University Student Management

Create a Student class to manage student data with the following features:

1. **Static**:
    - A static variable universityName shared across all students.
    - A static method displayTotalStudents() to show the number of students enrolled.
2. **This**:
    - Use this in the constructor to initialize name, rollNumber, and grade.
3. **Final**:
    - Use a final variable rollNumber for each student that cannot be changed.
4. **Instanceof**:
    - Check if a given object is an instance of the Student class before performing operations like displaying or updating grades.

Sample Program 6: Vehicle Registration System

Create a Vehicle class with the following features:

1. **Static**:
    - A static variable registrationFee common for all vehicles.
    - A static method updateRegistrationFee() to modify the fee.
2. **This**:
    - Use this to initialize ownerName, vehicleType, and registrationNumber in the constructor.
3. **Final**:
    - Use a final variable registrationNumber to uniquely identify each vehicle.

4. **Instanceof**:
    - Check if an object belongs to the Vehicle class before displaying its registration
    - details.

Sample Program 7: Hospital Management System

Create a Patient class with the following features:

1. **Static**:
    - A static variable hospitalName shared among all patients.
    - A static method getTotalPatients() to count the total patients admitted.
2. **This**:
    - Use this to initialize name, age, and ailment in the constructor.
3. **Final**:
    - Use a final variable patientID to uniquely identify each patient.
4. **Instanceof**:
    - Check if an object is an instance of the Patient class before displaying its details.