Java_Assignment-03

Instructions:

- Complete each assignment separately and test the output.
- Follow good coding practices (use meaningful variable and method names).
- Submit the completed programs before the deadline.

---

1. Library System

- Create a class named **Book** with:
  - Variables: `title`, `author`, `pages`
  - Method **displayBookDetails()** to print book details.
- In the main program, create an object of **Book**, assign values, and print the details.

Example Output:

```
Book Title: The Alchemist
Author: Paulo Coelho
Pages: 208
```

---

2. Inventory Management

- Create a class **Product** with:
  - Variables: `productName`, `price`, `quantity`
  - Method **displayProduct()** to print product details.
- In the main program, create multiple product objects and display their details.

Example Output:

```
Product: Laptop
Price: 55000
Quantity: 10

Product: Smartphone
Price: 30000
Quantity: 25
```

---

3. Using Objects of Another Class (Bank and Customer Example)

- Create a **Bank** class with:
  - Variables: `bankName`, `branchCode`
  - Method **displayBankDetails()** to print bank details.
- Create a **Customer** class with:
  - Variables: `customerName`, `accountNumber`
  - Method **displayCustomerDetails()**`
  - A **Bank** object inside the **Customer** class to associate a bank with a customer.
- In the main program, create a **Bank** object and link it to a **Customer** object.

Example Output:

```
Customer Name: Ravi
Account Number: 123456789
Bank: SBI
Branch Code: 1010
```

---

4. Implementing a Default Constructor

- Create a class **Student** with variables `name` and `age`.
- Define a **default constructor** that initializes `name` as "Unknown" and `age` as `0`.
- Create an object in the main program and display the initialized values.

Example Output:

```
Student Name: Unknown
Age: 0
```

5. Using a Parameterized Constructor

- Modify the **Student** class to include a **parameterized constructor** that takes `name` and `age` as parameters and initializes them.
- Create multiple objects with different values and display them.

Example Output:

```
Student Name: Rahul, Age: 20
Student Name: Priya, Age: 22
```

6. Multiple Constructors (Constructor Overloading)

- Modify the **Student** class to have:
  - A **default constructor** (sets name as "Unknown" and age as 0)
  - A **parameterized constructor** (sets name and age from arguments)
- Create objects using both constructors and print their values.

7. Implementing Encapsulation

- Create a class **BankAccount** with:
  - Private variables: `accountNumber`, `balance`
  - Public methods: `deposit(amount)`, `withdraw(amount)`, `getBalance()`
- Use **getter and setter methods** to access and modify private variables.
- In the main program, create a bank account object, deposit and withdraw money, and display the balance.

Example Output:

```
Deposited 5000. Current Balance: 5000
```

```
Withdrew 2000. Current Balance: 3000
```

---

8. Encapsulation with Validation

- Modify the **BankAccount** class to include:
    - **Validation in the withdraw method** (Cannot withdraw more than the balance)
    - **Validation in deposit method** (Cannot deposit negative values)
- Test different deposit and withdrawal cases.

Example Output:

```
Deposited 5000. Current Balance: 5000
Invalid Withdrawal. Insufficient Balance!
Withdrew 3000. Current Balance: 2000
```

- If you have doubts, feel free to ask!

Happy coding! 🚀🔥
