Experiment No-5

Experiment name: Understanding Class and Object (main() Outside the Class)

Task 1: Create a BankAccount Class with Static Members

Task Name: Create a class BankAccount with the following attributes: accountNumber, accountHolderName, balance, Static variable: bankName = "Sonali Bank", Create a constructor to initialize account details, Create a static method showBankName() to display the bank name. Write a Main class where you create two accounts and display their information.

Objective:

- To understand the concepts of class, object, static members, and placing the main() method outside the class in Java.

Problem analysis:

- We need a BankAccount class to model real-world bank accounts.
- Each account has instance variables:
    - accountNumber (e.g., String or long)
    - accountHolderName (String)
    - balance (double)
- A static variable bankName shared by all objects (since all accounts belong to the same bank: *Sonali Bank*).
- A constructor to initialize instance variables.
- A static method showBankName() to display the bank name (does not require an object).
- A separate Main class (outside BankAccount) containing the main() method to:
    - Create two BankAccount objects.
    - Display bank name using the static method.
    - Optionally display individual account details.

Algorithm:

1. Start

2. Create a class **BankAccount**

3. Declare data members: accountNumber, accountHolderName, balance

4. Declare a static variable **bankName = "Sonali Bank"**

5. Create a constructor to initialize account details

6. Create a static method **showBankName()** to print the bank name

7. Create a **Main** class

8. Inside main(), call **BankAccount.showBankName()**

9. Create two objects of BankAccount

10. Display the account details

11. End

Source code:

```
class BankAccount {
    int accountNumber;
    String accountHolderName;
    double balance;
    static String bankName = "Sonali Bank";
    BankAccount(int accountNumber, String accountHolderName, double balance) {
        this.accountNumber = accountNumber;
        this.accountHolderName = accountHolderName;
        this.balance = balance;
    }
    void displayAccountInfo() {
        System.out.println("Bank Name: " + bankName);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Holder: " + accountHolderName);
        System.out.println("Balance: " + balance);
    }
    static void showBankName() {
        System.out.println("Bank Name: " + bankName);
    }
}
public class Main {
    public static void main(String[] args) {
        BankAccount acc1 = new BankAccount(251046, "Atik", 500);
        BankAccount acc2 = new BankAccount(251018, "Atif", 5000);
        acc1.displayAccountInfo();
        System.out.println();
        acc2.displayAccountInfo();
        BankAccount.showBankName();
    }
}
```

Output:

```
AppData\Roaming\Code\User\workspaceStorage\57
ain "
Bank Name: Sonali Bank
Account Number: 251046
Account Holder: Atik
Balance: 500.0

Bank Name: Sonali Bank
Account Number: 251018
Account Holder: Atif
Balance: 5000.0
Bank Name: Sonali Bank

D:\java lab report VS code>
```

Discussion:

In this experiment, a class BankAccount was created to represent real-life bank accounts. The static variable BankName belongs to the class, so it is shared by all objects. Therefore, it can be accessed using the class name without creating objects. The constructor initializes each account individually. Two objects demonstrate how multiple accounts can exist with shared static information. This experiment helps understand the difference between static members (class-level) and instance members (object-level).

Task 2: Create a LibraryBook Class Using this Keyword

Task Name: Create a class LibraryBook with the following attributes: title, author, bookID. Use this keyword in the constructor to initialize values,Create a method displayBookInfo() to show the book details.Write a Main class where you create books and display their information.

Objective:

- To understand and implement the use of the this keyword in Java constructors to differentiate between instance variables and parameters with the same name.

Problem analysis:

- Create a class named LibraryBook with three instance variables:
  - title (String)
  - author (String)
  - bookID (int)
- Use the this keyword inside the constructor to assign parameter values to the corresponding instance variables.
- Implement a method displayBookInfo() to print all book details.
- Write a separate Main class (with main() outside LibraryBook) to create two LibraryBook objects and display their information.

Algorithm:

Step-1: Define LibraryBook Class

- Declare class: class LibraryBook
- Declare instance variables: String title; String author; int bookID;

2.Define constructor: public LibraryBook(String title, String author, int bookID)

- Assign using this: this.title = title; this.author = author; this.bookID = bookID;

3.Define method: public void displayBookInfo()

Print:

- "Title: " + title
- "Author: " + author
- "Book ID: " + bookID

Step 2: Define Main Class

1. Declare public class: public class Main

2. Inside main() method:

- Create first book:
- LibraryBook book1 = new LibraryBook("Java Programming", "Herbert Schildt", 101);
- Create second book:
  - LibraryBook book2 = new LibraryBook("Python for Beginners", "Eric Matthes", 102);
- Call book1.displayBookInfo();
- Call book2.displayBookInfo();

Source code:

```
class LibraryBook {
    String title;
    String author;
    int bookID;
    public LibraryBook(String title, String author, int bookID) {
        this.title = title;
        this.author = author;
        this.bookID = bookID;
    }
    public void displayBookInfo() {
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Book ID: " + bookID);
    }
}
public class Main {
    public static void main(String[] args) {
        System.out.println("=== Library Book Information ===\n");
        LibraryBook book1 = new LibraryBook("Java Programming", "Herbert Schildt", 101);
        LibraryBook book2 = new LibraryBook("Python for Beginners", "Eric Matthes", 102);
        book1.displayBookInfo();
        book2.displayBookInfo();
    }
}
```

Output:

```
=== Library Book Information ===

Title: Java Programming
Author: Herbert Schildt
Book ID: 101
Title: Python for Beginners
Author: Eric Matthes
Book ID: 102

D:\java lab report VS code>
```

Discussion:

In this experiment, the LibraryBook class was created with attributes title, author, and bookID. The constructor uses the this keyword to distinguish between instance variables and parameters with the same name. The method displayBookInfo() prints details of each book. By creating multiple book objects in the Main class, the experiment demonstrates how constructors and **this** help manage object data efficiently.