Day_3_Assignment

Design a modular banking system that supports: • Customer management • Account operations (deposit, withdraw, transfer) • Transaction history • Branch-level customer segregation

//interface - BankOperations

```
public interface BankOperations {
    void deposit(double amount);
    void withdraw(double amount);
    void transfer(Account target, double amount);
    double checkBalance();
    void showTransactionHistory();
}
```

//Abstract Class- Account

```
import java.util.*;
public abstract class Account implements BankOperations {
    protected String accountNumber;
    protected double balance;
    protected List<String> transactionHistory = new ArrayList<>();
    public Account(String accountNumber, double initialBalance) {
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
        addTransaction("Account created with balance: ₹" + initialBalance);
    }
    public abstract void deposit(double amount);
    public abstract void withdraw(double amount);
    public void transfer(Account target, double amount) {
        this.withdraw(amount);
        target.deposit(amount);
        addTransaction("Transferred to Account " + target.accountNumber + ": ₹" + amount);
        target.addTransaction("Received from Account " + this.accountNumber + ": ₹" + amount);
    }
    public double checkBalance() {
        return balance;
    }
    protected void addTransaction(String info) {
        transactionHistory.add(info);
    }
    public void showTransactionHistory() {
        System.out.println("Account: " + accountNumber);
        for (String tx : transactionHistory) {
            System.out.println(" - " + tx);
        }
    }
    public String getAccountNumber() {
        return accountNumber;
    }
}
```

//SavingsAccount Class

```
public class SavingsAccount extends Account {
    private final double MIN_BALANCE = 1000.0;
    public SavingsAccount(String accountNumber, double initialBalance) {
        super(accountNumber, initialBalance);
    }
    @Override
    public void deposit(double amount) {
        balance += amount;
        addTransaction("Deposited: ₹" + amount);
    }
    @Override
    public void withdraw(double amount) {
        if (balance - amount >= MIN_BALANCE) {
            balance -= amount;
            addTransaction("Withdrawn: ₹" + amount);
        } else {
            addTransaction("Withdrawal Failed: Insufficient balance (min ₹1000 required)");
        }
    }
}
```

//CurrentAccount Class

```
public class CurrentAccount extends Account {
    private final double OVERDRAFT_LIMIT = 2000.0;
    public CurrentAccount(String accountNumber, double initialBalance) {
        super(accountNumber, initialBalance);
    }
    @Override
    public void deposit(double amount) {
        balance += amount;
        addTransaction("Deposited: ₹" + amount);
    }
    @Override
    public void withdraw(double amount) {
        if (balance - amount >= -OVERDRAFT_LIMIT) {
            balance -= amount;
            addTransaction("Withdrawn: ₹" + amount);
        } else {
            addTransaction("Withdrawal Failed: Overdraft limit exceeded");
        }
    }
}
```

//Customer Class

```
import java.util.*;

public class Customer {

    private String customerId;

    private String name;

    private List<Account> accounts = new ArrayList<>();

    public Customer(String customerId, String name) {

        this.customerId = customerId;

        this.name = name;

        System.out.println(" Customer Created: " + name + " [Customer ID: " + customerId + "]");

    }

    public void addAccount(Account acc) {

        accounts.add(acc);

    }

    public List<Account> getAccounts() {

        return accounts;

    }

    public String getCustomerId() {

        return customerId;

    }

    public String getName() {

        return name;

    }

}
```

//BankBranch Class

```
import java.util.*;

public class BankBranch {

    private String branchId;

    private String branchName;

    private List<Customer> customers = new ArrayList<>();

    public BankBranch(String branchId, String branchName) {
        this.branchId = branchId;
        this.branchName = branchName;
        System.out.println(" Branch Created: " + branchName + " [Branch ID: " + branchId + "]");
    }
    public void addCustomer(Customer c) {
        customers.add(c);
        System.out.println(" Customer added to branch.");
    }
    public Customer findCustomerById(String id) {
        for (Customer c : customers) {
            if (c.getCustomerId().equals(id)) return c;
        }
        return null;
    }
    public void listAllCustomers() {
        for (Customer c : customers) {
            System.out.println("Customer: " + c.getName() + " [ID: " + c.getCustomerId() + "]");
        }
    }
}
```

//Demo

```
public class BankDemo {
    public static void main(String[] args) {
        BankBranch branch = new BankBranch("B001", "Main Branch");
        Customer c1 = new Customer("C001", "Alice");
        branch.addCustomer(c1);
        SavingsAccount sa = new SavingsAccount("S001", 5000.0);
        CurrentAccount ca = new CurrentAccount("C001", 2000.0);
        c1.addAccount(sa);
        c1.addAccount(ca);
        System.out.println(" Savings Account [S001] opened with initial balance: ₹5000.0");

        System.out.println(" Current Account [C001] opened with initial balance: ₹2000.0 and overdraft
limit ₹2000.0");

        sa.deposit(2000.0);

        System.out.println(" Deposited ₹2000.0 to Savings Account [S001]");

        System.out.println(" Current Balance: ₹" + sa.checkBalance());

        ca.withdraw(2500.0);

        System.out.println(" Withdrawn ₹2500.0 from Current Account [C001]");

        System.out.println(" Current Balance: ₹" + ca.checkBalance());

        sa.transfer(ca, 1000.0);

        System.out.println(" Transferred ₹1000.0 from Savings Account [S001] to Current Account
[C001]");

        System.out.println(" Savings Balance: ₹" + sa.checkBalance());

        System.out.println(" Current Balance: ₹" + ca.checkBalance());

        System.out.println("\n Transaction History:");

        sa.showTransactionHistory();

        ca.showTransactionHistory();

    }

}
```

O/P:

```
Branch Created: Main Branch [Branch ID: B001]

 Customer Created: Alice [Customer ID: C001]

 Customer added to branch.

 Savings Account [S001] opened with initial balance: ₹5000.0

 Current Account [C001] opened with initial balance: ₹2000.0 and overdraft limit ₹2000.0

 Deposited ₹2000.0 to Savings Account [S001]

 Current Balance: ₹7000.0

 Withdrawn ₹2500.0 from Current Account [C001]

 Current Balance: -₹500.0

 Transferred ₹1000.0 from Savings Account [S001] to Current Account [C001]

 Savings Balance: ₹6000.0

 Current Balance: ₹500.0
 Transaction History:
Account: S001
 - Account created with balance: ₹5000.0
 - Deposited: ₹2000.0
 - Transferred to Account C001: ₹1000.0
Account: C001
 - Account created with balance: ₹2000.0
 - Withdrawn: ₹2500.0
 - Received from Account S001: ₹1000.0
```