

Banking System App

//Banking – Interface

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

// Abstract class – Account

```
package bankopp;  
  
public abstract class Account implements BankOperations {  
    protected String accountNumber;  
    protected double balance;  
    protected String[] transactionHistory = new String[100];  
    protected int transactionCount = 0;  
  
    public Account(String accountNumber, double initialBalance) {  
        this.accountNumber = accountNumber;  
        this.balance = initialBalance;  
    }  
  
    public void addTransaction(String info) {  
        if (transactionCount < transactionHistory.length) {  
            transactionHistory[transactionCount++] = info;  
        }  
    }  
  
    public void transfer(Account target, double amount) {
```

```
        if (this.balance >= amount) {
            this.withdraw(amount);
            target.deposit(amount);
            addTransaction("Transferred Account " + target.accountNumber + ": Rs" + amount);
        } else {
            System.out.println(" Insufficient balance to transfer.");
        }
    }

    public double checkBalance() {
        return balance;
    }

    public void showTransactionHistory() {
        System.out.println("Transaction History for Account: " + accountNumber);
        for (int i = 0; i < transactionCount; i++) {
            System.out.println("- " + transactionHistory[i]);
        }
    }
}

//class savings account
package bankopp;

public class SavingsAccount extends Account {
    private final double MIN_BALANCE = 1000.0;
    private double initialBalance;

    public SavingsAccount(String accountNumber, double initialBalance) {
        super(accountNumber, initialBalance);
        this.accountNumber = accountNumber;
        this.initialBalance = initialBalance;
    }
}
```

```
public void deposit(double amount) {  
    balance += amount;  
    addTransaction("Deposited: Rs" + amount);  
}  
  
public void withdraw(double amount) {  
    if (balance - amount >= MIN_BALANCE) {  
        balance -= amount;  
        addTransaction("Withdrawn: Rs" + amount);  
    } else {  
        System.out.println("Cannot withdraw below minimum balance.");  
    }  
}  
  
//Current Account  
package bankopp;  
  
public class CurrentAccount extends Account {  
    private final double OVERDRAFT_LIMIT = 2000.0;  
  
    public CurrentAccount(String accountNumber, double initialBalance) {  
        super(accountNumber, initialBalance);  
    }  
  
    public void deposit(double amount) {  
        balance += amount;  
        addTransaction("Deposited: Rs" + amount);  
    }  
  
    public void withdraw(double amount) {
```

```
        if (balance - amount >= -OVERDRAFT_LIMIT) {
            balance -= amount;
            addTransaction("Withdrawn: Rs" + amount);
        } else {
            System.out.println(" Overdraft limit exceeded.");
        }
    }
}

// Customer Class
package bankopp;

public class Customer {
    private String customerId;
    private String name;
    private Account[] accounts = new Account[5];
    private int accountCount = 0;

    public Customer(String customerId, String name) {
        this.customerId = customerId;
        this.name = name;
    }

    public void addAccount(Account acc) {
        if (accountCount < accounts.length) {
            accounts[accountCount++] = acc;
        }
    }

    public Account[] getAccounts() {
        return accounts;
    }
}
```

```
    public String getCustomerId() {  
        return customerId;  
    }  
  
    public String getName() {  
        return name;  
    }  
}  
  
// Class Bankbranch  
package bankopp;  
  
public class BankBranch {  
    private String branchId;  
    private String branchName;  
    private Customer[] customers = new Customer[50];  
    private int customerCount = 0;  
  
    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) {  
        if (customerCount < customers.length) {  
            customers[customerCount++] = c;  
            System.out.println("Customer Created: " + c.getName() + " [Customer ID: " +  
c.getCustomerId() + "]");  
            System.out.println("Customer added to branch.");  
        }  
    }  
}
```

```
public Customer findCustomerById(String id) {  
    for (int i = 0; i < customerCount; i++) {  
        if (customers[i].getCustomerId().equals(id)) {  
            return customers[i];  
        }  
    }  
    return null;  
}  
  
public void listAllCustomers() {  
    System.out.println(" All Customers in Branch " + branchName + ":");  
    for (int i = 0; i < customerCount; i++) {  
        System.out.println("- " + customers[i].getName() + " [ID: " +  
customers[i].getCustomerId() + "]");  
    }  
}  
}  
// Main Class  
package bankopp;  
  
public class Main {  
  
    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: Rs.5000.0");
```

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

```
sa.deposit(2000.0);
```

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

```
ca.withdraw(2500.0);
```

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

```
sa.transfer(ca, 1000.0);
```

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

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

```
System.out.println();
```

```
sa.showTransactionHistory();
```

```
ca.showTransactionHistory();
```

```
}
```

```
}
```