Advancing Knowledge, Enriching Lives

EAST DELTA UNIVERSITY

School of Science Engineering & Technology

Computer Science & Engineering (B.Sc. in CSE)

ASSIGNMENT

Topic:	Abstraction and Exception Handling

Session:	Summer 2025
Program:	B.Sc. in CSE
Course Code:	CSE 124
Course Title:	Object Oriented Programming Language Laboratory
Date of Submission:	20-08-2025

Submitted By:	Mohammad Fahim	Submitted To:
Student ID:	242002112	Sourav Adhikary
Student Name:	MOHAMMAD FAHIM	
Semester:	4th	
Section:	07	

Introduction:

The problem is to design a simple Bank System in Java that can handle customer account details and loan eligibility. The system must allow valid deposits, display updated balances, and determine loan eligibility while preventing invalid operations using exception handling.

Problem Statement:

We are required to design a Bank System in Java with the following specifications:
Create an abstract class BankAccount with account holder name, balance, an abstract deposit() method that throws IllegalArgumentException, and a concrete displayBalance() method.
Create an interface LoanEligibility with a method checkEligibility() that throws IllegalStateException if balance is negative.
Create a class CustomerAccount that extends BankAccount and implements LoanEligibility, providing implementations for deposit() and checkEligibility().
In the main method, take user input for account holder name, initial balance, and deposit amount. Use try-catch blocks to handle invalid inputs and exceptions. Display updated balance and loan eligibility.

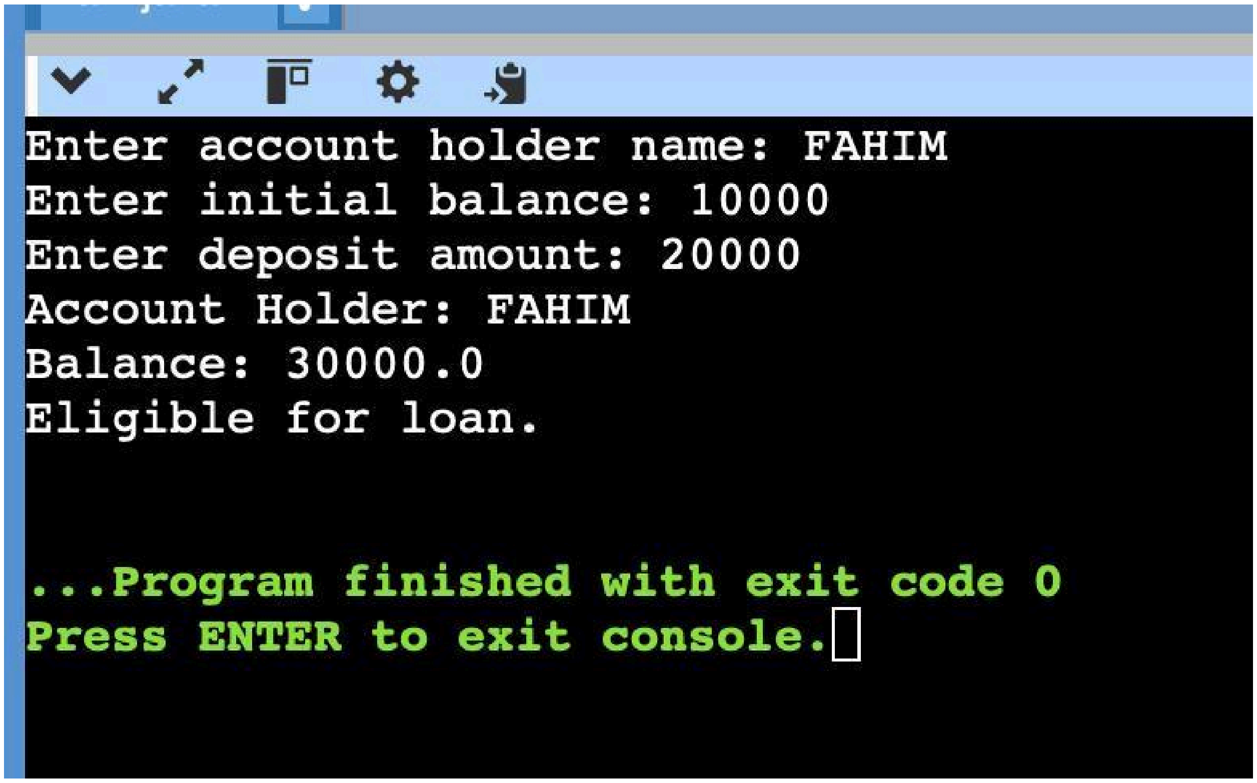
Code:

```
import java.util.Scanner;

abstract class BankAccount
{
	String accountHolder;
	double balance;

	BankAccount(String accountHolder, double balance)
	{
		this.accountHolder = accountHolder;
		this.balance = balance;
	}

	abstract void deposit(double amount) throws IllegalArgumentException;

	void displayBalance()
	{
		System.out.println("Account Holder: "+accountHolder);
		System.out.println("Balance: "+balance);
	}

}

interface LoanEligibility
{
	void checkEligibility();
}

class CustomerAccount extends BankAccount implements LoanEligibility
{

	CustomerAccount(String accountHolder, double balance)
	{
		super(accountHolder, balance);
	}

	public void deposit(double amount)
```

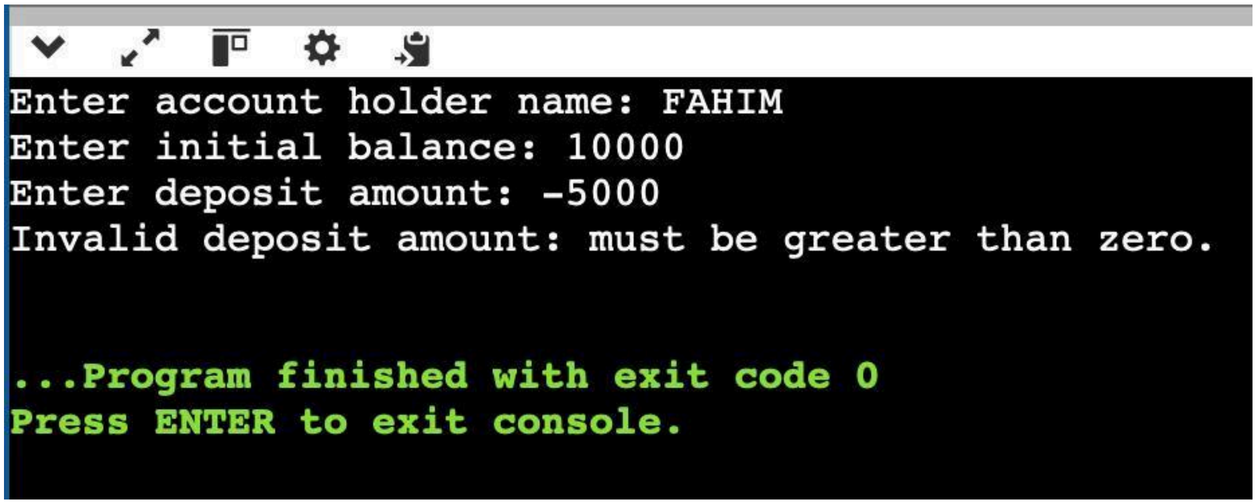
```
        {
                if(amount <= 0)
                {
                        throw new IllegalArgumentException("Invalid deposit amount: must
be greater than zero.");
                }
                balance += amount;
        }

        public void checkEligibility()
        {
                if (balance < 0)
                {
                        throw new IllegalStateException("Balance cannot be negative for
                loan eligibility check.");
                }
                if (balance >= 5000)
                {
                System.out.println("Eligible for loan.");
                }
                else
                {
                System.out.println("Not eligible for loan.");
                }
        }
}


public class Main
{
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);

                try
                {
                        System.out.print("Enter account holder name: ");
                        String name = sc.nextLine();

                        System.out.print("Enter initial balance: ");
```

```
				double initialBalance = sc.nextDouble();

				CustomerAccount customer = new CustomerAccount(name,
initialBalance);

				System.out.print("Enter deposit amount: ");
				double depositAmount = sc.nextDouble();

				try
				{
					customer.deposit(depositAmount);
				}
				catch (IllegalArgumentException e)
				{
					System.out.println(e.getMessage());
					return;
				}

				customer.displayBalance();

				try
				{
					customer.checkEligibility();
				}
				catch (IllegalStateException e)
				{
					System.out.println(e.getMessage());
				}

			}
			catch (Exception e)
			{
				System.out.println("Invalid input. Please enter valid data.");
			}
			finally
			{
				sc.close();
			}
		}
}
```

Output:



```
Enter account holder name: FAHIM
Enter initial balance: 10000
Enter deposit amount: 20000
Account Holder: FAHIM
Balance: 30000.0
Eligible for loan.


...Program finished with exit code 0
Press ENTER to exit console.
```



```
Enter account holder name: FAHIM
Enter initial balance: 10000
Enter deposit amount: -5000
Invalid deposit amount: must be greater than zero.


...Program finished with exit code 0
Press ENTER to exit console.
```

Discussion:

This project successfully demonstrates the application of Abstraction, Interfaces, and Exception Handling in Java. The abstract class encapsulates common account properties, while the interface enforces loan eligibility checking. The implementation ensures that deposits must be valid and balances cannot be negative when checking loan eligibility. Exception handling makes the program robust against invalid user inputs, ensuring reliable execution and clear error messages. This approach models real-world banking operations where validation and error handling are crucial.