Assignment-8

Title:

Create a program to simulate a banking system with exception handling for invalid transactions.

Objective:

To develop a Java program that simulates a banking system with exception handling to prevent invalid transactions such as negative balance initialization, negative deposit amounts, and insufficient funds during withdrawals.

Theory:

A banking system involves basic operations such as deposit, withdrawal, and balance inquiry. Exception handling in Java helps manage invalid operations by preventing erroneous transactions from affecting the system's integrity. Java provides built-in exceptions like `IllegalArgumentException` and `RuntimeException` to handle such cases effectively.

Algorithm:

1. Create a `BankAccount` class with a private balance variable.

2. Define a constructor that initializes the balance and checks for negative values.

3. Implement a `deposit` method that validates the deposit amount.

4. Implement a `withdraw` method that checks for sufficient funds before allowing withdrawal.

5. Use `try-catch` blocks to handle invalid transactions.

6. Create a `main` method to demonstrate the program.

Program Code:

```
import java.util.*;
```

```
}

class BankAccount {

    private double balance;



    public BankAccount(double initialBalance) {

        if (initialBalance < 0)

            throw new IllegalArgumentException("Initial balance cannot be negative.");

        this.balance = initialBalance;

    }



    public void deposit(double amount) {

        if (amount <= 0)

            throw new IllegalArgumentException("Deposit amount must be positive.");

        balance += amount;

        System.out.println("Deposited: " + amount + " | New Balance: " + balance);

    }



    public void withdraw(double amount) {

        if (amount <= 0)

            throw new IllegalArgumentException("Withdrawal amount must be
positive.");

        if (amount > balance)

            throw new RuntimeException("Insufficient funds.");

        balance -= amount;

        System.out.println("Withdrawn: " + amount + " | New Balance: " + balance);

    }
    public double getBalance() {
        return balance;
    }
}

public class BankingSystem {
    public static void main(String[] args) {
        try {
            BankAccount account = new BankAccount(500);
            account.deposit(200);
            account.withdraw(100);
            account.withdraw(700); // This should trigger an exception
        } catch (Exception e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

Output:

```
Deposited: 200.0 | New Balance: 700.0
Withdrawn: 100.0 | New Balance: 600.0
Exception: Insufficient funds.
```

Explanation:

1. A `BankAccount` object is created with an initial balance of 500.

2. A deposit of 200 is made, increasing the balance to 700.

3. A withdrawal of 100 is processed successfully, reducing the balance to 600.

4. A withdrawal of 700 is attempted, which exceeds the balance, triggering an exception.

Conclusion:

The program successfully simulates a banking system with exception handling for invalid transactions. It prevents negative deposits, withdrawals exceeding the balance, and improper balance initialization, ensuring system stability and reliability.