1. Problem Statement: Develop a BankAccount class that implements core banking operations: ○ balanceEnquiry(): Displays the current account balance. ○ withdraw(): Deducts the specified amount from the account balance. ○ deposit(): Adds the specified amount to the account balance. Implement user-defined exceptions: a. LowBalanceException: Thrown when a withdrawal amount exceeds the available balance. b. NegativeNumberException: Thrown when attempting to deposit or withdraw a negative amount. Develop a Java application program that demonstrates these functionalities and properly handles these exceptions using try-catch blocks.

```
import java.util.Scanner;

public class BankAccountApp {

    static class LowBalanceException extends Exception{
        public LowBalanceException(String message){
            super(message);
        }
    }

    static class NegativeNumberException extends Exception{
        public NegativeNumberException(String message){
            super(message);
        }
    }

    static class BankAccount{
        private String accHolder;
        private double balance;

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

        public void balanceEnquiry(){
            System.out.println("Account holder : "+accHolder);
            System.out.println("Current Balance : "+balance);
        }

        public void deposite(double amount) throws NegativeNumberException{
            if(amount < 0){
                throw new NegativeNumberException("Cannot add negative amount");
            }

            balance = balance+amount;
            System.out.println(amount+" Amount added !");
            System.out.println("Current balance : "+balance);
        }

        public void withdraw(double amount) throws NegativeNumberException, LowBalanceException{
            if(balance < 0){
                throw new LowBalanceException("Balance not sufficient");
            }
            if(amount < 0){
                throw new NegativeNumberException("Cannot add negative amount");
            }

            balance = balance - amount;
            System.out.println(amount + " is withdrawn");
            System.out.println("Current balance : "+balance);
        }

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

            System.out.println("Enter account holder name : ");
            String accHolder = scan.nextLine();

            System.out.println("Enter Initial balance : ");
            double balance = scan.nextDouble();

            BankAccount bank = new BankAccount(accHolder, balance);

            while (true){
                System.out.println("1. Balance Enquiry");
                System.out.println("2. Deposit");
                System.out.println("3. Withdraw");
                System.out.println("4. Exit");

                System.out.print("Enter your choice : ");
                int choice = scan.nextInt();
            try {
                switch (choice) {

                    case 1:
                        bank.balanceEnquiry();
                        break;

                    case 2:
                        System.out.print("Enter amount to deposite : ");
                        Double amount = scan.nextDouble();
                        bank.deposite(amount);
                        break;

                    case 3:
                        System.out.println("Enter amount to withdraw : ");
                        Double amount2 = scan.nextDouble();
                        bank.withdraw(amount2);
                        break;

                    case 4:
                        System.out.println("Exiting....");
                        System.exit(0);
                        break;

                    default:
                        System.out.println("Invalid Input");
                        break;
                }
            }
            catch (LowBalanceException | NegativeNumberException e){
                System.out.println("Exception : " + e.getMessage());
            }
            }
        }
    }
}
```

2. Write a Java program with a method that takes an integer as input. If the number is odd, the method should throw a custom exception (OddNumberException). Handle this exception in the main program.

```
import java.util.Scanner;

public class OddNumberCheck {

    // Custom Exception Class
    static class OddNumberException extends Exception {
        public OddNumberException(String message) {
            super(message);
        }
    }

    // Method that throws exception if number is odd
    public static void checkEven(int number) throws OddNumberException {
        if (number % 2 != 0) {
            throw new OddNumberException("OddNumberException: The number " + number + " is odd.");
        } else {
            System.out.println("The number " + number + " is even.");
        }
    }

    // Main method
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int num = sc.nextInt();

        try {
            checkEven(num);
        } catch (OddNumberException e) {
            System.out.println("Caught Exception: " + e.getMessage());
        }

        sc.close();
    }
}
```

3. Create a package ExceptionHandlingDemo containing classes Calculator and DivisionException. ○ The Calculator class should have a method divide(int a, int b) that performs division. ○ If b is zero, throw a custom exception DivisionException with an appropriate error message. ○ Handle the exception in the main program and display an error message instead of crashing.

```
package ExceptionHandlingDemo;

public class DivisionException extends Exception {

    public DivisionException(String message) {
        super(message);
    }
}
```

```
package ExceptionHandlingDemo;

public class Calculator {

    public int divide(int a, int b) throws DivisionException {
        if (b == 0) {
            throw new DivisionException("Division by zero is not allowed.");
        }
        return a / b;
    }

    public int add(int a, int b){
        return a + b;
    }

    public int sub(int a, int b){
        return a - b;
    }

    public int multi(int a, int b){
        return a*b;
    }


}
```

```
import ExceptionHandlingDemo.Calculator;
import ExceptionHandlingDemo.DivisionException;
import java.util.Scanner;

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

        System.out.print("Enter numerator: ");
        int a = sc.nextInt();
        System.out.print("Enter denominator: ");
        int b = sc.nextInt();

        try {
            int result = calc.divide(a, b);
            System.out.println("Result: " + result);
        } catch (DivisionException e) {
            System.out.println("Error: " + e.getMessage());
        }

        sc.close();
    }
}
```