Date: 18-11-2022

Roll No & Name : 21BCE231 Prajapati.Naitik.J

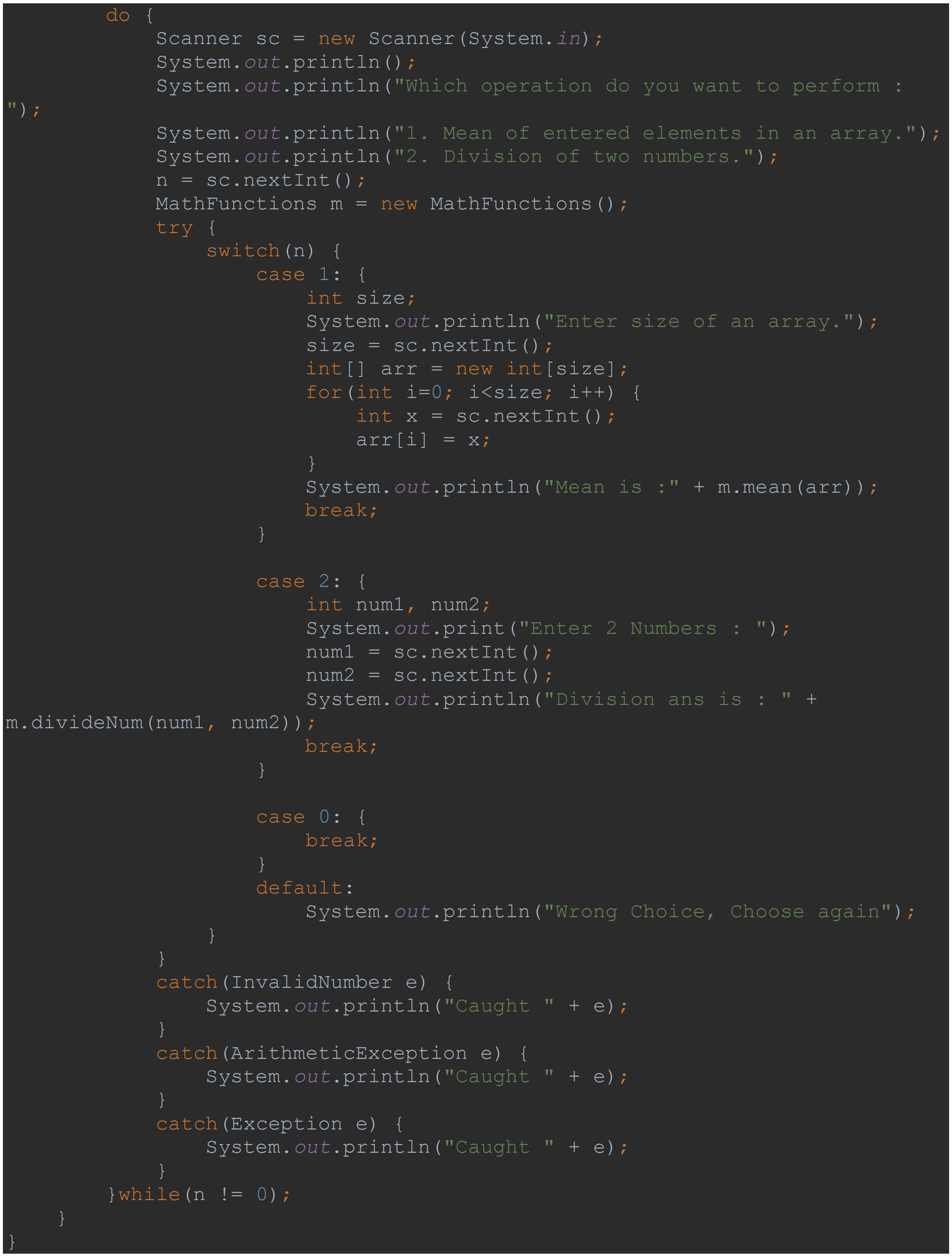
Course Code & Name : 2CS302 - Object Oriented Programming

Practical No : 9(A)

Aim : Try – Catch Implementation.

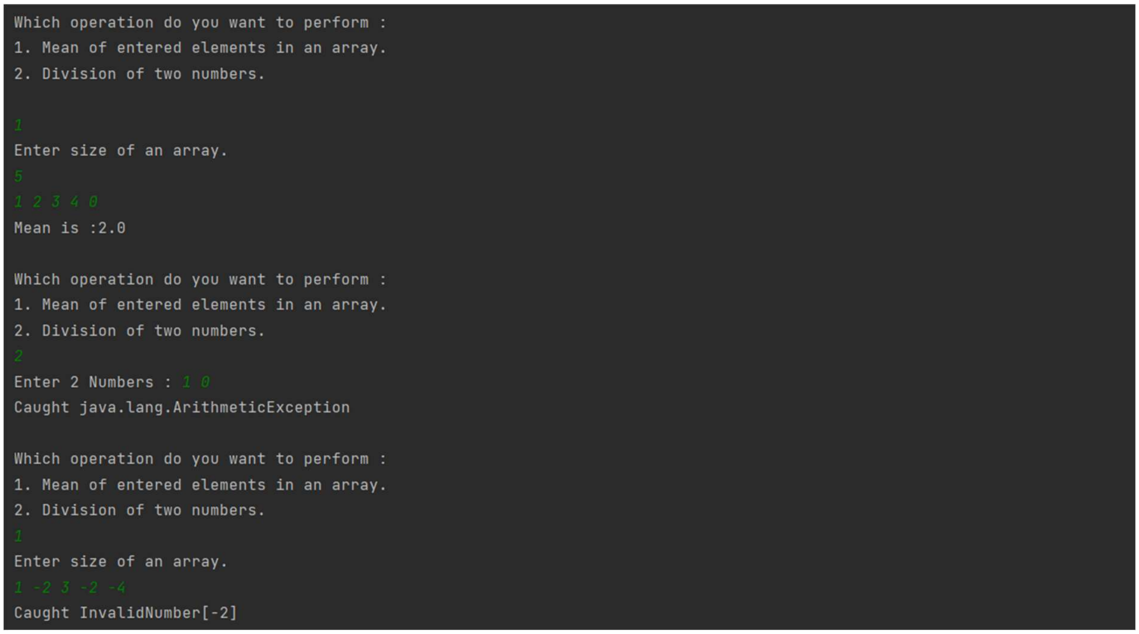
Methodology Followed :

```
import java.util.*;
class InvalidNumber extends Exception {
    private int num;

    InvalidNumber(int val) {
        num = val;
    }

    public String toString() {
        return "InvalidNumber[" + num + "]";
    }
}

class MathFunctions {
    float mean(int[] array) throws InvalidNumber {
        int sum = 0;
        int cnt = 0;
        for(int i=0; i<array.length; i++) {
            if(array[i] < 0 || array[i] > 100) {
                throw new InvalidNumber(array[i]);

            }
            else {
                sum += array[i];
                cnt++;
            }
        }
        float ans = sum / cnt;
        return ans;
    }

    float divideNum(int a, int b) throws ArithmeticException{
        if(b == 0) {
            throw new ArithmeticException();
        }
        else {
            return (a/b);
        }
    }
}

public class Main {
    public static void main(String[] args) {
        int n;
```



```
        do {
            Scanner sc = new Scanner(System.in);
            System.out.println();
            System.out.println("Which operation do you want to perform :
");
            System.out.println("1. Mean of entered elements in an array.");
            System.out.println("2. Division of two numbers.");
            n = sc.nextInt();
            MathFunctions m = new MathFunctions();
            try {
                switch(n) {
                    case 1: {
                        int size;
                        System.out.println("Enter size of an array.");
                        size = sc.nextInt();
                        int[] arr = new int[size];
                        for(int i=0; i<size; i++) {
                            int x = sc.nextInt();
                            arr[i] = x;
                        }
                        System.out.println("Mean is :" + m.mean(arr));
                        break;
                    }

                    case 2: {
                        int num1, num2;
                        System.out.print("Enter 2 Numbers : ");
                        num1 = sc.nextInt();
                        num2 = sc.nextInt();
                        System.out.println("Division ans is : " +
m.divideNum(num1, num2));
                        break;
                    }

                    case 0: {
                        break;
                    }
                    default:
                        System.out.println("Wrong Choice, Choose again");
                }
            }
            catch(InvalidNumber e) {
                System.out.println("Caught " + e);
            }
            catch(ArithmeticException e) {
                System.out.println("Caught " + e);
            }
            catch(Exception e) {
                System.out.println("Caught " + e);
            }
        }while(n != 0);
    }
}
```

Input/Output :

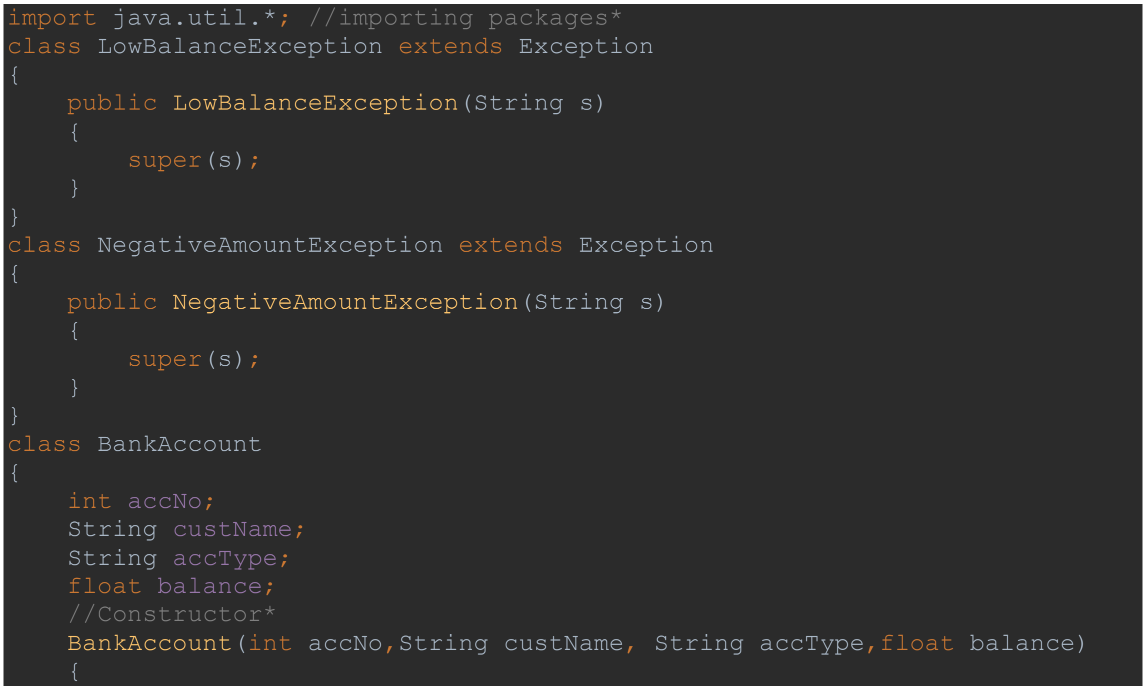


```
Which operation do you want to perform :
1. Mean of entered elements in an array.
2. Division of two numbers.

1
Enter size of an array.
5
1 2 3 4 0
Mean is :2.0

Which operation do you want to perform :
1. Mean of entered elements in an array.
2. Division of two numbers.
2
Enter 2 Numbers : 1 0
Caught java.lang.ArithmeticException

Which operation do you want to perform :
1. Mean of entered elements in an array.
2. Division of two numbers.
1
Enter size of an array.
1 -2 3 -2 -4
Caught InvalidNumber[-2]
```

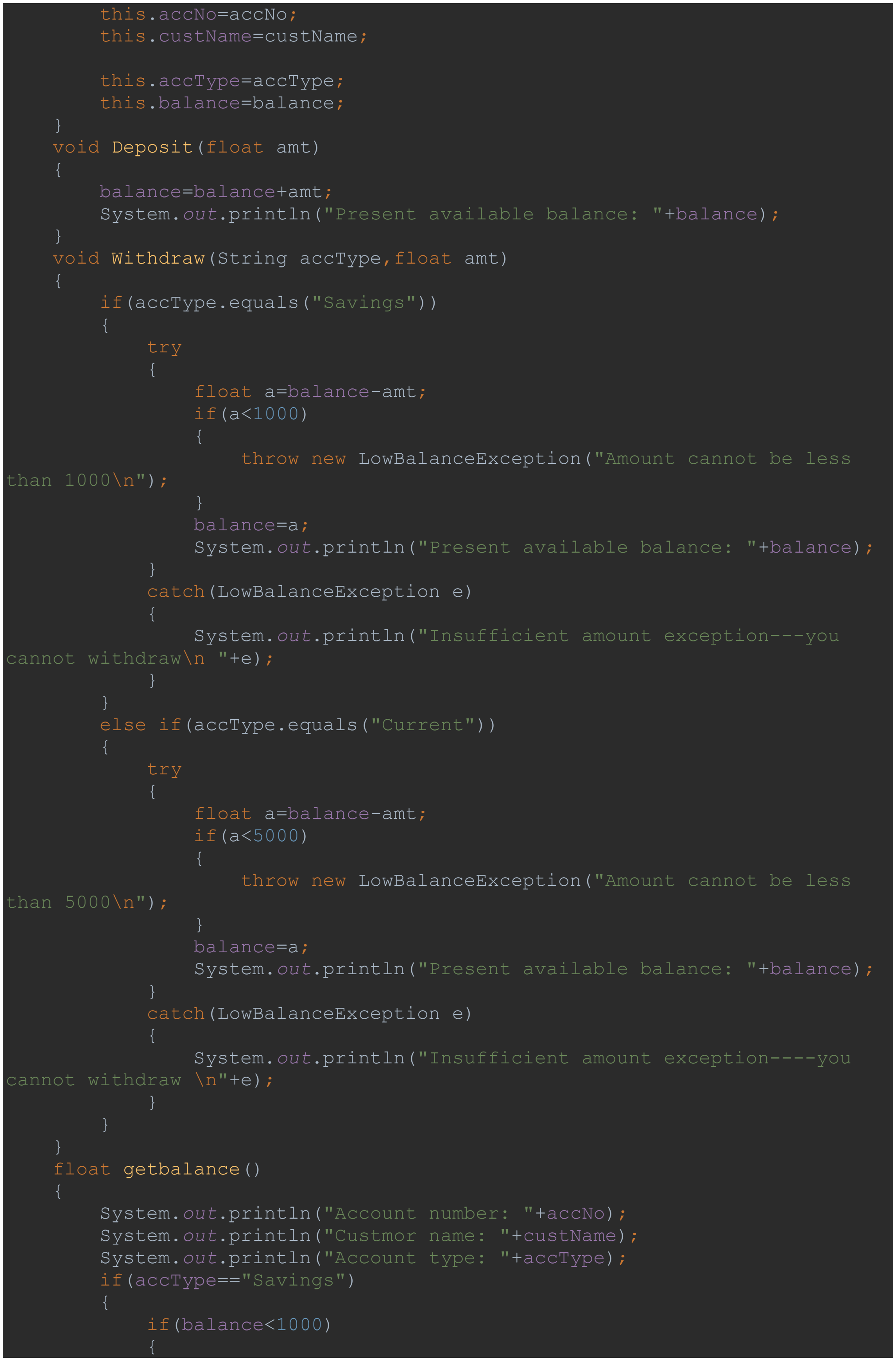
Practical No : 9(B)

Aim : Try – Catch Implementation.

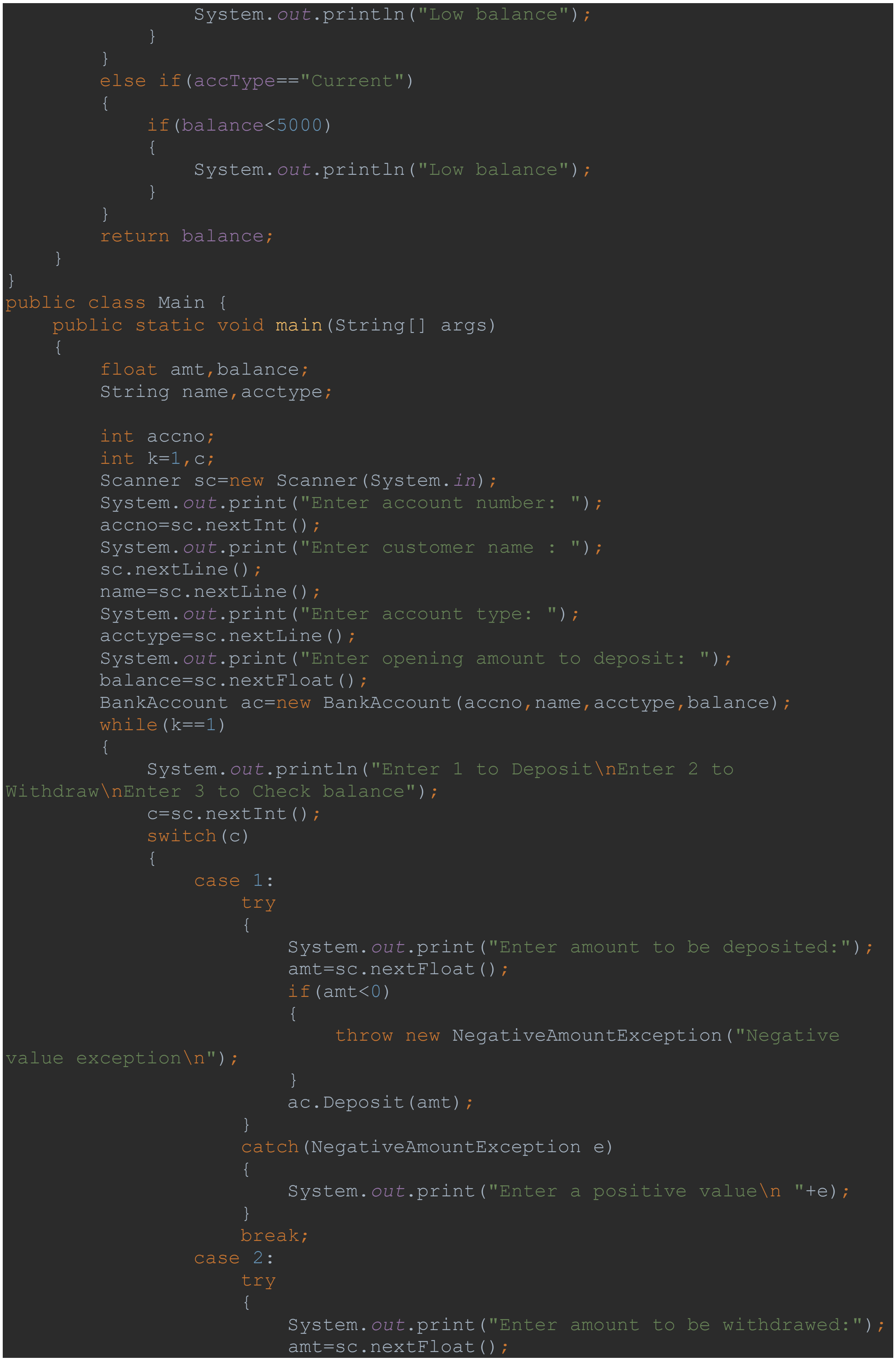
Methodology Followed :



```java
import java.util.*; //importing packages*
class LowBalanceException extends Exception
{
    public LowBalanceException(String s)
    {
        super(s);
    }
}
class NegativeAmountException extends Exception
{
    public NegativeAmountException(String s)
    {
        super(s);
    }
}
class BankAccount
{
    int accNo;
    String custName;
    String accType;
    float balance;
    //Constructor*
    BankAccount(int accNo,String custName, String accType,float balance)
    {
```

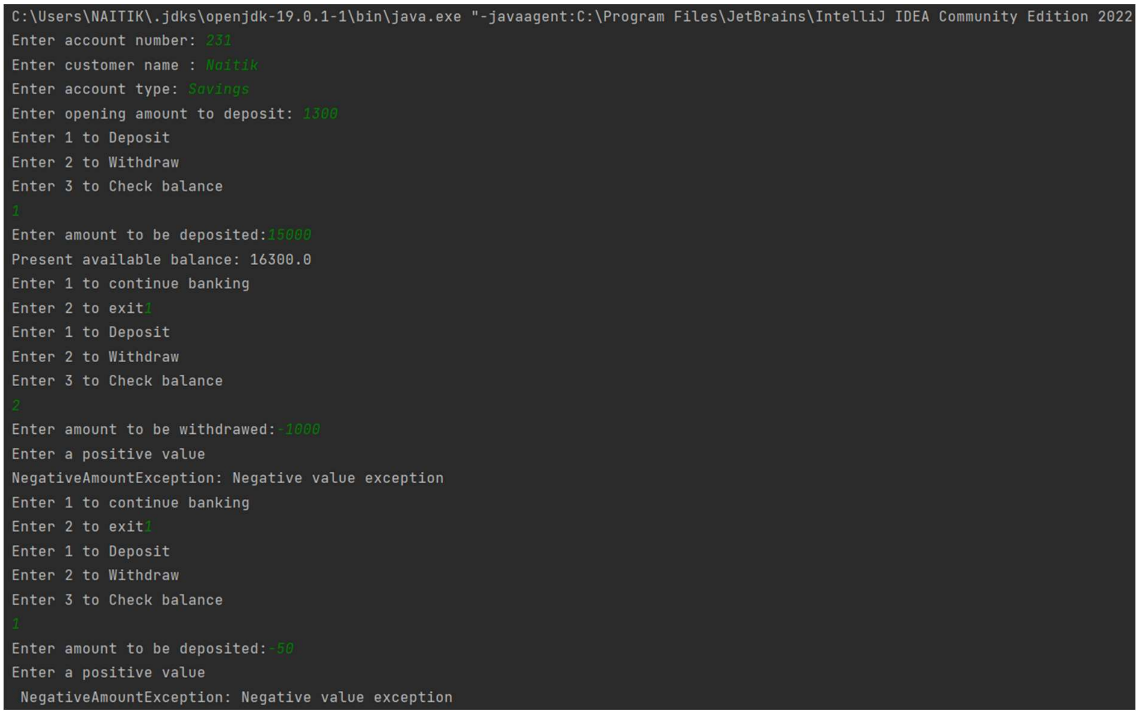


```
        this.accNo=accNo;
        this.custName=custName;

        this.accType=accType;
        this.balance=balance;
    }
    void Deposit(float amt)
    {
        balance=balance+amt;
        System.out.println("Present available balance: "+balance);
    }
    void Withdraw(String accType,float amt)
    {
        if(accType.equals("Savings"))
        {
            try
            {
                float a=balance-amt;
                if(a<1000)
                {
                    throw new LowBalanceException("Amount cannot be less
than 1000\n");
                }
                balance=a;
                System.out.println("Present available balance: "+balance);
            }
            catch(LowBalanceException e)
            {
                System.out.println("Insufficient amount exception---you
cannot withdraw\n "+e);
            }
        }
        else if(accType.equals("Current"))
        {
            try
            {
                float a=balance-amt;
                if(a<5000)
                {
                    throw new LowBalanceException("Amount cannot be less
than 5000\n");
                }
                balance=a;
                System.out.println("Present available balance: "+balance);
            }
            catch(LowBalanceException e)
            {
                System.out.println("Insufficient amount exception----you
cannot withdraw \n"+e);
            }
        }
    }
    float getbalance()
    {
        System.out.println("Account number: "+accNo);
        System.out.println("Custmor name: "+custName);
        System.out.println("Account type: "+accType);
        if(accType=="Savings")
        {
            if(balance<1000)
            {
```

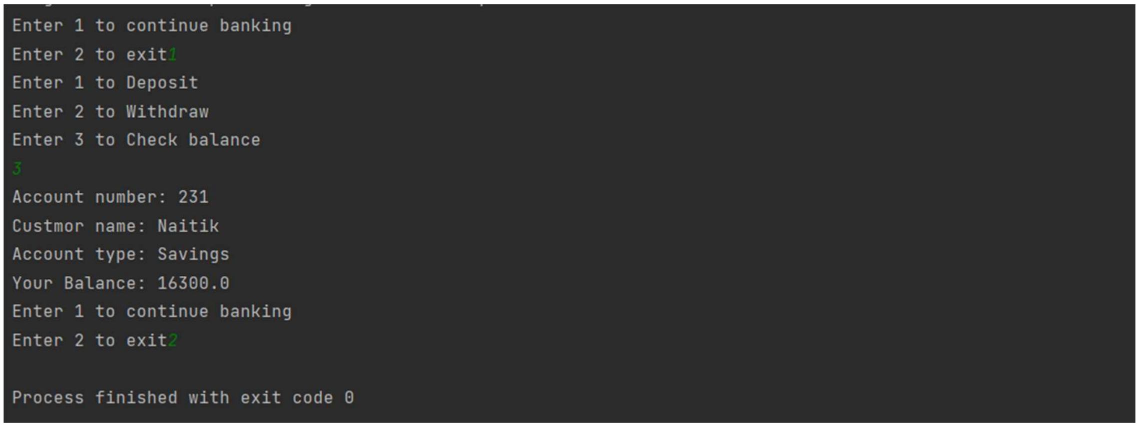


```
                System.out.println("Low balance");
            }
        }
        else if(accType=="Current")
        {
            if(balance<5000)
            {
                System.out.println("Low balance");
            }
        }
        return balance;
    }
}
public class Main {
    public static void main(String[] args)
    {
        float amt,balance;
        String name,acctype;

        int accno;
        int k=1,c;
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter account number: ");
        accno=sc.nextInt();
        System.out.print("Enter customer name : ");
        sc.nextLine();
        name=sc.nextLine();
        System.out.print("Enter account type: ");
        acctype=sc.nextLine();
        System.out.print("Enter opening amount to deposit: ");
        balance=sc.nextFloat();
        BankAccount ac=new BankAccount(accno,name,acctype,balance);
        while(k==1)
        {
            System.out.println("Enter 1 to Deposit\nEnter 2 to
Withdraw\nEnter 3 to Check balance");
            c=sc.nextInt();
            switch(c)
            {
                case 1:
                    try
                    {
                        System.out.print("Enter amount to be deposited:");
                        amt=sc.nextFloat();
                        if(amt<0)
                        {
                            throw new NegativeAmountException("Negative
value exception\n");
                        }
                        ac.Deposit(amt);
                    }
                    catch(NegativeAmountException e)
                    {
                        System.out.print("Enter a positive value\n "+e);
                    }
                    break;
                case 2:
                    try
                    {
                        System.out.print("Enter amount to be withdrawed:");
                        amt=sc.nextFloat();
                        if(amt<0)
                        {
                            throw new NegativeAmountException("Negative 
value exception\n");
                        }
                        ac.Withdraw(acctype,amt);
                    }
                    catch(NegativeAmountException e)
                    {
                        System.out.print("Enter a positive value \n"+e);
                    }
                    break;
                case 3:
                    float bal=ac.getbalance();
                    System.out.println("Your Balance:  "+bal);
                    break;
                default:
                    break;
            }
            System.out.print("Enter 1 to continue banking\nEnter 2 to 
exit");
            k=sc.nextInt();
        }
    }
}
```

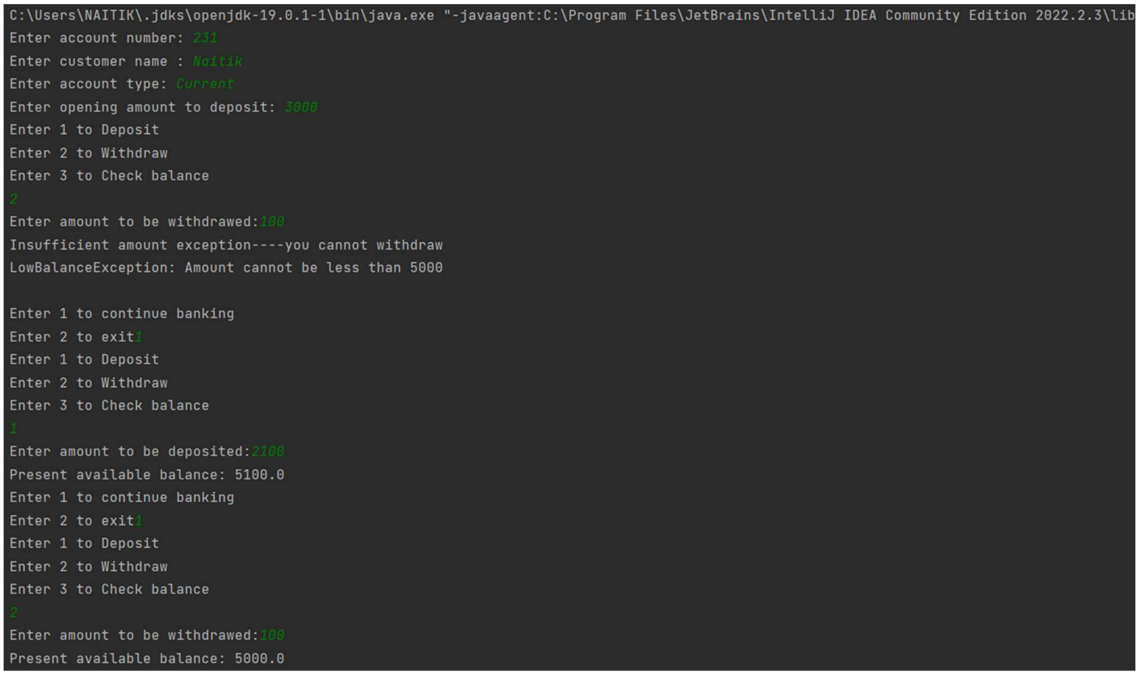
Input/Output :



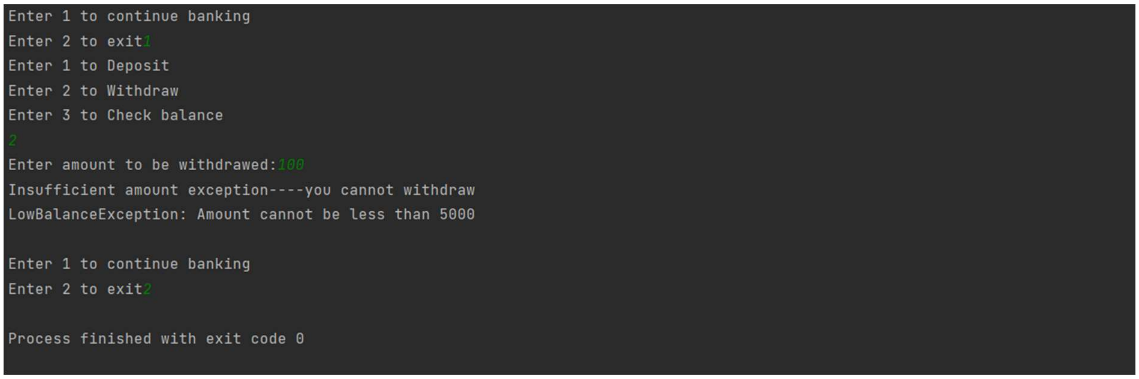
```
C:\Users\NAITIK\.jdks\openjdk-19.0.1-1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022
Enter account number: 231
Enter customer name : Naitik
Enter account type: Savings
Enter opening amount to deposit: 1300
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
1
Enter amount to be deposited:15000
Present available balance: 16300.0
Enter 1 to continue banking
Enter 2 to exit1
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
2
Enter amount to be withdrawed:-1000
Enter a positive value 
NegativeAmountException: Negative value exception
Enter 1 to continue banking
Enter 2 to exit1
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
1
Enter amount to be deposited:-50
Enter a positive value 
 NegativeAmountException: Negative value exception
```



```
Enter 1 to continue banking
Enter 2 to exit1
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
3
Account number: 231
Custmor name: Naitik
Account type: Savings
Your Balance: 16300.0
Enter 1 to continue banking
Enter 2 to exit2

Process finished with exit code 0
```



```
C:\Users\NAITIK\.jdks\openjdk-19.0.1-1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.2.3\lib
Enter account number: 231
Enter customer name : Naitik
Enter account type: Current
Enter opening amount to deposit: 3000
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
2
Enter amount to be withdrawed:100
Insufficient amount exception----you cannot withdraw
LowBalanceException: Amount cannot be less than 5000

Enter 1 to continue banking
Enter 2 to exit1
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
1
Enter amount to be deposited:2100
Present available balance: 5100.0
Enter 1 to continue banking
Enter 2 to exit1
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
2
Enter amount to be withdrawed:100
Present available balance: 5000.0
```



```
Enter 1 to continue banking
Enter 2 to exit1
Enter 1 to Deposit
Enter 2 to Withdraw
Enter 3 to Check balance
2
Enter amount to be withdrawed:100
Insufficient amount exception----you cannot withdraw
LowBalanceException: Amount cannot be less than 5000

Enter 1 to continue banking
Enter 2 to exit2

Process finished with exit code 0
```

CONCLUSION: - FROM THIS PRACTICAL WE HAVE LEARNT THE CONCEPT OF EXCEPTION HANDLING AND KEYWORDS LIKE TRY CATCH, FINALLY,CUSTOM EXCEPTIONS.