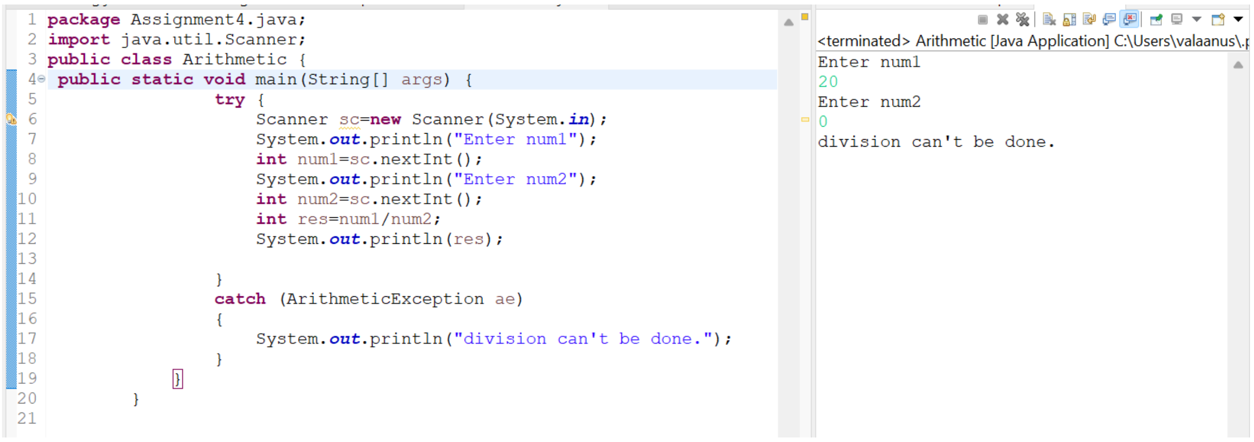
Assignments on Exception Handling

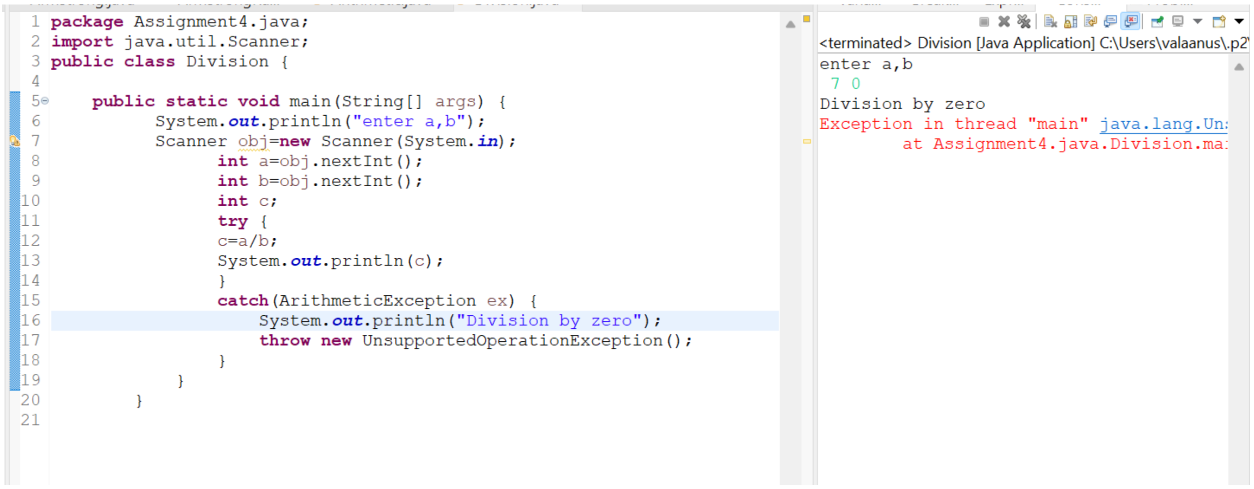
1) Write an application that accepts two numbers, divides the first number with the second number and display the result. Hint: You need to handle **ArithmeticException** which is thrown when there is an attempt to divide a number by zero.



```java
package Assignment4.java;
import java.util.Scanner;
public class Arithmetic {
 public static void main(String[] args) {
                try {
                        Scanner sc=new Scanner(System.in);
                        System.out.println("Enter num1");
                        int num1=sc.nextInt();
                        System.out.println("Enter num2");
                        int num2=sc.nextInt();
                        int res=num1/num2;
                        System.out.println(res);

                }
                catch (ArithmeticException ae)
                {
                        System.out.println("division can't be done.");
                }
        }
    }

```

```
<terminated> Arithmetic [Java Application] C:\Users\valaanus\.
Enter num1
20
Enter num2
0
division can't be done.
```

2) Carrying forward with the above problem, handle **ArithmeticException** by raising **UnsupportedOperationException** as a solution.



```java
package Assignment4.java;
import java.util.Scanner;
public class Division {

    public static void main(String[] args) {
          System.out.println("enter a,b");
          Scanner obj=new Scanner(System.in);
                int a=obj.nextInt();
                int b=obj.nextInt();
                int c;
                try {
                c=a/b;
                System.out.println(c);
                }
                catch(ArithmeticException ex) {
                    System.out.println("Division by zero");
                    throw new UnsupportedOperationException();
                }
        }
    }

```

```
<terminated> Division [Java Application] C:\Users\valaanus\.p2
enter a,b
 7 0
Division by zero
Exception in thread "main" java.lang.Un
        at Assignment4.java.Division.ma
```

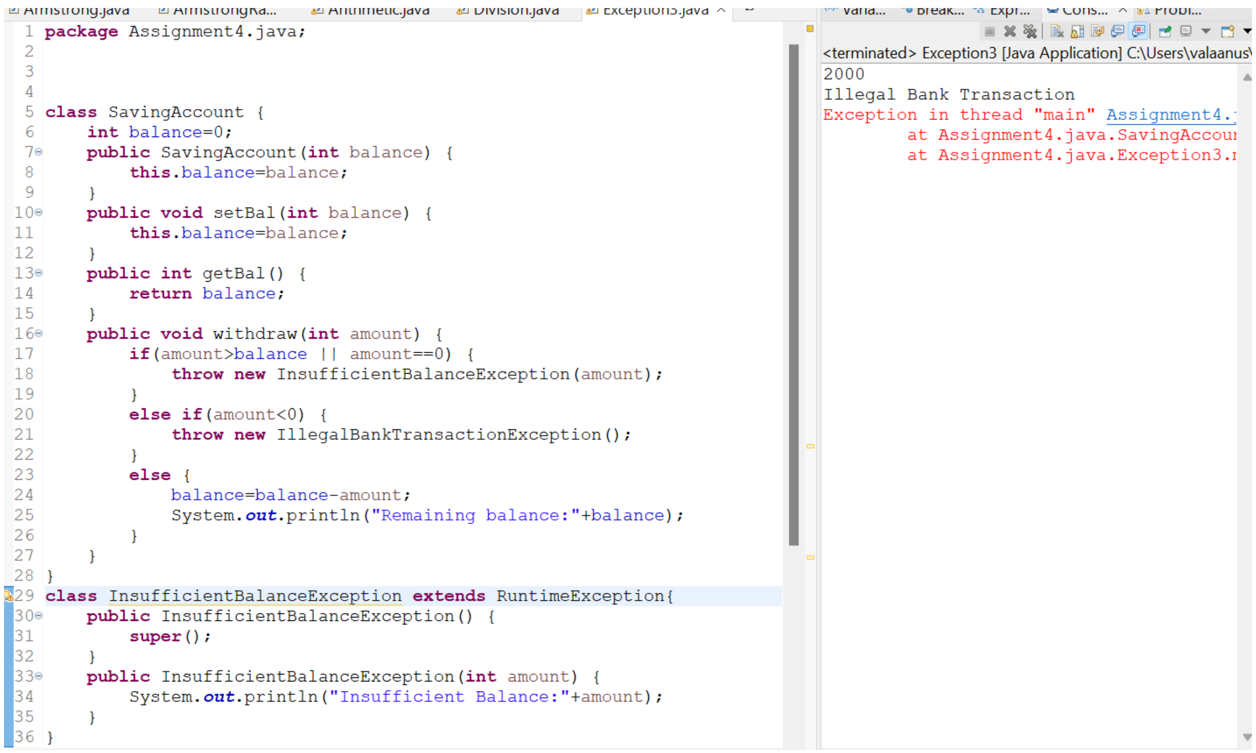
3) Write an application to perform withdraw functionality on a SavingAccount object. Point to note:

   a. Raise **InsufficientBalanceException** if you are trying to withdraw more than balance or when you balance is zero. E.g. if you balance is 2000 and if you are trying to withdraw 2100 or if you balance is 0 and you are trying to withdraw positive value.

   b. Raise **IllegalBankTransactionException** if you are trying to withdraw a negative value from your balance. E.g. if you try to withdraw a negative value savingAcc.withdraw(-1000);

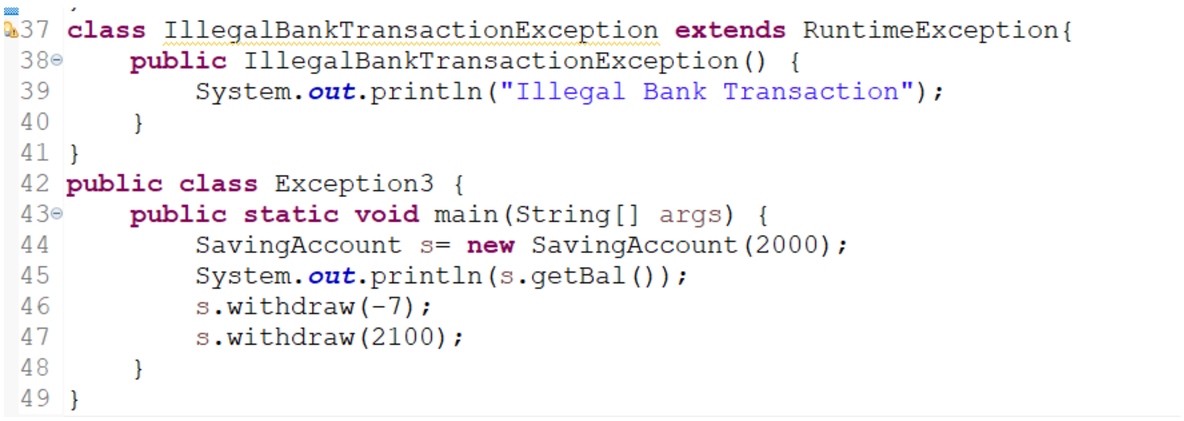
Note: SavingAccount

```
|-- long id
|-- double balance
|--double withdraw(double amount)
|--double deposit(double amount)
```



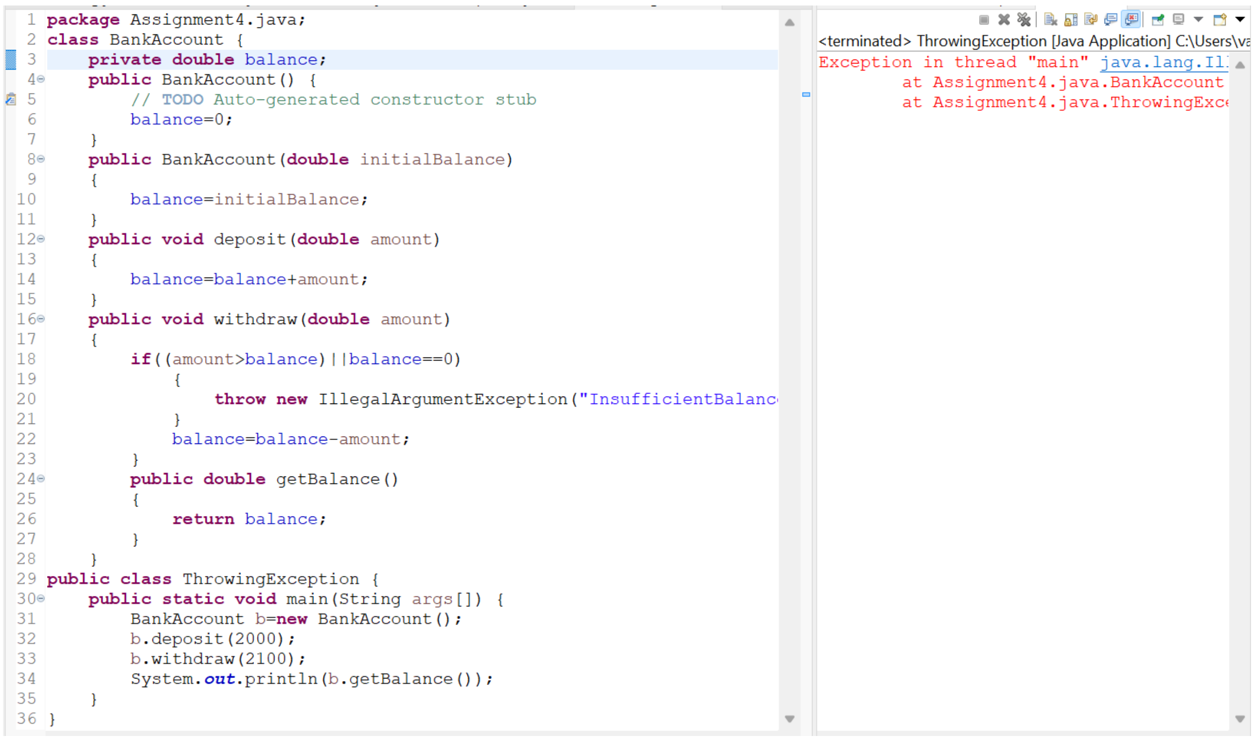
```java
package Assignment4.java;



class SavingAccount {
    int balance=0;
    public SavingAccount(int balance) {
        this.balance=balance;
    }
    public void setBal(int balance) {
        this.balance=balance;
    }
    public int getBal() {
        return balance;
    }
    public void withdraw(int amount) {
        if(amount>balance || amount==0) {
            throw new InsufficientBalanceException(amount);
        }
        else if(amount<0) {
            throw new IllegalBankTransactionException();
        }
        else {
            balance=balance-amount;
            System.out.println("Remaining balance:"+balance);
        }
    }
}
class InsufficientBalanceException extends RuntimeException{
    public InsufficientBalanceException() {
        super();
    }
    public InsufficientBalanceException(int amount) {
        System.out.println("Insufficient Balance:"+amount);
    }
}
```

```
<terminated> Exception3 [Java Application] C:\Users\valaanus\
2000
Illegal Bank Transaction
Exception in thread "main" Assignment4.
        at Assignment4.java.SavingAccoun
        at Assignment4.java.Exception3.m
```



```java
class IllegalBankTransactionException extends RuntimeException{
    public IllegalBankTransactionException() {
        System.out.println("Illegal Bank Transaction");
    }
}
public class Exception3 {
    public static void main(String[] args) {
        SavingAccount s= new SavingAccount(2000);
        System.out.println(s.getBal());
        s.withdraw(-7);
        s.withdraw(2100);
    }
}
```

3(b)



```java
package Assignment4.java;
class BankAccount {
    private double balance;
    public BankAccount() {
        // TODO Auto-generated constructor stub
        balance=0;
    }
    public BankAccount(double initialBalance)
    {
        balance=initialBalance;
    }
    public void deposit(double amount)
    {
        balance=balance+amount;
    }
    public void withdraw(double amount)
    {
        if((amount>balance)||balance==0)
            {
                throw new IllegalArgumentException("InsufficientBalanc
            }
            balance=balance-amount;
        }
        public double getBalance()
        {
            return balance;
        }
    }
public class ThrowingException {
    public static void main(String args[]) {
        BankAccount b=new BankAccount();
        b.deposit(2000);
        b.withdraw(2100);
        System.out.println(b.getBalance());
    }
}
```

```
<terminated> ThrowingException [Java Application] C:\Users\va
Exception in thread "main" java.lang.Il
        at Assignment4.java.BankAccount
        at Assignment4.java.ThrowingExce
```