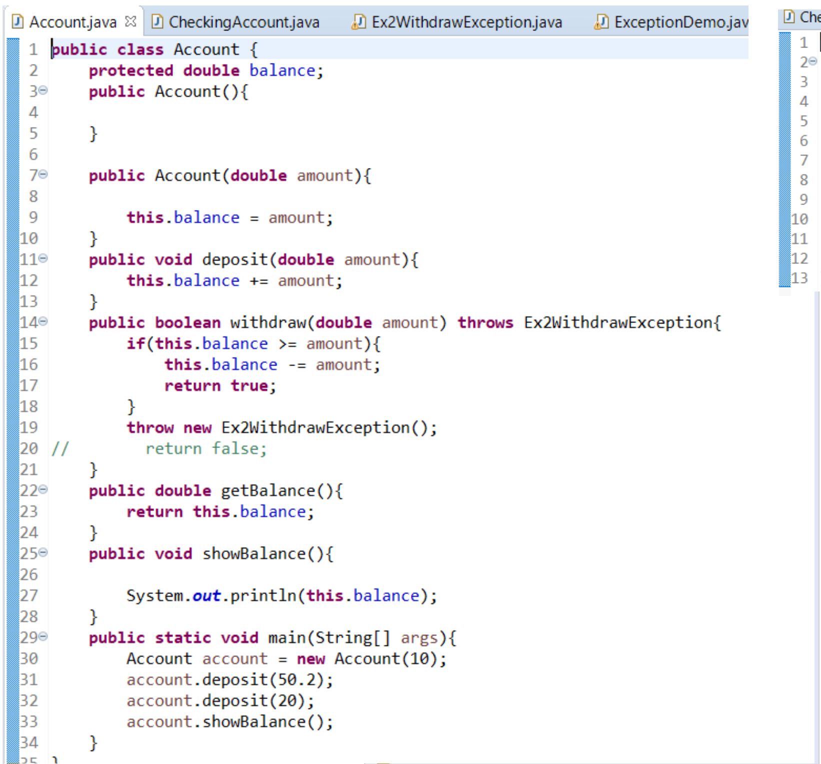
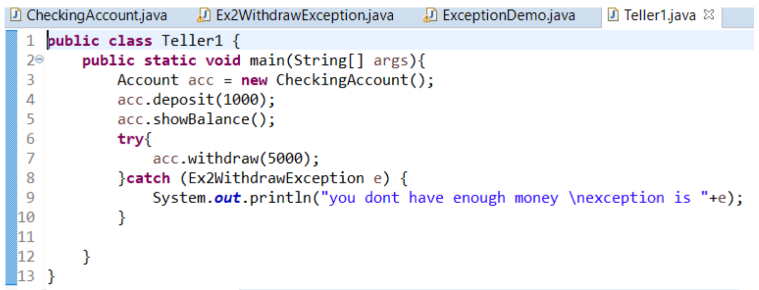
Lab 9 Exception การจัดการข้อผิดพลาด



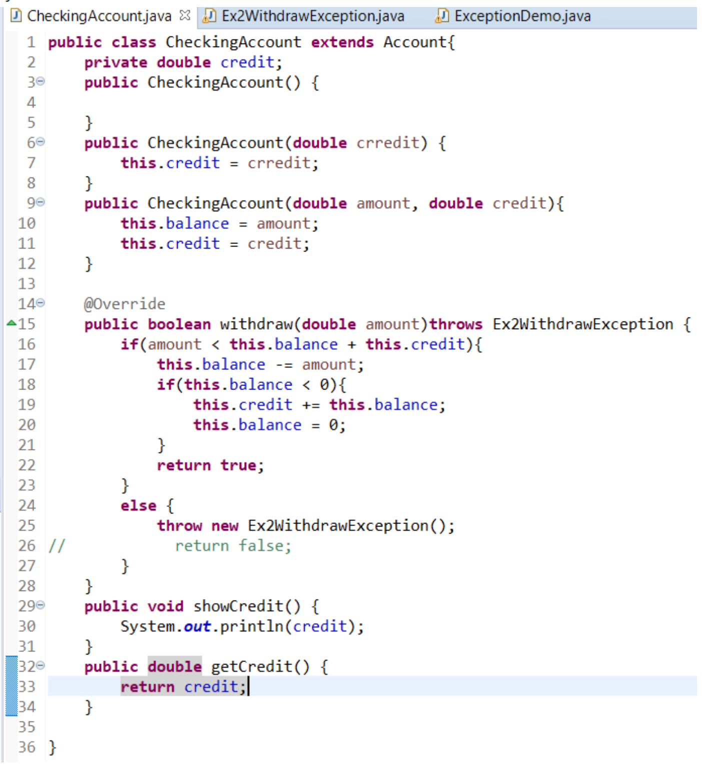
Account.java | CheckingAccount.java | Ex2WithdrawException.java | ExceptionDemo.jav

```java
public class Account {
    protected double balance;
    public Account(){

    }

    public Account(double amount){

        this.balance = amount;
    }
    public void deposit(double amount){
        this.balance += amount;
    }
    public boolean withdraw(double amount) throws Ex2WithdrawException{
        if(this.balance >= amount){
            this.balance -= amount;
            return true;
        }
        throw new Ex2WithdrawException();
//          return false;
    }
    public double getBalance(){
        return this.balance;
    }
    public void showBalance(){

        System.out.println(this.balance);
    }
    public static void main(String[] args){
        Account account = new Account(10);
        account.deposit(50.2);
        account.deposit(20);
        account.showBalance();
    }
```



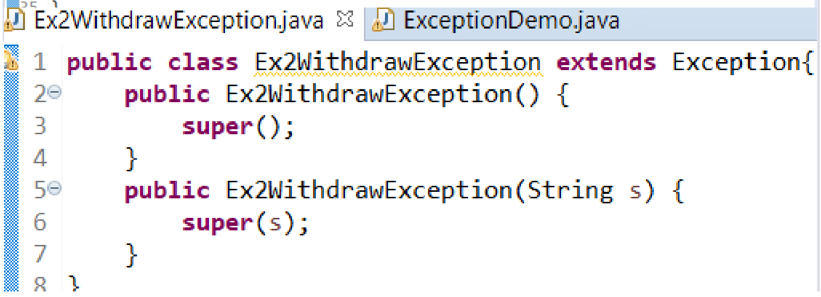
CheckingAccount.java | Ex2WithdrawException.java | ExceptionDemo.java | Teller1.java

```java
public class Teller1 {
    public static void main(String[] args){
        Account acc = new CheckingAccount();
        acc.deposit(1000);
        acc.showBalance();
        try{
            acc.withdraw(5000);
        }catch (Ex2WithdrawException e) {
            System.out.println("you dont have enough money \nexception is "+e);
        }


    }
}
```



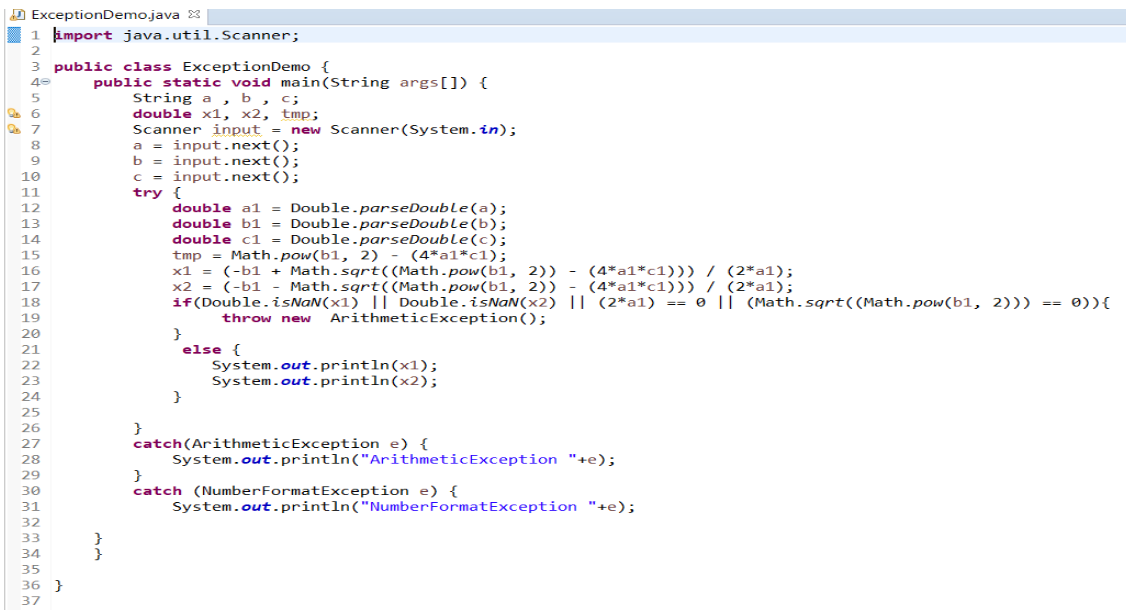
CheckingAccount.java | Ex2WithdrawException.java | ExceptionDemo.java

```java
public class CheckingAccount extends Account{
    private double credit;
    public CheckingAccount() {

    }
    public CheckingAccount(double crredit) {
        this.credit = crredit;
    }
    public CheckingAccount(double amount, double credit){
        this.balance = amount;
        this.credit = credit;
    }

    @Override
    public boolean withdraw(double amount)throws Ex2WithdrawException {
        if(amount < this.balance + this.credit){
            this.balance -= amount;
            if(this.balance < 0){
                this.credit += this.balance;
                this.balance = 0;
            }
            return true;
        }
        else {
            throw new Ex2WithdrawException();
//              return false;
        }
    }
    public void showCredit() {
        System.out.println(credit);
    }
    public double getCredit() {
        return credit;
    }
}
```



Ex2WithdrawException.java | ExceptionDemo.java

```java
public class Ex2WithdrawException extends Exception{
    public Ex2WithdrawException() {
        super();
    }
    public Ex2WithdrawException(String s) {
        super(s);
    }
}
```



ExceptionDemo.java

```java
import java.util.Scanner;

public class ExceptionDemo {
    public static void main(String args[]) {
        String a , b , c;
        double x1, x2, tmp;
        Scanner input = new Scanner(System.in);
        a = input.next();
        b = input.next();
        c = input.next();
        try {
            double a1 = Double.parseDouble(a);
            double b1 = Double.parseDouble(b);
            double c1 = Double.parseDouble(c);
            tmp = Math.pow(b1, 2) - (4*a1*c1);
            x1 = (-b1 + Math.sqrt((Math.pow(b1, 2)) - (4*a1*c1))) / (2*a1);
            x2 = (-b1 - Math.sqrt((Math.pow(b1, 2)) - (4*a1*c1))) / (2*a1);
            if(Double.isNaN(x1) || Double.isNaN(x2) || (2*a1) == 0 || (Math.sqrt((Math.pow(b1, 2))) == 0)){
                throw new  ArithmeticException();
            }
             else {
                System.out.println(x1);
                System.out.println(x2);
            }

        }
        catch(ArithmeticException e) {
            System.out.println("ArithmeticException "+e);
        }
        catch (NumberFormatException e) {
            System.out.println("NumberFormatException "+e);

    }
    }

}
```