

Objectives:

Write Programs that make uses of Exception Handling

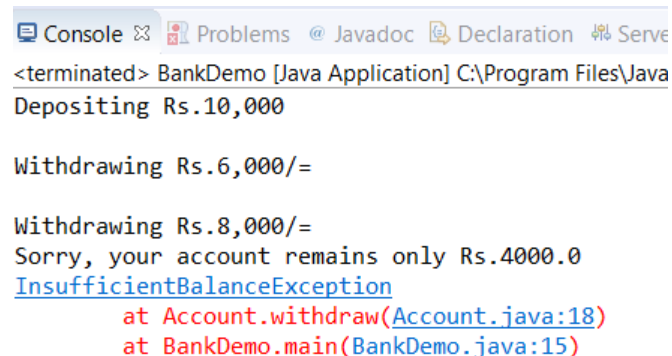
Exercise 1**Compulsory**

- 1) Consider the following **BankDemo** Application to perform deposit and withdraw amount from the customer account. To perform these operations, you should create an **Account** class and validate the withdrawal amount lest make the account **overdue**. You should create custom exception class **"InsufficientBalanceException"**.

The sample **BankDemo** Application main program is given below with sample output. Your implementation should satisfy the same.

```
public class BankDemo {  
  
    public static void main(String[] args) {  
  
        Account account = new Account(123);  
        System.out.println("Depositing Rs.10,000");  
        account.deposit(10000.00);  
  
        try {  
            System.out.println("\nWithdrawing Rs.6,000/=");  
            account.withdraw(6000.00);  
  
            System.out.println("\nWithdrawing Rs.8,000/=");  
            account.withdraw(8000.00);  
  
        } catch (InsufficientBalanceException e) {  
            System.out.println("Sorry, your account remains only Rs." + e.getAmount());  
            e.printStackTrace();  
        }  
    }  
}
```

When you withdraw more than the existing account throw **InsufficientBalanceException**. When you run the program out put should be as follows.



```
<terminated> BankDemo [Java Application] C:\Program Files\Java  
Depositing Rs.10,000  
  
Withdrawing Rs.6,000/=
```

Withdrawing Rs.8,000/=

Sorry, your account remains only Rs.4000.0

[InsufficientBalanceException](#)

at Account.withdraw(Account.java:18)
at BankDemo.main(BankDemo.java:15)

Lab Exercise 7**IT2030 – Object Oriented Programming****Semester 1, 2020**

- Create **InsufficientBalanceException** class and amount should be able to pass through the constructor of this custom exception class
- Create **Account** class that holds **balance** and **Account No.** Implement operations to display existing balance, account number and account number can be assigned through the Constructor
- Implement the **deposit** operation and that increases the existing balance in the account
- Implement the withdraw operation and that reduces the balance with given value. In case if balance is not sufficient **throw InsufficientBalanceException** in the method and you should handle it in the BankDemo Application. You throw this in the withdraw operation as below **throw new InsufficientBalanceException(amount);**

Exercise 2

- Refer the **Question 1)** and Modify the above **BankDemo** class to give the below output

```
Console Problems Javadoc Declaration Servers Data Source Explorer
<terminated> BankDemo2 [Java Application] C:\Program Files\Java\jre1.8.0_20\bin\javaw.exe (A
Depositing Rs.10,000
Please enter amount to be withdrawn = 3000

Withdrawing Rs.3000.0/=
existing amount = 7000.0Please enter amount to be withdrawn = 3000

Withdrawing Rs.3000.0/=
existing amount = 4000.0Please enter amount to be withdrawn = 3000

Withdrawing Rs.3000.0/=
existing amount = 1000.0Please enter amount to be withdrawn = 3000

Withdrawing Rs.3000.0/=
Sorry, your account remains only Rs.2000.0
InsufficientBalanceException
Do you wish to continue? yes/no
    at Account.withdraw(Account.java:18)
    at BankDemo2.continueTransaction(BankDemo2.java:42)
    at BankDemo2.main(BankDemo2.java:12)
yes
Depositing Rs.10,000
Please enter amount to be withdrawn = 12000

Withdrawing Rs.12000.0/=
InsufficientBalanceException
    at Account.withdraw(Account.java:18)
    at BankDemo2.continueTransaction(BankDemo2.java:42)
    at BankDemo2.main(BankDemo2.java:24)
```

(No need to consider the keyboard input validations in your implementation)

- a) In the modified program user should enter the withdrawal amount as keyboard input and this activity should continue as infinite loop until user response for the question “**Do you wish to continue ?**” If user answers as “**no**” program will terminate
- b) You should extend the above exception handling with including **finally block**. In the finally block you should ask the above question “**Do you wish to continue?**”
- c) If user response “**yes**” for the above question **a)** in your program should deposit the same amount for the account and continue the withdrawal process
- d) Make sure you should not duplicate the logics in the program for above modification (Consider OOP concepts)