UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

Informatics Institute of Technology
Department of Computing

Module: 4COSC0010C – Programming Principles 02
Degree Program: BScCS

Tutorial Group: Group A

Module Leader: Mr. Guganathan Poravi

Coursework 01 – Stage 6

Date of submission: 25.03.2018

Student ID - IIT Student ID: 2017091

UoW ID : 16737363/1

Student First Name: Dinithi

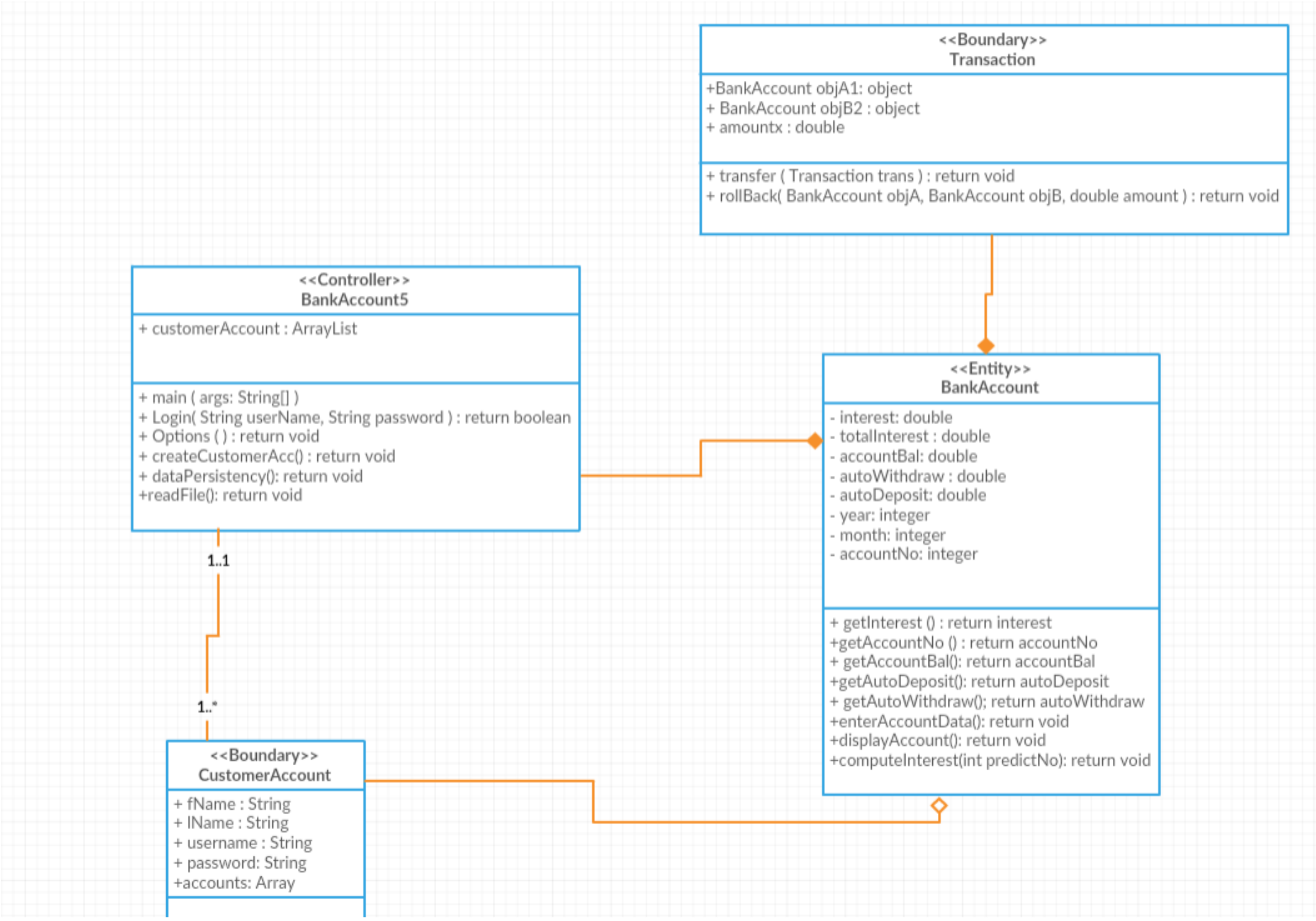
Student Surname : Jayasekara

Contents

1. INTRODUCTION

This is a JAVA console-based application developed for InterBanking Pty. The purpose of this system is to produce a next generation customer and account management system. This application has been developed for employees of the bank to add new customer accounts and bank accounts and is very user friendly serving its purpose to the fullest.

2. DESIGN


<<Boundary>>
Transaction
+BankAccount objA1: object
+ BankAccount objB2 : object
+ amountx : double
+ transfer (Transaction trans) : return void
+ rollBack(BankAccount objA, BankAccount objB, double amount) : return void
<<Controller>>
BankAccount5
+ customerAccount : ArrayList
+ main (args: String[])
+ Login(String userName, String password) : return boolean
+ Options ( ) : return void
+ createCustomerAcc() : return void
+ dataPersistency(): return void
+readFile(): return void
<<Entity>>
BankAccount
- interest: double
- totalInterest : double
- accountBal: double
- autoWithdraw : double
- autoDeposit: double
- year: integer
- month: integer
- accountNo: integer
+ getInterest () : return interest
+getAccountNo () : return accountNo
+ getAccountBal(): return accountBal
+getAutoDeposit(): return autoDeposit
+ getAutoWithdraw(); return autoWithdraw
+enterAccountData(): return void
+displayAccount(): return void
+computeInterest(int predictNo): return void
1..1
1..*
<<Boundary>>
CustomerAccount
+ fName : String
+ lName : String
+ username : String
+ password: String
+accounts: Array

3. IMPLEMENTATION

- A transaction class with transaction object holding bank account objects to transfer from safely and Boolean conditions to observe transaction.
  A rollback method to undo transaction.

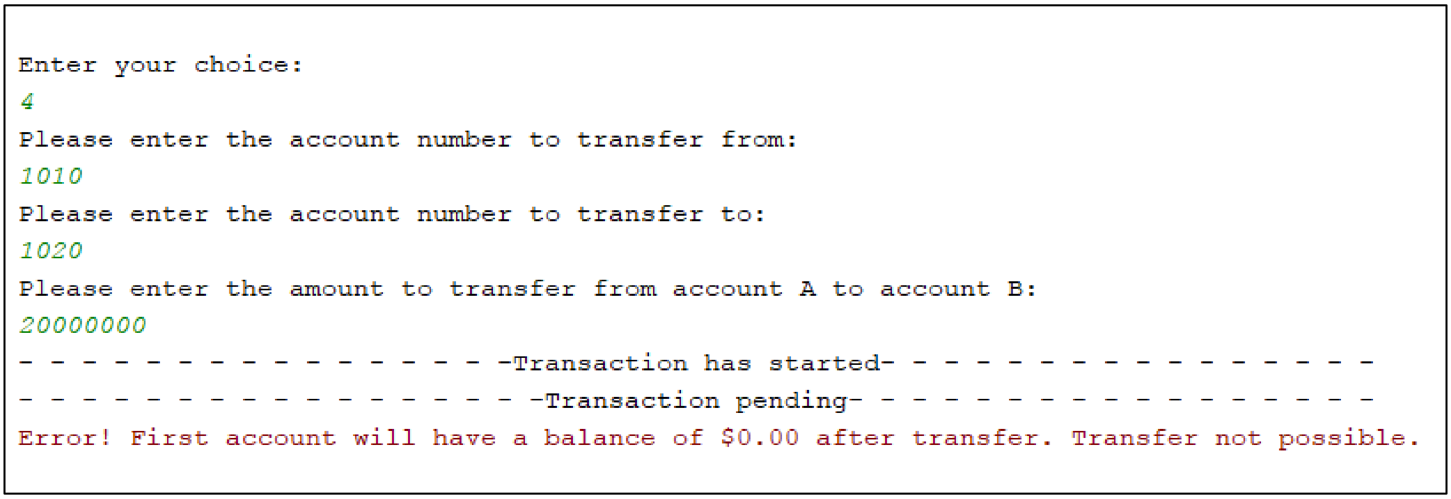
```
package com.company;

public class Transaction extends BankAccount6 {
    BankAccount objA1;
    BankAccount objB2;
    double amountx;

    public Transaction(BankAccount objA, BankAccount objB, double amount) {
        this.objA1 = objA;
        this.objB2 = objB;
        this.amountx = amount;
    }


    public static void transfer(Transaction trans) {
        BankAccount objA = trans.objA1;
        BankAccount objB = trans.objB2;
        double amount = trans.amountx;
        boolean start = false;

        System.out.println ("- - - - - - - - - - - - - - - -Transaction has started- - -
- - - - - - - - - - - - -");
        start = true;


        boolean pending = true;


        if (pending) {
            System.out.println ("- - - - - - - - - - - - - - - - -Transaction pending- -
- - - - - - - - - - - - - - -");
            boolean success = false;
            if (objA.accountBal < amount) {
                System.err.println ("Error! First account will have a balance of $0.00
after transfer. Transfer not possible.");
                System.exit (0);
            } else if ((objB.accountBal + amount) >= 100000) {
                System.err.println ("Warning! Second account has reached the highest
amount that is federally insured.");
            } else {
                if ((objA.accountBal - amount) <= 10) {
                    System.err.println ("Warning! First account has a balance less than
$10.");
                } else {
                    success = true;
                    System.out.println ("- - - - - - - - - - - - - - - -Transaction 
Successful- - - - - - - - - - - - - - - - -");
                }

                System.out.println ("Bank account A balance (before transfer): " + "$" + 
objA.accountBal);
                System.out.println ("Bank account B balance (before transfer): " + "$" + 
objB.accountBal);
                System.out.println ("Transfer amount: " + "$" + amount);
                objA.accountBal -= amount;
                System.out.println ("Bank account A balance after transfer: " + 
objA.accountBal);
                objB.accountBal += amount;
                System.out.println ("Bank account B balance after transfer: " + 
objB.accountBal);
            }
        }
        System.out.println ("Do you want to undo the transaction? Press 1 to undo and 0 
to continue.");
        char roll = '\0';
        roll = sc.next ().charAt (0);
        switch (roll) {
            case '1':
                rollBack (objA, objB, amount);
                break;
            case '0':
                Options ();
                break;
        }
    }

    public static void rollBack(BankAccount objA, BankAccount objB, double amount) {
        System.out.println ("Bank account A balance (before rollback): " + "$" + 
objA.accountBal);
        System.out.println ("Bank account B balance (before rollback): " + "$" + 
objB.accountBal);
        objA.accountBal += amount;
        System.out.println ("Bank account A balance after rollback: " + objA.accountBal);
        objB.accountBal -= amount;
        System.out.println ("Bank account B balance after rollback: " + objB.accountBal);
    }
}
```

- Calling the transfer method in transaction class to the main method:

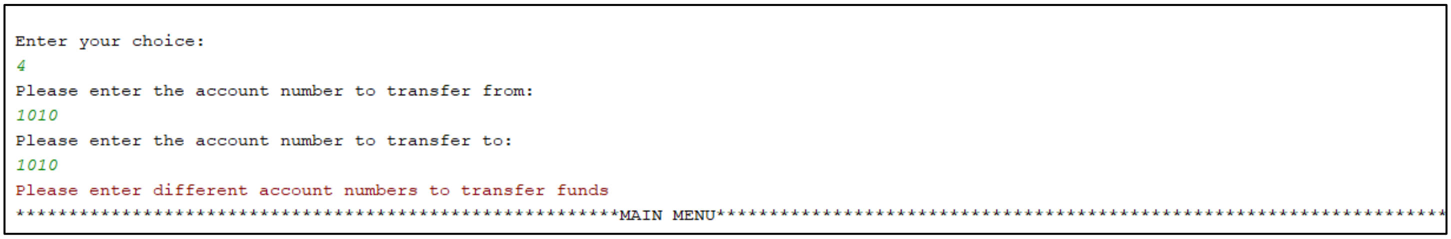
```
case '4':
    System.out.println ("Please enter the account number to transfer from: ");
    int acc_A = sc.nextInt ();

    System.out.println ("Please enter the account number to transfer to: ");
    int acc_B = sc.nextInt ();
    if (acc_A == acc_B) {
        System.err.println ("Please enter different account numbers to transfer funds");
        Options ();
    }
    System.out.println ("Please enter the amount to transfer from account A to account B: ");
    double amount = sc.nextDouble ();


    BankAccount objA = null, objB = null;
    for (com.company.BankAccount6.customerAccount x : customerAccount) {
        for (BankAccount y : x.accounts) {
            if (y != null && y.getAccountNo () == acc_A) {
                objA = y;

            }
            if (y != null && y.getAccountNo () == acc_B) {
                objB = y;
            }
        }
    }
    if (objA == null || objB == null) {
        System.err.println ("invalid account numbers.");
        Options ();
    }
    Transaction newTransaction = new Transaction (objA, objB, amount);
    Transaction.transfer (newTransaction);
    Options ();
    break;
```

4. SCREENSHOTS

- Validation for transfer using flags:

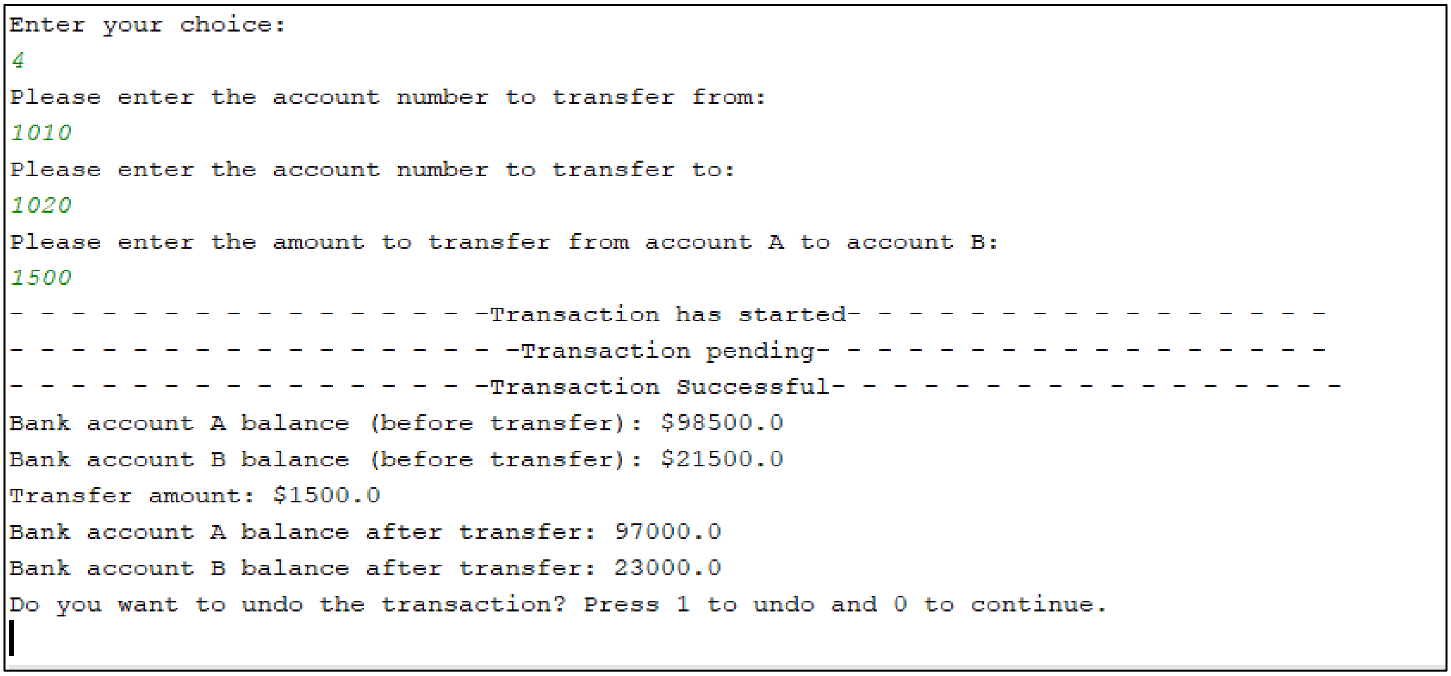


```
Enter your choice:
4
Please enter the account number to transfer from:
1010
Please enter the account number to transfer to:
1020
Please enter the amount to transfer from account A to account B:
20000000
- - - - - - - - - - - - - - - - -Transaction has started- - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - -Transaction pending- - - - - - - - - - - - - - - - -
Error! First account will have a balance of $0.00 after transfer. Transfer not possible.
```



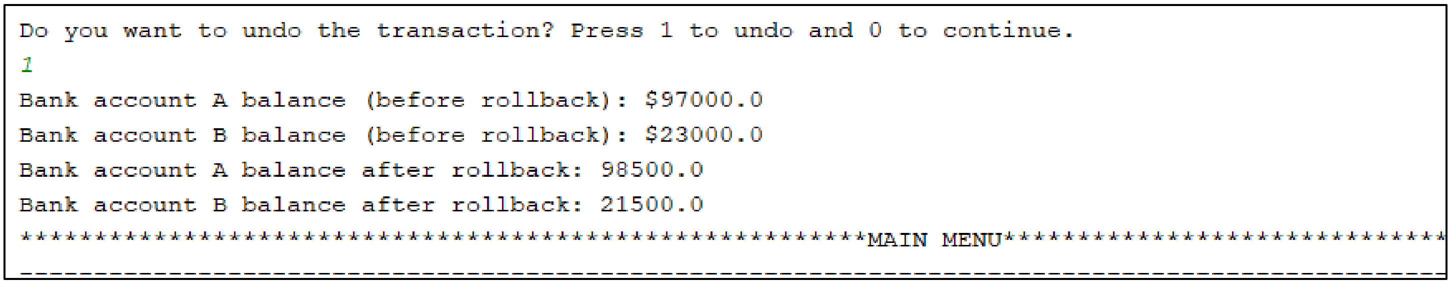
```
Enter your choice:
4
Please enter the account number to transfer from:
1010
Please enter the account number to transfer to:
1010
Please enter different account numbers to transfer funds
************************************************************MAIN MENU*************************************************************************
```

- Successful Transaction:



```
Enter your choice:
4
Please enter the account number to transfer from:
1010
Please enter the account number to transfer to:
1020
Please enter the amount to transfer from account A to account B:
1500
- - - - - - - - - - - - - - - -Transaction has started- - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - -Transaction pending- - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - -Transaction Successful- - - - - - - - - - - - - - - - -
Bank account A balance (before transfer): $98500.0
Bank account B balance (before transfer): $21500.0
Transfer amount: $1500.0
Bank account A balance after transfer: 97000.0
Bank account B balance after transfer: 23000.0
Do you want to undo the transaction? Press 1 to undo and 0 to continue.
```

- RollBack() Method



```
Do you want to undo the transaction? Press 1 to undo and 0 to continue.
1
Bank account A balance (before rollback): $97000.0
Bank account B balance (before rollback): $23000.0
Bank account A balance after rollback: 98500.0
Bank account B balance after rollback: 21500.0
*************************************************MAIN MENU**************************
------------------------------------------------------------------------------------
```

5. CONCLUSION

The transaction class safely handles all transactions and clearly shows the status of the transactions in the transfer() method. Further, the RollBack() method is capable of undo function to undo the transaction if required by user.