/\*\*

\* This class stimulates the profile of customer's with bankaccounts with their name and a number ID

\* @author T03(team 5)

\*

\*/

import java.lang.Double;

public class BankAccount

{

/\*\*

\* Setting initial amounts

\*/

double balance;

double overdraftAmount = 100.0;

Customer newCustomer;

/\*\*

\* Create instance of Customer

\* @param customerName

\* @param CustomerIDnum

\*/

public void Customer(String customerName, int CustomerIDnum)

{

newCustomer = new Customer(customerName,CustomerIDnum);

}

/\*\*

\* Default constructor for BankAccount

\*/

public BankAccount()

{

}

/\*\*

\* Constructor for BankAccount that takes type Customer and double as parameters

\* @param newCustomer

\* @param money

\*/

public BankAccount(Customer newCustomer, double money)

{

Customer(newCustomer.getName(),newCustomer.getID());

balance = money;

}

/\*\*

\* This method adds the desired amount of money to the balance

\* @param money

\*/

public void deposit(double money)

{

if (money < 0 || Double.isNaN(money) || Double.isInfinite(money))

{

System.out.println("Error : You can not deposit a negative amount.");

}

else

{

balance += money;

System.out.println("Your new balance after the deposit is: " +balance);

}

}

/\*\*

\* This method uses a conditional statement to make sure the amount withdrawn is not over the overdraft amount,

\* as well as checking that the current bank account balance is not 0.

\* If none of these conditions are met, then the desired amount of money is subtracted from the balance.

\* @param money

\*/

public void withdraw(double money)

{

if((balance - money) < (0 - overdraftAmount) || Double.isNaN(money) || Double.isInfinite(money))

{

System.out.println("You have reached your over draft limit.");

}

else

{

if(balance - money > 0)

{

balance -= money;

}

System.out.println("Your new balance after the withdrawl is: " + balance);

}

}

/\*\*

\* Returns the value of the balance variable

\* @return

\*/

public double getBalance()

{

return balance;

}

/\*\*

\* Returns the value of the global variable

\* @param overdraftAmount

\*/

protected void setBalance(double amount)

{

balance += amount;

}

public void setOverdraftAmount(double overdraftAmount)

{

if (balance >= 0){

if ((!Double.isInfinite(overdraftAmount) || !Double.isNaN(overdraftAmount)) && overdraftAmount >= -1000){

this.overdraftAmount = overdraftAmount;

}

}

else{

if(balance > -overdraftAmount){

if ((!Double.isInfinite(overdraftAmount) || !Double.isNaN(overdraftAmount)) && overdraftAmount >= -1000){

this.overdraftAmount = overdraftAmount;

}

}

}

}

public Customer getCustomer() {

Customer aCustomer;

return aCustomer = new Customer(newCustomer);

}

public void transfer(double amount, BankAccount toAccount)

{

}

/\*\*

\* This main method is contains the block of code that is to be traced

\* @param args

\*/

public static void main(String[] args)

{

Customer c1 = new Customer("Alison Brown", 123);

BankAccount b1 = new BankAccount(c1, 100.00);

Customer c2 = b1.getCustomer();

c2.setName("Charles Green");

System.out.println(c1);

System.out.println(c2);

Customer c3 = c1;

c3.setName("Eva White");

System.out.println(b1.getCustomer());

System.out.println(c3);

}

}