**CS 342 Report for (Lab 05)**

**Lab05.class**

//Programmer: Fayaz Khan

//Assignment: Lab 5

//Date: September 24, 2015

//Description:Program uses test drivers to test methods in BankAccount class with different

// bank account types

public class Lab05

{

public static void main(String[] args)

{

final static int INITIALBALANCE = 1000; //Initial deposit for both bank account

final static String STYPE = "savings"; //Bank account type "savings"

final static String CTYPE = "checking"; //Bank account type "checking"

finla static String FIRSTHOLDER = "Sidra"; //account owner number one

final static String SECONDHOLDER = "Bushra"; //account owner number two

final static double WITHDRAW$250 = 250; //first withdraw intended to be successful

final static double WITHDRAW$1100 = 1100; //second withdraw intended to be unsuccessful

//Two different bank accounts, sAccount == savings and cAccount == checking

BankAccount sAccount = new BankAccount(INITIALBALANCE,FIRSTHOLDER,STYPE);

BankAccount cAccount = new BankAccount(INITIALBALANCE,SECONDHOLDER,CTYPE);

//Prints the starting balance

System.out.printf("Your balance is currently $%.2f\n",sAccount.GetBalance());

//checks the type of account whether it is savings or checking

System.out.println("This is a " + sAccount.accountType() + " bank account");

//checks the type of account whether it is savings or checking

System.out.println("This is a " + cAccount.accountType() + " bank account");

//Attempting to withdraw from bank account

System.out.printf(String.format("Withdrawl is pending of amount $%.2f\n",WITHDRAW$250));

if(sAccount.Withdraw(WITHDRAW$250)) //Checks if funds are sufficient

System.out.println("Withdrawal Succesful");

else

System.out.println("Withdrawal Unsuccesful, Insufficient funds");

//Displays the current balance of account

sAccount.DisplayBalance();

//Second test Attempting to withdraw from bank account

System.out.printf(String.format("Withdrawl is pending of amount $%.2f\n",WITHDRAW$1100));

if(cAccount.Withdraw(WITHDRAW$1100)) //Checks if funds are sufficient

System.out.println("Withdrawal Succesful");

else

System.out.println("Withdrawal Unsuccesful, Insufficient funds");

//Displays the current balance of account

cAccount.DisplayBalance();

//Displays whether it is a savings or checking account

System.out.println("This is a " + sAccount.accountType() + " bank account");

//Reseting account with a new name and new balance

sAccount.ResetAccount(SECONDHOLDER,INITIALBALANCE);

System.out.println("The savings account is resetting with new name and balance");

System.out.printf("Your balance is currently $%.2f\n",sAccount.GetBalance());

}

}

Khan 2

**BankAccount.class**

//Description:Includes methods to construct objects and utlizing overloading methods.Includes a method

// for display balance.Modifiers to modify the balance, reset the account and withdraw from // account.In this class also includes accesors to get balance and get the account type.

public class BankAccount

{

private String name; // name of account holder

private double balance; // how much money is in account, $

private boolean accountType; // Savings == true, Checking == false

public BankAccount()

// POST: A default BankAccount object is created with name set to a blank and

// and balance set to $0.00

{

this(0.0);

}

public BankAccount(double balance)

// PRE: balance >= 0.00 and balance is in dollars

// POST: A BankAccount object is created with name set to a blank and the class member

// balance set to balance accountType is set to false(Checking)

{

accountType = false;

name = " ";

if(balance >= 0) //validate proposed initial balance

this.balance = balance;

else

this.balance = 0;

}

public BankAccount(double balance,String name,String accountType)

// PRE: balance in dollars and balance >= 0.00

// accountType need to be initialized

// name needs to be initialized

// POST:When a BankAccount object is created, name will set (this.name) and balance

// will set (this.balance). If accountType is "savings" then (this.acccountType)

// is set to true and accountType is "checking" (this.accountType) set to false

{

this.name = name;

if(balance >= 0) //validate proposed initial balance

this.balance = balance;

else

this.balance = 0;

if(accountType.equals("savings")) //checks if String accountType is savings

this.accountType = true;

if(accountType.equals("checking")) //checks if String accountType is checking

this.accountType = false;

}

public String accountType()

// This method checks the object (this.accountType) for either true or false.

// If (this.accountType) is true then returns "savings", is false then returns "checking"

{

if(this.accountType == true)

return "savings";

else

return "checking";

}

public void ResetAccount(String newName, double newBalance)

// PRE: newName has been assigned a value

// && newBalance >= 0.00 and newBalance is in dollars

// POST: This account object is reset with name set to newName

// and balance set to newBalance

{

name = newName; // Match up private variables with parameters

balance = newBalance; // Could do error checking here with an if(balance >= 0)

}

Khan 3

public boolean Withdraw(double amount)

// PRE: amount >= 0.00 and amount is in dollars

// POST: amount is deducted from the balance stored for this account

{

if(balance - amount >= 0) //checks if balance is sufficient for withdraw

{

balance = balance - amount; //updates balance after withdrawing

return true;

}

else

return false;

}

public double GetBalance()

// POST: FCTVAL == current balance of this account in dollars

{

return balance;

}

public void DisplayBalance()

// POST: The current balance of this account has been displayed to the screen

{

System.out.printf("Your balance is currently $%.2f\n", balance);

}

}

Khan 4

**Sample Run**

