Tenzin Choeying

Lab 1

02.04.18

Word Doc

/\*

\* Tenzin Choeying

\* 02.04.18

\* AccountHolder.java

\* Lab 1

\*/

**public** **class** AccountHolder {

**private** **static** **double** *annualInterestRate* = 0.0;

**private** **double** balance = 0.0;

**private** **boolean** below500;

//constructor accepts initial balance.

**public** AccountHolder(**double** newBalance) {

**if** (newBalance<0) { //Error if newBalance is negative

System.***out***.print("ERROR: Balance can not start out negative.\n");

}

**if**(newBalance<500.0) { //if balance is below 500 set below500 to true so under 500 transaction fee isn't applied.

balance = newBalance;

below500 = **true**;

}

**else** { //if newBalance is not negative and above 500, set balance

balance = newBalance;

below500 = **false**;

}

}

//deposit accepts a double amount and updates balance by adding the amount to it

**public** **void** deposit(**double** depositAmt) {

balance += depositAmt;

**if** (balance>=500.0) {

below500 = **false**;

}

}

//withdrawal accepts a double withdrawal amount and updates balance by subtracting the withdrawal amount from the balance

**public** **void** withdrawal(**double** withdrawalAmt) {

**if** (withdrawalAmt>(balance-100.0)) { //Error if withdrawal leaves balance with less than 100

System.***out***.print("ERROR: Withdrawal can not decrease balance to less than $100.00.\n");

}

**else** **if** (withdrawalAmt>(balance-500.0) && !below500) { //transaction fee if withdrawal leaves balance below 500 and balance wasn't already below 500

System.***out***.print("This withdrawl will leave a balance of less than $500.00. A one time transaction fee of $50.00 will be applied.\n");

balance -= 50.00 + withdrawalAmt;

below500 = **true**;

}

**else** { //if withdrawal leaves balance above 500, change balance to reflect the withdrawal

balance -= withdrawalAmt;

}

}

//monthlyInterest method updates the account holders balance by adding the interest amount

**public** **void** monthlyInterest() {

balance += balance \* (*annualInterestRate* / 12.0);

}

//modifyMonthlyInterest updates annualInterestRate with updated interest rate

**public** **static** **void** modifyMonthlyInterest(**double** rateUpdate) {

**if** (rateUpdate>=0.0 && rateUpdate<=1.0) { //rateUpdate is between 0.0 and 1.0

*annualInterestRate* = rateUpdate;

}

**else** { //if rateUpdate is not between 0.0 and 1.0, error

System.***out***.print("ERROR: Interest rate must be between 0.0 and 1.0 inclusive\n");

}

}

//toString formats and returns the balance to be in USD

**public** String toString() {

**return** String.*format*("$%.2f", balance);

}

//get balance returns a double with the value of balance

**public** **double** getBalance() {

**return** balance;

}

}

/\*

\* Tenzin Choeying

\* 02.04.18

\* AccountHolderTest.java

\* Lab 1

\*/

**import** java.text.SimpleDateFormat;

**import** java.util.Calendar;

**import** java.util.Scanner;

**public** **class** AccountHolderTest {

**public** **static** **void** main(String[]args) {

Scanner scan = **new** Scanner(System.***in***);

**boolean** valid = **false**;

**double** initialBalance=0.0;

//Ask user for initial Balance

**while**(!valid) {

System.***out***.print("Please enter an Initial Account Balance: ");

**if** (scan.hasNextDouble()) {

initialBalance = scan.nextDouble();

**if**(initialBalance>0) {

valid = **true**;

}

**else** {

**new** AccountHolder(initialBalance);

}

}

**else** {

System.***out***.println("Initial Blanace must be a number.");

scan.next();

}

}

AccountHolder ah = **new** AccountHolder(initialBalance);

//Ask user for deposit amount

valid = **false**;

**while**(!valid) {

System.***out***.print("Please Enter a deposit amount: ");

**if** (scan.hasNextDouble()) {

ah.deposit(scan.nextDouble());

valid = **true**;

}

**else** {

System.***out***.println("Deposit amount must be a number.");

scan.next();

}

}

//Ask user for withdrawal amount

valid = **false**;

**while**(!valid) {

System.***out***.print("Please enter a withdrawal amount: ");

**if**(scan.hasNextDouble()) {

**double** withdrawal = scan.nextDouble();

**if**((ah.getBalance()-withdrawal)<100) {

ah.withdrawal(withdrawal);

}

**else** {

ah.withdrawal(withdrawal);

valid = **true**;

}

}

**else** {

System.***out***.println("Withdrawal amount must be a number.");

scan.next();

}

}

//set initial interest to 4%

AccountHolder.*modifyMonthlyInterest*(0.04);

//output monthly interest over 12 months

System.***out***.println("Balance Reports");

System.***out***.println("Montly balances for one year at 4% interest");

System.***out***.println("Balances:");

System.***out***.printf("%-15.15s %-30.30s%n","Month","Balance w. Interest");

System.***out***.printf("%-15.15s %-30.30s%n","Base",ah);

**for**(**int** i=1;i<=12;i++) {

ah.monthlyInterest();

System.***out***.printf("%-15.15s %-30.30s%n","Month " + i + ":",ah);

}

//update interest amount to 4%

AccountHolder.*modifyMonthlyInterest*(0.04);

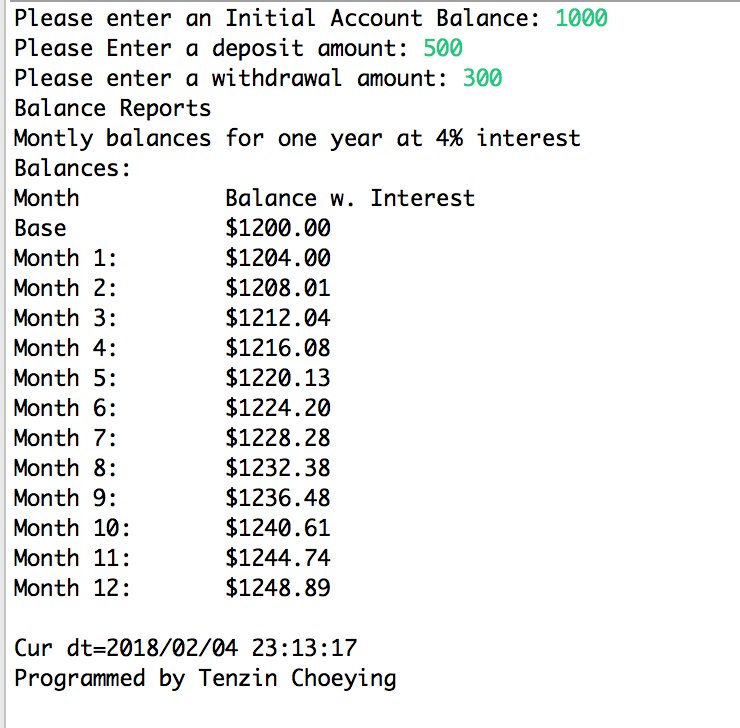
//time stamp and name

String timeStamp = **new** SimpleDateFormat("yyyy/MM/dd HH:mm:ss").format(Calendar.*getInstance*().getTime());

System.***out***.println("\nCur dt=" + timeStamp + "\nProgrammed by Tenzin Choeying\n");

}

}



**Showing Error Message if balance is below $100**

