**package** com.htc.homework5;

**import** java.util.ArrayList;

**public** **class** Bank **implements** IBankServiceProvider {

**public** ArrayList<BankAccount> bankAccounts = **new** ArrayList<>();

**static** **int** *lastAssignedNo* = 0;

**public** ArrayList<BankAccount> getBankAccounts() {

**return** bankAccounts;

}

**public** Bank(ArrayList<BankAccount> bankAccounts) {

**super**();

**this**.bankAccounts = bankAccounts;

}

**public** BankAccount checkAccount(String accountNo) {

BankAccount Bank1 = **new** BankAccount();

Bank1 = **null**;

**for** (BankAccount acctlist:**this**.bankAccounts) {

**if** (acctlist.getAccountNo().equals(accountNo)) {

Bank1 = acctlist;

**break**;

}

}

**return** Bank1;

}

**public** **double** getBalance(BankAccount account) {

BankAccount Bank1 = **new** BankAccount();

Bank1 = **this**.checkAccount(account.getAccountNo());

**if** (Bank1.equals(**null**))

**return** -1.0;

**else**

**return** Bank1.getBalance();

}

@Override

**public** **boolean** depositMoney(BankAccount account, **double** amount) **throws** InvalidAmountException {

**if** (amount <= 0)

{ String E1 = "Amount less than or equal to zero can't be deposited";

InvalidAmountException ida = **new** InvalidAmountException(E1);

**throw** ida;

}

BankAccount Bank1 = **new** BankAccount();

Bank1 = **this**.checkAccount(account.getAccountNo());

**if** (Bank1.equals(**null**))

**return** **false**;

**else** {

Bank1.setBalance(Bank1.getBalance() + amount);

**return** **true**;

}

}

@Override

**public** **boolean** withdrawMoney(BankAccount account, **double** amount)

**throws** InvalidAmountException, InsufficientFundException {

**if** (amount <= 0)

{ String E1 = "Amount less than or equal to zero can't be withdrawn";

InvalidAmountException ide = **new** InvalidAmountException(E1);

**throw** ide;

}

BankAccount Bank1 = **new** BankAccount();

Bank1 = **this**.checkAccount(account.getAccountNo());

**if** (Bank1.equals(**null**))

**return** **false**;

**else** {

**if** (Bank1.getBalance() < amount)

{ String E1 = "Insufficent funds to withdraw";

InsufficientFundException ide = **new** InsufficientFundException(E1);

**throw** ide;

}

Bank1.setBalance(Bank1.getBalance() - amount);

**return** **true**;

}

}

**public** **boolean** transferMoney(BankAccount fromAccount, BankAccount toAccount, **double** amount)

**throws** InvalidAmountException, InsufficientFundException {

**if** (amount <= 0)

{ String E1 = "Amount less than or equal to zero can't be transferred";

InvalidAmountException ide = **new** InvalidAmountException(E1);

**throw** ide;

}

BankAccount fromAcct = **new** BankAccount();

fromAcct = **this**.checkAccount(fromAccount.getAccountNo());

BankAccount toAcct = **new** BankAccount();

toAcct = **this**.checkAccount(toAccount.getAccountNo());

**double** fromBalance = fromAcct.getBalance();

**if** (fromBalance < amount)

{ String E1 = "Insuffient fund to transfer";

InsufficientFundException ide = **new** InsufficientFundException(E1);

**throw** ide;

}

**if** (fromAcct.equals(**null**) || toAcct.equals(**null**) || fromBalance < amount)

**return** **false**;

**else** {

toAcct.setBalance(toAcct.getBalance() + amount);

fromAcct.setBalance(fromAcct.getBalance() - amount);

**return** **true**;

}

}

**public** **void** AddAccount(BankAccount bankAccount) {

bankAccounts.add(bankAccount);

}

}

**package** com.htc.homework5;

**import** java.util.ArrayList;

**public** **class** BankAccount {

**private** String accountNo;

**private** String accountName;

**private** **double** balance;

**public** String getAccountNo() {

**return** accountNo;

}

**public** **void** setAccountNo(String accountNo) {

**this**.accountNo = accountNo;

}

**public** String getAccountName() {

**return** accountName;

}

**public** **void** setAccountName(String accountName) {

**this**.accountName = accountName;

}

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

**return** balance;

}

**public** **void** setBalance(**double** balance) {

**this**.balance = balance;

}

**public** BankAccount() {

**super**();

**this**.accountNo = "0";

**this**.accountName = " ";

**this**.balance = 0.00;

}

**public** BankAccount(String accountNo, String accountName) {

**super**();

**this**.accountNo = accountNo;

**this**.accountName = accountName;

**this**.balance=1000.00;

}

**public** BankAccount(String accountName, **double** balance) {

**super**();

**this**.accountNo = String.*valueOf*(Bank.*lastAssignedNo* + 1);

**this**.accountName = accountName;

**this**.balance=balance;

Bank.*lastAssignedNo* = Integer.*parseInt*(**this**.accountNo);

}

**public** String toString() {

**return** "BankAccount [accountNo=" + accountNo + ", accountName=" + accountName + ", balance=" + balance

+ ", getAccountNo()=" + getAccountNo() + ", getAccountName()=" + getAccountName() + ", getBalance()="

+ getBalance() + ", getClass()=" + getClass() + ", hashCode()=" + hashCode() + ", toString()="

+ **super**.toString() + "]";

}

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

BankAccount Bank1;

ArrayList<BankAccount> AccountsList = **new** ArrayList<>();

Bank1 = **new** BankAccount("First",7000);

AccountsList.add(Bank1);

Bank Bank = **new** Bank(AccountsList);

BankAccount Bank2 = **new** BankAccount("Second",5000);

Bank.AddAccount(Bank2);

BankAccount Bank3 = **new** BankAccount("Third",15000);

Bank.AddAccount(Bank3);

System.***out***.println("CheckAccount 2: " + Bank.checkAccount("2"));

System.***out***.println("Get Balance from Account1 (7000): " + Bank.getBalance(Bank1));

**try**

{

**if** (Bank.depositMoney(Bank2,2000) == **true**)

{

System.***out***.println("Deposit 2,000 into Account2 is successful ");

System.***out***.println("Balance of account 2 after deposit : " + Bank.getBalance(Bank2));

}

**if** (Bank.withdrawMoney(Bank3,6000) == **true**)

{

System.***out***.println("Withdraw 6,000 from Account3 is successful");

System.***out***.println("Balance of account 3 after withdraw : " + Bank.getBalance(Bank3));

}

**if** (Bank.transferMoney(Bank1,Bank2,-1000) == **true**)

{

System.***out***.println("Transfer 2000 from Bank1 to Bank2 (7000-2000, 10000+2000) is successful");

System.***out***.println("Balance after transfer, account 1 and account 2 : " + Bank.getBalance(Bank1) + "and" + Bank.getBalance(Bank2));

}

}

**catch** (InvalidAmountException exp)

{

System.***out***.println(exp.getErrorMessage());

exp.printStackTrace();

}

**catch** (InsufficientFundException exp1)

{

System.***out***.println(exp1.getErrorMessage());

exp1.printStackTrace();

}

**catch** (Exception exp2)

{

System.***out***.println("Common Exception");

exp2.printStackTrace();

}

**finally**

{

System.***out***.println("Finally block!!");

}

}

}

**package** com.htc.homework5;

**public** **interface** IBankServiceProvider {

BankAccount checkAccount(String accountNo);

**double** getBalance( BankAccount account);

**boolean** depositMoney(BankAccount account, **double** amount) **throws** InvalidAmountException;

**boolean** withdrawMoney(BankAccount account, **double** amount) **throws** InvalidAmountException,InsufficientFundException;

**boolean** transferMoney(BankAccount fromAccount, BankAccount toAccount, **double** amount) **throws** InvalidAmountException,InsufficientFundException;

}

**package** com.htc.homework5;

**public** **class** InvalidAmountException **extends** Exception {

**public** String errorMessage = " ";

**public** InvalidAmountException(String errorMessage) {

**super**();

**this**.errorMessage = errorMessage;

}

**public** String getErrorMessage() {

**return** errorMessage;

}

**public** **void** setErrorMessage(String errorMessage) {

**this**.errorMessage = errorMessage;

}

}

**package** com.htc.homework5;

**public** **class** InsufficientFundException **extends** Exception {

**public** String errorMessage=" ";

**public** InsufficientFundException(String errorMsg) {

**super**();

**this**.errorMessage = errorMsg;

}

**public** String getErrorMessage() {

**return** errorMessage;

}

**public** **void** setErrorMessage(String errorMsg) {

**this**.errorMessage = errorMsg;

}

}