// https://github.com/mannanarama/CS-256

// I implementing everything, but the code still has bugs.

#include "stdafx.h"

#include <iostream>

using namespace std;

double withdrawAmount;

double depositAmount;

int withdrawCounter = 0;

class BankAccount

{

public:

double balance;

int deposits;

int withdrawals;

double annualInterestRate;

double monthlyServiceCharge;

void setBalance(double b)

{

balance = b;

}

void setDeposit(int d)

{

deposits = d;

}

void setWithdraw(int w)

{

withdrawals = w;

}

void setAnnualInterestRate(double a)

{

annualInterestRate = a;

}

void setMsc(double m)

{

monthlyServiceCharge = m;

}

double getBalance();

int getDeposit();

int getWithdraw();

double getAnnualInterestRate();

double getMsc();

virtual void deposit(double);

virtual void withdraw(double);

virtual void calcInt();

virtual void monthlyProc();

};

double BankAccount::getBalance() { return balance; }

int BankAccount::getDeposit() { return deposits; }

int BankAccount::getWithdraw() { return withdrawals; }

double BankAccount::getAnnualInterestRate()

{

return annualInterestRate;

}

double BankAccount::getMsc() { return monthlyServiceCharge; }

void BankAccount::deposit(double amtDep)

{

balance += amtDep;

deposits++;

}

void BankAccount::withdraw(double amtWd)

{

balance -= amtWd;

withdrawals++;

}

void BankAccount::calcInt()

{

double monthlyInterestRate = annualInterestRate / 12;

double monthlyInterest = balance \* monthlyInterestRate;

balance = +monthlyInterest;

}

void BankAccount::monthlyProc()

{

balance = -monthlyServiceCharge;

calcInt();

withdrawals = 0;

deposits = 0;

monthlyServiceCharge = 0;

}

class SavingsAccount : public BankAccount

{

public:

bool active;

void status();

void withdraw();

void deposit();

void monthlyProc();

};

void SavingsAccount::status()

{

if (balance <25) {

active = false;

}

else { active = true; }

}

void SavingsAccount::withdraw()

{

if (active = true)

{

BankAccount::withdraw(withdrawAmount);

}

status();

}

void SavingsAccount::deposit()

{

BankAccount::deposit(depositAmount);

status();

}

void SavingsAccount::monthlyProc()

{

if (withdrawals > 4)

{

monthlyServiceCharge = +withdrawals - 4;

}

BankAccount::monthlyProc();

status();

}

class CheckingAccount : public BankAccount

{

public:

void withdraw();

void monthlyProc();

};

void CheckingAccount::withdraw()

{

int serviceCharge = 15;

if (getBalance() < 0)

{

}

withdrawCounter++;

}

void CheckingAccount::monthlyProc()

{

monthlyServiceCharge = 5 + (.1 \* withdrawCounter);

BankAccount::monthlyProc();

}

int main() {

BankAccount ba;

SavingsAccount sa;

CheckingAccount ca;

int amount;

int choice;

ba.setBalance(500);

cout << "This is C++ Bank\n";

cout << "Your balance is $" << ba.getBalance() << endl;

cout << "Choose 1 for Savings or 2 for Checking\n";

cin >> choice;

if (choice == 1)

{

do {

cout << "Press 1 to withdraw, 2 to deposit, Insert 0 to skip.\n";

cin >> amount;

if (amount == 1)

{

sa.withdraw();

}

if (amount == 2)

{

sa.deposit();

}

} while (amount != 0);

}

if (choice == 2)

{

do {

cout << "Press 1 to withdraw, 2 to deposit, Press 0 to skip.\n";

cin >> amount;

if (amount == 1)

{

sa.withdraw();

}

if (amount == 2)

{

sa.deposit();

}

} while (amount != 0);

}

}