// MATEO S PEREZ OLMEDO

// CECS 282 LAB 2 - 1

#include <iostream>

using namespace std;

class Account{

private:

double balance; //declare our variable that will hold the balance

const double PENALTY = 20; //and the penalty constant

public:

Account(double balance){ //constructor to add the balance to the obj

this->balance = balance;

}

void deposit(double money){ //this method adds the money variable to the

balance += money; //total balance

}

void withdraw(double money){ //this method withdraws money from the total

if (money > balance){ //balance and if you withdraw more than you have

balance -= PENALTY; //it takes away $20

cout << "You tried to withdraw more money than you had, ";

cout << "we will deduce 20$ from your balance."<<endl;

}

else{

balance -= money;

}

}

double get\_balance(){ //this getter returns the balance of the obj

return balance;

}

};

int main(){

Account my\_account(100); // Set up my account with $100

my\_account.deposit(50);

my\_account.withdraw(175); // Penalty of $20 will apply

my\_account.withdraw(25);

cout << "Account balance: " << my\_account.get\_balance() << "\n";

my\_account.withdraw(my\_account.get\_balance()); // withdraw all

cout << "Account balance: " << my\_account.get\_balance() << "\n";

return 0;

}

// MATEO S PEREZ OLMEDO

// CECS 282 LAB 2 - 2

#include <iostream>

using namespace std;

class Account

{

private:

double balance; //declare and initialize some variables we will

const double PENALTY = 20; //need later in the functions

double doubleInvestment;

double annualInterestRate;

double monthlyInterestRate;

int month = 0;

public:

Account(double balance){ //constructor to add the balance to the obj

this->balance = balance;

}

Account(double balance, double annualInterestRate){ //contructor that

this->balance = balance; //takes in 2 variables

this->annualInterestRate = annualInterestRate; //for the compound interest

this->monthlyInterestRate = (annualInterestRate/12)/100;

//and sets up the monthly

this->doubleInvestment = balance\*2; //rate and investment x 2

}

void deposit(double money){ //this method adds the money variable to the

balance += money; //total balance

}

void withdraw(double money){ //this method withdraws money from the total

if (money > balance){ //balance and if you withdraw more than you have

balance -= PENALTY; //it takes away $20

cout << "You tried to withdraw more money than you had, ";

cout << "we will deduce 20$ from your balance."<<endl;

}

else{

balance -= money;

}

}

void compound\_monthly\_rate(){ //find the months to get the double

while(balance < doubleInvestment){//of the investments

balance += balance \* monthlyInterestRate;

month++;

}

}

double get\_balance(){ //this getter returns the balance of the obj

return balance;

}

int get\_months(){ //this getter returns the amount of months to double

return month; //the invstment

}};

int main()

{

char input; //will store variable to tell us if we want to do a custom investment

double invest = 0;

double rate = 0;

cout<<"The current account balance is $10,000.00 and has an annual interest rate of ;

cout<<6%"<<endl;

cout<<"Would you like to enter your own balace and rate? (Enter Y or N)"<<endl;

cin >> input;

if(input == 'Y' || input == 'y'){ //if statement to see if we will do a custom investment

cout<<"Please enter your initial balance"<<endl;

cin>>invest;

cout<<"Please enter the annual interest rate"<<endl;

cin>>rate;

Account my\_account(invest,rate); //create the account object with the custom

//variables

my\_account.compound\_monthly\_rate();//get the monthly rate and time

cout.precision(2);

cout<<"It will take "<<my\_account.get\_months();//dislay the months it will take

//to double

cout<<" months to double your investment of $"<<fixed<<invest<<endl;

}

else{ //else we use our default values given in the instructions

invest = 10000;

rate = 6;

Account my\_account(invest,rate); //create the account object with the custom

//variables

my\_account.compound\_monthly\_rate();//get the monthly rate and time

cout.precision(2);

cout<<"It will take "<<my\_account.get\_months();//dislay the months it will take

//to double

cout<<" months to double your investment of $"<<fixed<<invest<<endl;

}

return 0;}

// MATEO S PEREZ OLMEDO

// CECS 282 LAB 2 - 3

#include <iostream>

using namespace std;

class Account

{

private:

double balance;//declare and initialize some variables we will

const double PENALTY = 20; //need later in the functions

double doubleInvestment;

double annualInterestRate;

double monthlyInterestRate;

int month = 0;

public:

Account(double balance){ //constructor to add the balance to the obj

this->balance = balance;

}

Account(double balance, double annualInterestRate){ //contructor that

this->balance = balance; //takes in 2 variables

this->annualInterestRate = annualInterestRate; //for the compound interest

this->monthlyInterestRate = (annualInterestRate/12)/100;

//and sets up the monthly

this->doubleInvestment = balance\*2; //rate and investment x 2

}

void deposit(double money){ //this method adds the money variable to the

balance += money; //total balance

}

void withdraw(double money){ //this method withdraws money from the total

if (money > balance){ //balance and if you withdraw more than you have

balance -= PENALTY; //it takes away $20

cout << "You tried to withdraw more money than you had, ";

cout << "we will deduce 20$ from your balance."<<endl;

}

else{

balance -= money;

}

}

void compound\_monthly\_rate(){ //find the months to get the double

while(balance < doubleInvestment){//of the investments

balance += balance \* monthlyInterestRate;

month++;

}

}

double get\_balance(){ //this getter returns the balance of the obj

return balance;

}

int get\_months(){ //this getter returns the amount of months to double

return month; //the invstment

}

};

class Bank

{

private:

const string SAVE = "S";//create variables that will be used to compare

const string CHECK = "C";//to see wich type of account is used

Account savings; //create the account obj

Account checkings;

public:

Bank() : savings(0), checkings(0){ //pass in the objs parameters

}

void deposit(double amount, string account){

if (account == SAVE){ //if the string passed is an S we do the

savings.deposit(amount);//opertation on the savings account

}

else{ //if it is a C we do it on the checkings

checkings.deposit(amount);

}

}

void withdraw(double amount, string account){

if (account == SAVE){//if the string passed is an S we do the

savings.withdraw(amount);//opertation on the savings account

}

else{//if it is a C we do it on the checkings

checkings.withdraw(amount);

}

}

void transfer(double amount, string account){

if (account == SAVE){//if the string passed is an S we do the

if(savings.get\_balance() < amount){//opertation on the savings account

savings.withdraw(amount); // if the amount withdrawn exceeds the

// balance we withdraw $20

else{

checkings.deposit(amount);

savings.withdraw(amount);

}

}

else{//if it is a C we do it on the checkings

if(checkings.get\_balance() < amount){// if the amount withdrawn exceeds

//the balance

checkings.withdraw(amount);//we withdraw $20

}

else{

savings.deposit(amount);

checkings.withdraw(amount);

}

}

}

void print\_balances(){//this method prints out the two balances

cout.precision(2);

cout << "Checkings account balance: $" << fixed << checkings.get\_balance() << endl;

cout << "Savings account balance: $" << fixed << savings.get\_balance() << endl;

}

};

int main(){

Bank my\_bank;

cout << "Inital bank balances: \n";

my\_bank.print\_balances(); /\* set up empty accounts \*/

cout << "Adding some money to accounts: \n";

my\_bank.deposit(1000, "S"); /\* deposit $1000 to savings \*/

my\_bank.deposit(2000, "C"); /\* deposit $2000 to checking \*/

my\_bank.print\_balances();

cout << "Taking out $1500 from checking,and moving $200 from";

cout << " savings to checking.\n";

my\_bank.withdraw(1500, "C"); /\* withdraw $1500 from checking \*/

my\_bank.transfer(200, "S"); /\* transfer $200 from savings \*/

my\_bank.print\_balances();

cout << "Trying to transfer $900 from Checking.\n";

my\_bank.transfer(900,"C");

my\_bank.print\_balances();

cout << "trying to transfer $900 from Savings.\n";

my\_bank.transfer(900,"S");

my\_bank.print\_balances();

return 0; }