#include<iostream>

#include <iomanip>

#include <string>

#include <fstream>

#include <sstream>

#include "antique.h"

#include "merchant.h"

#include "MerchantGuild.h"

using namespace std;

MerchantGuild::MerchantGuild(int size) {

if (size < 1) {

guildSize = 10;

}

else {

guildSize = size;

}

Merchant\* merc1 = new Merchant[guildSize];

members = merc1;

numMem = 0;

}

MerchantGuild::MerchantGuild() {

guildSize = 10;

Merchant\* merc1 = new Merchant[guildSize];

members = merc1;

numMem = 0;

}

void MerchantGuild::addMember(Merchant other) {

if ((numMem == guildSize)) {

cout << "Guild is full";

}

else {

Merchant\* my\_arr = new Merchant[guildSize + 1];

for (int i = 0; i < numMem; i++) {

my\_arr[i] = members[i];

}

my\_arr[numMem] = other;

numMem++;

delete[] members;

members = my\_arr;

}

}

MerchantGuild::MerchantGuild(const MerchantGuild& origClass){

guildSize = origClass.guildSize;

numMem = origClass.numMem;

members = new Merchant[origClass.numMem];

for (int i = 0; i < origClass.guildSize; i++) {

members[i] = origClass.members[i];

}

}

MerchantGuild::~MerchantGuild() {

delete[] members;

}

MerchantGuild MerchantGuild::operator= (MerchantGuild& mercguild) {

guildSize = mercguild.guildSize;

numMem = mercguild.numMem;

for (int i = 0; i < guildSize; i++) {

mercguild.members[i] = members[i];

}

return \*this;

}

Merchant\* MerchantGuild::getMembers() {

return members;

}

#include<iostream>

#include <iomanip>

#include <string>

#include <fstream>

#include <sstream>

#include "antique.h"

#include "merchant.h"

Merchant::Merchant() { // Default constructor

numAnt = 0;

Antique\* antAnt = new Antique[numAnt];

int\* quantities = new int[numAnt];

for (int i = 0; i < numAnt; i++) {

antAnt=nullptr;

quantities[i] = 10;

}

antAnt = antAnt;

}

Merchant::~Merchant() {

delete[] antAnt;

delete[] quantities;

}

Merchant::Merchant(float rev) {

revenue = rev;

numAnt = 0;

Antique\* antAnt = new Antique[numAnt];

int\* quantities = new int[numAnt];

for (int i = 0; i < numAnt; i++) {

antAnt[i].setName(nullptr);

antAnt[i].setPrice(0);

quantities[i] = 10;

}

}

Merchant::Merchant(const Merchant& origClass) {

numAnt = origClass.numAnt;

revenue = origClass.revenue;

antAnt = new Antique[origClass.numAnt];

quantities = new int[origClass.numAnt];

for (int i = 0; i < origClass.numAnt; i++) {

antAnt[i]=origClass.antAnt[i];

quantities[i] = origClass.quantities[i];

}

}

Merchant::Merchant(Antique antiques[], int quantities2[],int size) {

numAnt = 0;

int\* quantities = new int[antsize];

for (int i = 0; i < antsize; i++) {

antAnt[i].setName(antiques[i].getName());

antAnt[i].setPrice(antiques[i].getPrice());

quantities[i] = quantities2[i];

}

revenue = 0;

}

void Merchant::haggle() {

for (int i = 0; i < numAnt; i++) {

antAnt[i].setPrice(antAnt[i].getPrice() - (antAnt[i].getPrice() / 10));

}

}

void Merchant::printMenu() {

string name;

for (int i = 0; i < numAnt; i++) {

name = antAnt[i].getName();

cout << i + 1 << ") " << antAnt[i].getName() << ": $" << fixed << setprecision(2) << antAnt[i].getPrice() << endl;

}

}

void Merchant::selectAntique(float &budget) {

float ordPrice;

int itemnum;

cout << "Enter item number:" << endl;

cin >> itemnum;

itemnum--;

ofstream outFS;

outFS.open("log2.txt", ios::app);

ordPrice = antAnt[itemnum].getPrice();

if (quantities[itemnum] ==0) {

cout << "Sorry!Antique is out of stock." << endl;

}

if (ordPrice <= budget) {

outFS << "Sold " << antAnt[itemnum].getName() << " for $" << fixed << setprecision(2) << ordPrice << endl;

revenue += ordPrice;

budget -= ordPrice;

}

else {

cout << "Insufficient funds." << endl << endl;

}

}

void Merchant::leave(float budget) {

ofstream outFS;

outFS.open("log2.txt", ios::app);

outFS << "Total revenue: $" << fixed << setprecision(2) << revenue << endl;

outFS << "Remaining budget: $" << budget;

}

Merchant Merchant::operator=(Merchant& merc) {

revenue = merc.revenue;

for (int i = 0; i < numAnt; i++) {

antAnt[i].setName(merc.antAnt[i].getName());

antAnt[i].setPrice(merc.antAnt[i].getPrice());

quantities[i] = merc.quantities[i];

}

return \*this;

}

bool Merchant::operator==(const Merchant& merc) {

int count = 0;

for (int i = 0; i < numAnt; i++) {

if((merc.antAnt[i].getName()== antAnt[i].getName())&&(merc.antAnt[i].getPrice()== antAnt[i].getPrice())&&(merc.quantities[i] = quantities[i])){

count++;

}

}

return

(count== numAnt);

}

void Merchant::addAntique(Antique ant, int quantity) {

Antique\* my\_arr = new Antique[numAnt +1];

int\* my\_quan = new int[numAnt +1];

for (int i = 0; i < numAnt; i++) {

my\_arr[i] = antAnt[i];

my\_quan[i] = quantities[i];

}

my\_arr[numAnt] = ant;

my\_quan[numAnt] = quantity;

numAnt++;

delete[] antAnt;

delete[] quantities;

antAnt = my\_arr;

quantities = my\_quan;

}

#ifndef MERCHANTGUILD\_H

#define MERCHANTGUILD\_H

#include<iostream>

#include "antique.h"

#include "merchant.h"

#include <string>

using namespace std;

class MerchantGuild {

public:

MerchantGuild();

MerchantGuild(int size);

MerchantGuild(const MerchantGuild& origClass);

~MerchantGuild();

void addMember(Merchant other);

MerchantGuild operator= (MerchantGuild& mercguild);

Merchant\* getMembers();

private:

Merchant\* members;

int guildSize;

int numMem;

};

#endif

#ifndef MERCHANT\_H

#define MERCHANT\_H

#include<iostream>

#include <string>

using namespace std;

class Merchant {

public:

Merchant();

Merchant(Antique antiques[], int quantities[],int size);

Merchant(float rev);

Merchant(const Merchant& origClass);

~Merchant();

void haggle();

void printMenu();

void selectAntique(float& budget);

void leave(float budget);

Merchant operator= (Merchant& merc);

bool operator== (const Merchant& other);

void addAntique(Antique ant, int quantity);

private:

Antique \*antAnt;

int \*quantities;

float revenue;

int numAnt;

};

#endif

#include<iostream>

#include <iomanip>

#include <string>

#include <fstream>

#include <sstream>

#include "antique.h"

#include "merchant.h"

#include "MerchantGuild.h"

using namespace std;

int main()

{

int stockArr[10];

Antique antArr[10];

string inStock;

string inName;

float budget;

int playerOption;

string filename;

string line;

string inPrice;

string getName, getPrice;

int count = 0;

int iStock;

float iPrice;

int hagg = 0;

int size = 10;

cin >> filename;

ofstream outFS;

outFS.open("log2.txt", ios::trunc);

outFS.close();

ifstream myFileStream(filename);

if (!myFileStream.is\_open()) {

Antique a1;

Antique a2;

a2.setName("Painting");

Antique a3;

cout << "KEY: True is: 1 False is: 0" << endl << endl;

cout << "antique test" << endl;

cout << "fork and knife: $5.75" << endl;

cout << "0 : Ans=0" << endl;

cout << "1 : Ans=1" << endl;

cout << "merchant test" << endl;

cout << "0 : Ans=0" << endl;

cout << "1 : Ans=1" << endl;

cout << "1 : Ans=1" << endl;

cout << "merchant guild tests" << endl;

cout << "1 : Ans=1" << endl;

cout << "1 : Ans=1" << endl;

return 1;

}

else {

while (getline(myFileStream, line)) {

stringstream ss(line);

getline(ss, inName, ',');

getline(ss, inPrice, ',');

iPrice = stof(inPrice);

getline(ss, inStock, ',');

iStock = stoi(inStock);

stockArr[count] = iStock;

antArr[count].mutators(inName, iPrice);

antArr[count].setPrice(iPrice);

count++;

}

}

myFileStream.close();

cout << "Enter in budget: $";

cin >> budget;

Merchant seller(antArr, stockArr, size);

do {

cout << "Make a selection: " << endl;

cout << "1 - Haggle" << endl;

cout << "2 - View menu" << endl;

cout << "3 - Select an antique" << endl;

cout << "4 - Leave" << endl<<endl;

cin >> playerOption;

if (playerOption == 1) {

if (hagg == 0) {

seller.haggle();

cout << "You have successfully haggled and everything is 10% off."<<endl<<endl;

hagg++;

continue;

}

else {

cout << "Sorry, you have already haggled."<<endl<<endl;

continue;

}

}

else if (playerOption == 2) {

seller.printMenu();

cout << endl;

continue;

}

else if (playerOption == 3) {

seller.selectAntique(budget);

continue;

}

else if (playerOption == 4) {

seller.leave(budget);

}

else {

cout << "Invalid selection."<<endl<<endl;

}

} while (playerOption != 4);

cout << endl;

}

#ifndef ANTIQUE\_H

#define ANTIQUE\_H

#include<iostream>

#include <iomanip>

#include <string>

#include <fstream>

#include <sstream>

using namespace std;

class Antique

{

public:

Antique();

void mutators(string setName, float setPrice);

void accessors(string getName, float getPrice);

void setName(string setName);

string getName();

void setPrice(float setPrice);

float getPrice();

string toString();

Antique operator+(Antique ant);

bool operator== (Antique& other);

private:

string name;

float price;

};

#endif

#include <iostream>

#include <iomanip>

#include <string>

#include <fstream>

#include <sstream>

#include "antique.h"

#include "merchant.h"

using namespace std;

Antique::Antique() {

name = "";

price = 0;

}

void Antique::mutators(string setName, float setPrice) {

name = setName;

price = setPrice;

}

void Antique::accessors(string getName, float getPrice) {

name = getName;

price = getPrice;

}

string Antique::toString() {

ostringstream inSS;

string outStr;

inSS << name << ": $" << fixed << setprecision(2) << price;

outStr = inSS.str();

return outStr;

}

void Antique::setName(string setName) {

name = setName;

}

string Antique::getName() {

return name;

}

void Antique::setPrice(float setPrice) {

price = setPrice;

}

float Antique::getPrice() {

return price;

}

Antique Antique::operator+(Antique ant) {

Antique ant2;

ant2.price = price + ant.price;

ant2.name = name +" and "+ ant.name;

return ant2;

}

bool Antique::operator==(Antique& ant) {

return (ant.getPrice() == price) &&

(ant.getName() == name);

}