#include <iostream>

#include <fstream>

#include <iomanip>

#include <string>

using namespace std;

struct Item {

string name;

string category;

int quantity;

double price;

};

struct Transaction {

string type;

string itemName;

int quantity;

double unitPrice;

};

int loadInventory(Item items[]);

void saveInventory(Item items[], int count);

void saveTransaction(Transaction t);

void displayInventory(Item items[], int count);

void addNewItem(Item items[], int& count);

void buyItem(Item items[], int count);

void returnItem(Item items[], int count);

void showProfitReport();

int findItem(Item items[], int count, string name);

int main() {

Item items[100];

int itemCount = loadInventory(items);

int choice;

cout << "\*\*\*\*\*\*\*\* Welcome to Store Inventory & Profit Tracker \*\*\*\*\*\*\*\*" << endl;

do {

cout << "\nPlease choose one of the following operations:\n";

cout << "1. Add New Item to Inventory\n";

cout << "2. Display Inventory\n";

cout << "3. Buy Item\n";

cout << "4. Return Item\n";

cout << "5. View Profit Report\n";

cout << "6. Exit\n→ ";

cin >> choice;

cin.ignore();

switch (choice) {

case 1:

addNewItem(items, itemCount);

break;

case 2:

displayInventory(items, itemCount);

break;

case 3:

buyItem(items, itemCount);

break;

case 4:

returnItem(items, itemCount);

break;

case 5:

showProfitReport();

break;

case 6:

saveInventory(items, itemCount);

cout << "Inventory and transactions saved. Goodbye!" << endl;

break;

default:

cout << "Invalid option. Try again.\n";

}

} while (choice != 6);

return 0;

}

int loadInventory(Item items[]) {

ifstream file("inventory.txt");

int count = 0;

string line;

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

cout << "(No existing inventory found — starting new.)\n";

return 0;

}

getline(file, line);

while (getline(file, line)) {

if (line.empty()) continue;

size\_t pos1 = line.find(',');

size\_t pos2 = line.find(',', pos1 + 1);

size\_t pos3 = line.find(',', pos2 + 1);

items[count].name = line.substr(0, pos1);

items[count].category = line.substr(pos1 + 1, pos2 - pos1 - 1);

items[count].quantity = stoi(line.substr(pos2 + 1, pos3 - pos2 - 1));

items[count].price = stod(line.substr(pos3 + 1));

count++;

}

file.close();

return count;

}

void saveInventory(Item items[], int count) {

ofstream file("inventory.txt");

file << "Name,Category,Quantity,Price\n";

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

file << items[i].name << "," << items[i].category << ","

<< items[i].quantity << "," << fixed << setprecision(2)

<< items[i].price << "\n";

}

file.close();

}

void saveTransaction(Transaction t) {

ofstream file("transactions.txt", ios::app);

if (file.tellp() == 0)

file << "Type,ItemName,Quantity,UnitPrice\n";

file << t.type << "," << t.itemName << ","

<< t.quantity << "," << fixed << setprecision(2)

<< t.unitPrice << "\n";

file.close();

}

void displayInventory(Item items[], int count) {

cout << "\n============== INVENTORY ==============\n";

cout << left << setw(15) << "Item Name" << setw(15) << "Category"

<< setw(8) << "Qty" << setw(8) << "Price" << endl;

cout << "---------------------------------------------\n";

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

cout << left << setw(15) << items[i].name

<< setw(15) << items[i].category

<< setw(8) << items[i].quantity

<< "$" << fixed << setprecision(2) << items[i].price << endl;

}

}

void addNewItem(Item items[], int& count) {

if (count >= 100) {

cout << "Inventory full!\n";

return;

}

cout << "Enter item name: ";

getline(cin, items[count].name);

cout << "Enter category: ";

getline(cin, items[count].category);

cout << "Enter quantity: ";

cin >> items[count].quantity;

cout << "Enter price: ";

cin >> items[count].price;

cin.ignore();

count++;

cout << "Item added!\n";

}

int findItem(Item items[], int count, string name) {

for (int i = 0; i < count; i++)

if (items[i].name == name)

return i;

return -1;

}

void buyItem(Item items[], int count) {

string name;

int qty;

cout << "Enter item to buy: ";

getline(cin, name);

cout << "Quantity: ";

cin >> qty;

cin.ignore();

int idx = findItem(items, count, name);

if (idx == -1) {

cout << "Item not found!\n";

return;

}

if (items[idx].quantity < qty) {

cout << "Not enough stock!\n";

return;

}

items[idx].quantity -= qty;

double total = qty \* items[idx].price;

cout << "Successfully purchased " << qty << " x " << name

<< " for $" << fixed << setprecision(2) << total << endl;

Transaction t = { "BUY", name, qty, items[idx].price };

saveTransaction(t);

}

void returnItem(Item items[], int count) {

string name;

int qty;

cout << "Enter item to return: ";

getline(cin, name);

cout << "Quantity: ";

cin >> qty;

cin.ignore();

int idx = findItem(items, count, name);

if (idx == -1) {

cout << "Item not found!\n";

return;

}

items[idx].quantity += qty;

cout << qty << " x " << name << " returned and added back to inventory.\n";

Transaction t = { "RETURN", name, qty, items[idx].price };

saveTransaction(t);

}

void showProfitReport() {

ifstream file("transactions.txt");

string line;

double revenue = 0, refunds = 0;

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

cout << "No transaction records found.\n";

return;

}

getline(file, line);

while (getline(file, line)) {

if (line.empty()) continue;

size\_t pos1 = line.find(',');

size\_t pos2 = line.find(',', pos1 + 1);

size\_t pos3 = line.find(',', pos2 + 1);

string type = line.substr(0, pos1);

int qty = stoi(line.substr(pos2 + 1, pos3 - pos2 - 1));

double price = stod(line.substr(pos3 + 1));

double amount = qty \* price;

if (type == "BUY") revenue += amount;

else if (type == "RETURN") refunds += amount;

}

file.close();

cout << "\n========== PROFIT REPORT ==========\n";

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

cout << "Total Refunds: $" << refunds << endl;

cout << "Net Profit: $" << revenue - refunds << endl;

cout << "==================================\n";

}