```cpp
/*
    Zach Hofmeister     3/1/19
    Assignment 2: Object Oriented Programming - Inventory management system
    Description: This program keeps track of a store's inventory
*/

#include "pch.h"
#include <iostream>
#include <iomanip>
#include "Item.h"

using namespace std;

void displayItems(Item[4]); //Displays a list / menu of the items available in the store. Inventory is passed in to display.
void performTransaction(Item[4]); //Allows the user to purchase items from the inventory. Recursive. Inventory is passed in to access.

int main() {
    cout << fixed << showpoint << setprecision(2);
    cout << "===Welcome to the inventory helper===" << endl;
    cout << "Store hours will begin shortly." << endl;
    cout << "Please update your inventory..." << endl;
    cout << endl;

    Item items[4]; //Inventory array
    for (int i = 0; i < 4; i++) { //For loop for initializing the inventory of items.
        string name;
        int id, amount;
        double price;

        cout << "Enter item " << i + 1 << " to add to inventory." << endl;
        cout << "Please enter the product name: ";
        cin >> name;
        cout << "Enter the manufacturer's ID: ";
        cin >> id;
        cout << "Enter the retail value: ";
        cin >> price;
        cout << "Enter the quantity available: ";
        cin >> amount;
        cout << endl;

        items[i] = Item(name, id, price, amount); //Uses overloaded constructor to set the private values for each item.
    }

    cout << endl;
    displayItems(items); //Displays the inventory just created.
    cout << "Business hours are now open!" << endl;
    performTransaction(items); //Recursive function for transactions.

    cout << endl << "Closing shop - inventory left:" << endl;
    displayItems(items); //Displays the final inventory.

    return 0;
}

void displayItems(Item items[4]) { //Displays a list / menu of the items available in the store. Inventory is passed in to display.
    for (int i = 0; i < 4; i++) {
        cout << i+1 << ". " << items[i].getAmount() << " " << items[i].getName() << " left in stock at $" << items[i].getPrice() << " item id " << items[i].getID() << endl;
    }
}

void performTransaction(Item items[4]) { //Allows the user to purchase items from the inventory. Recursive. Inventory is passed in to access.
    static int runs = 0; //keeps track of runs to see if a purchase has already been made once.
    char input;
    cout << endl;
    cout << "Would you like to perform " << (runs < 1? "a" : "another") << " transaction? (y/n): " << endl;
    cin >> input;

    if (input == 'y') { //Make a purchase
        int itemSelection = 0, amount = 0;
        cout << "===Menu===" << endl;
        displayItems(items);
        do {
            cout << "Enter an item which you would like to purchase: ";
            cin >> itemSelection;
            if (itemSelection < 1 || itemSelection > 4) { //Input validation.
                cout << "Invalid item selection. Please enter 1-4." << endl;
            }
        } while (itemSelection < 1 || itemSelection > 4); //Input validation.

        do {
            cout << "How many: ";
            cin >> amount;
            if (amount < 1 || amount > items[itemSelection - 1].getAmount()) { //Input validation.
                cout << "Invalid amount. Current stock total is " << items[itemSelection - 1].getAmount() << "." << endl;
            }
        } while (amount < 1 || amount > items[itemSelection-1].getAmount()); //Input validation.
        cout << "SOLD " << amount << " " << items[itemSelection-1].getName() << " for $" << items[itemSelection-1].getPrice() * amount << endl;
        items[itemSelection-1].setAmount(items[itemSelection-1].getAmount() - amount); //Subtracts purchased items from the inventory.
        runs++;
        performTransaction(items); //Recursion
    } else if (input != 'y' && input != 'n') { //Input validation.
        cout << "Invalid input. Please try again." << endl;
        performTransaction(items); //Recursion
    }
}

/*
SAMPLE OUTPUT
===Welcome to the inventory helper===
Store hours will begin shortly.
Please update your inventory...

Enter item 1 to add to inventory.
Please enter the product name: Milk
Enter the manufacturer's ID: 1
Enter the retail value: 4.45
Enter the quantity available: 10

Enter item 2 to add to inventory.
Please enter the product name: Cookies
Enter the manufacturer's ID: 2
Enter the retail value: 1.00
Enter the quantity available: 40

Enter item 3 to add to inventory.
Please enter the product name: Roses
Enter the manufacturer's ID: 3
Enter the retail value: 2.00
Enter the quantity available: 12

Enter item 4 to add to inventory.
Please enter the product name: Carrots
Enter the manufacturer's ID: 4
Enter the retail value: .59
Enter the quantity available: 33


1. 10 Milk left in stock at $4.45 item id 1
2. 40 Cookies left in stock at $1.00 item id 2
3. 12 Roses left in stock at $2.00 item id 3
4. 33 Carrots left in stock at $0.59 item id 4
Business hours are now open!

Would you like to perform a transaction? (y/n):
y
===Menu===
1. 10 Milk left in stock at $4.45 item id 1
2. 40 Cookies left in stock at $1.00 item id 2
3. 12 Roses left in stock at $2.00 item id 3
4. 33 Carrots left in stock at $0.59 item id 4
Enter an item which you would like to purchase: 2
```

```
How many: 50
Invalid amount. Current stock total is 40.
How many: 25
SOLD 25 Cookies for $25.00

Would you like to perform another transaction? (y/n):
y
===Menu===
1. 10 Milk left in stock at $4.45 item id 1
2. 15 Cookies left in stock at $1.00 item id 2
3. 12 Roses left in stock at $2.00 item id 3
4. 33 Carrots left in stock at $0.59 item id 4
Enter an item which you would like to purchase: 12
Invalid item selection. Please enter 1-4.
Enter an item which you would like to purchase: 3
How many: 12
SOLD 12 Roses for $24.00

Would you like to perform another transaction? (y/n):
y
===Menu===
1. 10 Milk left in stock at $4.45 item id 1
2. 15 Cookies left in stock at $1.00 item id 2
3. 0 Roses left in stock at $2.00 item id 3
4. 33 Carrots left in stock at $0.59 item id 4
Enter an item which you would like to purchase: 4
How many: 23
SOLD 23 Carrots for $13.57

Would you like to perform another transaction? (y/n):
n

Closing shop - inventory left:
1. 10 Milk left in stock at $4.45 item id 1
2. 15 Cookies left in stock at $1.00 item id 2
3. 0 Roses left in stock at $2.00 item id 3
4. 10 Carrots left in stock at $0.59 item id 4

Press any key to close this window . . .
*/
```

```cpp
#pragma once
#ifndef ITEM_H
#define ITEM_H

#include <string>

using namespace std;

class Item { //Represents an item in the store.
    private:
        //Private data members
        string name;
        int id, amount;
        double price;
    public:
        //Constructors
        Item(); //Default constructor so that it is easier to initialize an array of Items without setting values.
        Item(string, int, double, int); //Overloaded constructor sets all values for Items.
        //Getters
        string getName();
        int getID();
        double getPrice();
        int getAmount();
        //Setters
        void setAmount(int a);
};

Item::Item() { //Default constructor

}

Item::Item(string n, int i, double p, int a) { //Constructor
    name = n;
    id = i;
    price = p;
    amount = a;
}

string Item::getName() {
    return name;
}

int Item::getID() {
    return id;
}

double Item::getPrice() {
    return price;
}

```

```
int Item::getAmount() {
    return amount;
}

void Item::setAmount(int a) {
    amount = a;
}

#endif

```