```
1 /*
 2
        Zach Hofmeister
                             3/1/19
 3
        Assignment 2: Object Oriented Programming - Inventory management system
 4
        Description: This program keeps track of a store's inventory
 5
   */
 6
7 #include "pch.h"
8 #include <iostream>
9 #include <iomanip>
10 #include "Item.h"
11
12 using namespace std;
13
14 void displayItems(Item[4]); //Displays a list / menu of the items available in
      the store. Inventory is passed in to display.
15 void performTransaction(Item[4]); //Allows the user to purchase items from the
      inventory. Recursive. Inventory is passed in to access.
16
17
   int main() {
18
        cout << fixed << showpoint << setprecision(2);</pre>
        cout << "===Welcome to the inventory helper===" << endl;</pre>
19
        cout << "Store hours will begin shortly." << endl;</pre>
20
        cout << "Please update your inventory..." << endl;</pre>
21
22
        cout << endl;</pre>
23
24
        Item items[4]; //Inventory array
25
        for (int i = 0; i < 4; i++) { //For loop for initializing the inventory of
          items.
            string name;
26
            int id, amount;
27
28
            double price;
29
            cout << "Enter item " << i + 1 << " to add to inventory." << endl;</pre>
30
31
            cout << "Please enter the product name: ";</pre>
32
            cin >> name;
33
            cout << "Enter the manufacturer's ID: ";</pre>
34
            cin >> id;
35
            cout << "Enter the retail value: ";</pre>
36
            cin >> price;
            cout << "Enter the quantity available: ";</pre>
37
38
            cin >> amount;
39
            cout << endl;</pre>
40
41
            items[i] = Item(name, id, price, amount); //Uses overloaded constructor
              to set the private values for each item.
42
        }
43
44
        cout << endl;</pre>
45
        displayItems(items); //Displays the inventory just created.
46
        cout << "Business hours are now open!" << endl;</pre>
47
        performTransaction(items); //Recursive function for transactions.
48
```

```
...CS121 Assignment2\CS121 Assignment2\CS121 Assignment2.cpp
```

```
2
```

```
cout << endl << "Closing shop - inventory left:" << endl;</pre>
50
        displayItems(items); //Displays the final inventory.
51
52
        return 0;
53 }
54
55 void displayItems(Item items[4]) { //Displays a list / menu of the items
      available in the store. Inventory is passed in to display.
56
        for (int i = 0; i < 4; i++) {
57
            cout << i+1 << ". " << items[i].getAmount() << " " << items[i].getName() >
              << " left in stock at $" << items[i].getPrice() << " item id " << items >
              [i].getID() << endl;</pre>
58
        }
59 }
60
61 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 →
62
          been made once.
63
        char input;
64
        cout << endl;</pre>
        cout << "Would you like to perform " << (runs < 1? "a" : "another") << "</pre>
65
          transaction? (y/n): " << endl;</pre>
66
        cin >> input;
67
68
        if (input == 'y') { //Make a purchase
69
            int itemSelection = 0, amount = 0;
            cout << "===Menu===" << endl;</pre>
70
71
            displayItems(items);
72
            do {
                cout << "Enter an item which you would like to purchase: ";</pre>
73
74
                cin >> itemSelection;
                if (itemSelection < 1 || itemSelection > 4) { //Input validation.
75
76
                    cout << "Invalid item selection. Please enter 1-4." << endl;</pre>
77
            } while (itemSelection < 1 | itemSelection > 4); //Input validation.
78
79
80
            do {
                cout << "How many: ";</pre>
81
                cin >> amount;
82
                if (amount < 1 || amount > items[itemSelection - 1].getAmount()) { // >
83
                  Input validation.
84
                    cout << "Invalid amount. Current stock total is " << items</pre>
                       [itemSelection - 1].getAmount() << "." << endl;</pre>
85
86
            } while (amount < 1 || amount > items[itemSelection-1].getAmount()); //
              Input validation.
            cout << "SOLD " << amount << " " << items[itemSelection-1].getName() << " >
87
               for $" << items[itemSelection-1].getPrice() * amount << endl;</pre>
            items[itemSelection-1].setAmount(items[itemSelection-1].getAmount() -
88
              amount); //Subtracts purchased items from the inventory.
89
            runs++;
```

```
...CS121 Assignment2\CS121 Assignment2\CS121 Assignment2.cpp
            performTransaction(items); //Recursion
 91
        } else if (input != 'y' && input != 'n') { //Input validation.
            cout << "Invalid input. Please try again." << endl;</pre>
 92
 93
            performTransaction(items); //Recursion
 94
        }
 95 }
 96
 97 /*
 98 SAMPLE OUTPUT
99 ===Welcome to the inventory helper===
100 Store hours will begin shortly.
101 Please update your inventory...
102
103 Enter item 1 to add to inventory.
104 Please enter the product name: Milk
105 Enter the manufacturer's ID: 1
106 Enter the retail value: 4.45
107 Enter the quantity available: 10
108
109 Enter item 2 to add to inventory.
110 Please enter the product name: Cookies
111 Enter the manufacturer's ID: 2
112 Enter the retail value: 1.00
113 Enter the quantity available: 40
114
115 Enter item 3 to add to inventory.
116 Please enter the product name: Roses
117 Enter the manufacturer's ID: 3
118 Enter the retail value: 2.00
119 Enter the quantity available: 12
120
121 Enter item 4 to add to inventory.
122 Please enter the product name: Carrots
123 Enter the manufacturer's ID: 4
124 Enter the retail value: .59
125 Enter the quantity available: 33
126
127
128 1. 10 Milk left in stock at $4.45 item id 1
129 2. 40 Cookies left in stock at $1.00 item id 2
130 3. 12 Roses left in stock at $2.00 item id 3
131 4. 33 Carrots left in stock at $0.59 item id 4
132 Business hours are now open!
133
134 Would you like to perform a transaction? (y/n):
135 y
136 ===Menu===
137 1. 10 Milk left in stock at $4.45 item id 1
138 2. 40 Cookies left in stock at $1.00 item id 2
139 3. 12 Roses left in stock at $2.00 item id 3
```

140 4. 33 Carrots left in stock at \$0.59 item id 4
141 Enter an item which you would like to purchase: 2

3

```
142 How many: 50
143 Invalid amount. Current stock total is 40.
144 How many: 25
145 SOLD 25 Cookies for $25.00
146
147 Would you like to perform another transaction? (y/n):
148 y
149 ===Menu===
150 1. 10 Milk left in stock at $4.45 item id 1
151 2. 15 Cookies left in stock at $1.00 item id 2
152 3. 12 Roses left in stock at $2.00 item id 3
153 4. 33 Carrots left in stock at $0.59 item id 4
154 Enter an item which you would like to purchase: 12
155 Invalid item selection. Please enter 1-4.
156 Enter an item which you would like to purchase: 3
157 How many: 12
158 SOLD 12 Roses for $24.00
160 Would you like to perform another transaction? (y/n):
161 y
162 ===Menu===
163 1. 10 Milk left in stock at $4.45 item id 1
164 2. 15 Cookies left in stock at $1.00 item id 2
165 3. 0 Roses left in stock at $2.00 item id 3
166 4. 33 Carrots left in stock at $0.59 item id 4
167 Enter an item which you would like to purchase: 4
168 How many: 23
169 SOLD 23 Carrots for $13.57
171 Would you like to perform another transaction? (y/n):
172 n
173
174 Closing shop - inventory left:
175 1. 10 Milk left in stock at $4.45 item id 1
176 2. 15 Cookies left in stock at $1.00 item id 2
177 3. 0 Roses left in stock at $2.00 item id 3
178 4. 10 Carrots left in stock at $0.59 item id 4
180 Press any key to close this window . . .
181 */
```

```
1 #pragma once
 2 #ifndef ITEM_H
 3 #define ITEM_H
 5 #include <string>
 6
 7 using namespace std;
 9 class Item { //Represents an item in the store.
10
       private:
            //Private data members
11
12
            string name;
            int id, amount;
13
14
            double price;
15
       public:
16
            //Constructors
            Item(); //Default constructor so that it is easier to initialize an array ➤
17
              of Items without setting values.
            Item(string, int, double, int); //Overloaded constructor sets all values
18
              for Items.
19
            //Getters
            string getName();
20
21
            int getID();
22
            double getPrice();
23
            int getAmount();
24
            //Setters
25
            void setAmount(int a);
26 };
27
28 Item::Item() { //Default constructor
29
30 }
31
32 Item::Item(string n, int i, double p, int a) { //Constructor
33
       name = n;
34
        id = i;
35
       price = p;
36
        amount = a;
37 }
39 string Item::getName() {
40
       return name;
41 }
42
43 int Item::getID() {
44
       return id;
45 }
46
47 double Item::getPrice() {
48
       return price;
49 }
50
```

```
\dots cuments \verb|\GitHub| CS121 Assignment2| CS121 Assignment2| Item.h
```

2

```
51 int Item::getAmount() {
52    return amount;
53 }
54
55  void Item::setAmount(int a) {
66    amount = a;
57 }
58
59  #endif
60
```