Final Project- Pawn Shop

ELEC 3150 – Object Oriented Programming (Fall 2023)

Nick Cebula

Requirements:

- Your project topic could be anything that utilize knowledge in object-oriented programming. You project ***must*** include class, object, inheritance, member functions and variable
- In the project proposal, Please describe your plan on utilizing class and inheritance in your final project
- This project can be done in C++ or Python
- Your project's topic must get approval from instructor first, otherwise it wouldn't count as your final project
- Topic of your final project is first come first serve. Every group must have different project's topic.
- Difficulty level:
  - The difficulty level should be similar to question 2 of lab 4 or question 2 of lab 5

Assumptions/Description:

Methods: Customer(user) can come in and buy items or sell items. There will be a list of current items in the shop's inventory. You can print items in inventory and their value.

Base class: Shop

Inherited class: LuxuryShop, has same attributes and methods, but has additional attribute of "feature" which displays the special feature of luxury items.

Polymorphism: When doing buy or sell method it will refer to the method of either shop or luxury shop with prefix of "*$*" and ends with in parenthesis (feature), luxury item special feature.

LuxuryShop has tax of 20%, regular shop has no tax.

When selling items, you can choose to negotiate with the shop for a better price when selling an item. When selling a luxury item it is 100 subtracted from original price automatically, then random number generator.

```
--------Welcome to my Pawn Shop!--------
Please select one of the following options:
1 - Print inventory
2 - Buy an item
3 - Sell an item
4 - Exit
1
Inventory:
 Book- $ 10
 Clock- $ 30
 Radio- $ 50
 Chromebook- $ 150
 TV- $ 300
*$* GucciBelt- $ 1200(Leather)
*$* iPhone- $ 2000(15 Pro Max)
*$* DiamondRing- $ 3000(Flawless)
*$* Rolex- $ 15000(Diamond)

Please select one of the following options:
1 - Print inventory
2 - Buy an item
3 - Sell an item
4 - Exit
2
Enter item you want to buy (Luxury items have 20% tax): Rolex
You have bought Rolex Diamond for $ 18000
```

```
Inventory:
 Book- $ 10
 Clock- $ 30
 Radio- $ 50
 Chromebook- $ 150
 TV- $ 300
*$* GucciBelt- $ 1200(Leather)
*$* iPhone- $ 2000(15 Pro Max)
*$* DiamondRing- $ 3000(Flawless)

Please select one of the following options:
1 - Print inventory
2 - Buy an item
3 - Sell an item
4 - Exit
3
Enter item you are selling: Gold
Is it Luxury? $1000+ (1=No, 2=Yes)2
How much is it worth? 15000
We offer $ 11475 for your item
Do you accept this offer? (1=No, 2=Yes): 1
New offer: 2382
Do you accept this offer? (1=No, 2=Yes): 1
New offer: 10473
Do you accept this offer? (1=No, 2=Yes): 1
New offer: 13870
Do you accept this offer? (1=No, 2=Yes): 2
You have sold Gold for $ 13870
What is its feature?: 24k
```

```
Inventory:
 Book- $ 10
 Clock- $ 30
 Radio- $ 50
 Chromebook- $ 150
 TV- $ 300
*$* GucciBelt- $ 1200(Leather)
*$* iPhone- $ 2000(15 Pro Max)
*$* DiamondRing- $ 3000(Flawless)
*$* Gold- $ 13870(24k)

Please select one of the following options:
1 - Print inventory
2 - Buy an item
3 - Sell an item
4 - Exit
2
Enter item you want to buy (Luxury items have 20% tax): Gikd
Item not found. Try again
Enter item you want to buy (Luxury items have 20% tax): Gold
You have bought Gold 24k for $ 16644
Please select one of the following options:
1 - Print inventory
2 - Buy an item
3 - Sell an item
4 - Exit
1
Inventory:
```

```
Please select one of the following options:
1 - Print inventory
2 - Buy an item
3 - Sell an item
4 - Exit
4
Exiting program. Thanks for shopping :)
```

Main.cpp:

```cpp
#include <iostream>
#include <string>
#include <list>
#include "Store.h"
#include "LuxuryStore.h"
using std::cin;
using std::cout;
using std::string;
using std::endl;
using std::list;
using std::find;

int main() {
    cout << "--------Welcome to my Pawn Shop!--------" << endl;
    list<Store*> pawn;
    pawn.push_back(new Store(10, "Book"));
    pawn.push_back(new Store(30, "Clock"));
    pawn.push_back(new Store(50, "Radio"));
    pawn.push_back(new Store(150, "Chromebook"));
    pawn.push_back(new Store(300, "TV"));
    pawn.push_back(new LuxuryStore(1200, "GucciBelt", "Leather"));
    pawn.push_back(new LuxuryStore(2000, "iPhone", "15 Pro Max"));
    pawn.push_back(new LuxuryStore(3000, "DiamondRing", "Flawless"));
    pawn.push_back(new LuxuryStore(15000, "Rolex", "Diamond"));

    int choice = 0;
    cout << "Please select one of the following options: " << endl;
    cout << "1 - Print inventory" << endl;
    cout << "2 - Buy an item" << endl;
    cout << "3 - Sell an item" << endl;
    cout << "4 - Exit" << endl;
    cin >> choice;

    while (choice >= 1 && choice <= 4) {
        if (choice == 1) {
            cout << "Inventory:" << endl;
            for (auto& store : pawn) {
                (store)->print_list();
                cout << endl;
            }
            cout << endl;
        }
        else if (choice == 2) {
            string item;
            int found = 1;
            while (found != 0) {
                cout << "Enter item you want to buy (Luxury items have 20% tax): ";
                cin >> item;
                //std::getline(cin, item);
                auto buying = find_if(pawn.begin(), pawn.end(), [item](Store* store) {
                    return store->return_item() == item;
                    });

                if (buying != pawn.end()) {
                    (*buying)->buy_item(item);
                    pawn.erase(buying);
                    found = 0;
                }
                else {
                    cout << "Item not found. Try again" << endl;
                }
            }
        }
        else if (choice == 3) {
            string item1;
            string feature;
            int luxq = 0;
            int value = 0;
            cout << "Enter item you are selling: ";
            cin >> item1;
            cout << "Is it Luxury? $1000+ (1=No, 2=Yes)";
            cin >> luxq;
            Store* soldItem = new Store(50, item1);
            LuxuryStore* sellLux = new LuxuryStore(50, item1, feature);
            if (luxq == 1) {
                value = soldItem->sell_item();
                soldItem->set_cost(value);
                pawn.push_back(soldItem);
            }
            if (luxq == 2) {
                value = sellLux->sell_item();
                cout << "What is its feature?: ";
                cin >> feature;
                sellLux->set_cost(value);
                sellLux->set_feature(feature);
                pawn.push_back(sellLux);
            }
        }
```

```
        else if(choice == 4) {
            cout << "Exiting program. Thanks for shopping :)" << endl;
            break;
        }
        cout << "Please select one of the following options: " << endl;
        cout << "1 - Print inventory" << endl;
        cout << "2 - Buy an item" << endl;
        cout << "3 - Sell an item" << endl;
        cout << "4 - Exit" << endl;
        cin >> choice;
    }
    return 0;
}
```

Description of main.cpp: I started off the program by making a list on the stack. Then update conditions to ask the user which option they want to choose. Then it enters a while loop to keep asking the user. While their choice is between 1-4. Option 1 will print the inventory using auto variable specifier for list. While using method for either store or luxurystore which has different specifier for luxury items (as seen in results). Option 2 will receive input from user of what item they want to buy. This is a string, I used find_if function from STL list, it searches the list from top to bottom and returns the item (or last item if not found) and then a if statement compares it to the input word.if found it sets a flag found=0 which exits the loop. (Here it would be better if I had a exit the loop option in case user doesn't want to buy item anymore.) option 3 has input string of item being sold, asks the user to enter name of item they are selling, and if it is a luxury item it will go through the steps to push_back to list. In order to do this I had to set a temp item and then use a method to set the price. Choice 4 exits the program and leaves a nice message.

Store.h:

```
#pragma once
#include <string>
using std::string;
class Store
{
    //attributes
protected:
    string item;
    int cost;
public:
    //constructors
    Store();
    Store(int in_cost, string in_item);
    //methods
    virtual void print_list();
    string return_item();
    virtual void buy_item(string item);
    int sell_item();
    void set_cost(int newCost);
    //destructor
    ~Store();
};
```

Store.cpp:

```
#include "Store.h"
#include <iostream>
#include <cstdlib>
#include <ctime>
using std::cout;
using std::cin;
using std::endl;
Store::Store(int in_cost, string in_item)
{
    cost = in_cost;
    item = in_item;
}

void Store::print_list()
{
    cout << " " << item << "- $ " << cost;
}

string Store::return_item()
{
    return item;
}

void Store::buy_item(string item)
{
    //for regular item
    int total_cost;
    total_cost = cost;  //no proft for regular
    cout << "You have bought " << item << " for $ " << total_cost << endl;
}
```

```cpp
int Store::sell_item()
{
    std::srand(std::time(0));
    int bargain = 0;
    int est_val = 0;
    cout << "How much is it worth? ";
    cin >> est_val;
    int value = est_val - (std::rand() % 100); //sub max $10 to price
    cout << "We offer $ " << value << " for your item" << endl;
    int user = 0;
    int value1 = 0;
    while (user!=2) {
        cout << "Do you accept this offer? (1=No, 2=Yes): ";
        cin >> user;
        if (user == 1) {
            bargain = 1;    //flag for bargain occured
            int range = est_val;
            value1 = est_val - (rand() % range); //sub max $10 to price
            cout << "New offer: " << value1 << endl;
        }
    }
    if (bargain == 1) {
        cout << "You have sold " << item << " for $ " << value1 << endl;
        return value1;
    }
    else {
        cout << "You have sold " << item << " for $ " << value << endl;
        return value;
    }

}
```

```cpp
void Store::set_cost(int newCost) {
    cost = newCost;
}
Store::~Store()
{
}
```

Store Class description: I have print list as virtual so it can be overridden by luxurystore class when required. Same as buy item. These are simple methods they set class cost to cost set in list. Sell item uses random number generator to negotiate with the customer on a price, they can keep trying until they agree. Set cost method is needed to set cost for an item in a list since a customer is selling a item it needs to add a new item to the list and store it.

LuxuryStore class:

```cpp
#pragma once
#include "Store.h"
class LuxuryStore :
    public Store
{
    string feature;
public:
    LuxuryStore(int in_cost, string in_item, string in_feature);
    void print_list();
    void buy_item(string item);
    int sell_item();
    void set_cost(int newCost);
    void set_feature(string new_feature);
};
```

```cpp
#include "LuxuryStore.h"
#include <iostream>
using std::cout;
using std::cin;
using std::endl;

LuxuryStore::LuxuryStore(int in_cost, string in_item, string in_feature)
{
    cost = in_cost;
    item = in_item;
    feature = in_feature;
}

void LuxuryStore::print_list()
{
    cout << "*$*";
    Store::print_list();
    cout << "(" << feature << ")";

}

void LuxuryStore::set_feature(string new_feature) {
    feature = new_feature;
}

void LuxuryStore::set_cost(int newCost)
{
    cost = newCost;
}
```

```cpp
void LuxuryStore::buy_item(string item)
{
    //for regular item
    int total_cost;
    total_cost = cost*1.2;  //shop makes 20% profit off luxury item
    cout << "You have bought " << item << " " << feature << " for $ " << total_cost  << endl;
}

int LuxuryStore::sell_item()
{
    std::srand(std::time(0));
    int bargain = 0;
    int est_val = 0;
    int luxq = 0;
    cout << "How much is it worth? ";
    cin >> est_val;
    int value = est_val - ((std::rand() % est_val)+100);
    cout << "We offer $ " << value << " for your item" << endl;
    int user = 0;
    int value1 = 0;
    while (user != 2) {
        cout << "Do you accept this offer? (1=No, 2=Yes): ";
        cin >> user;
        if (user == 1) {
            bargain = 1;    //flag for bargain occured
            value1 = est_val - ((std::rand() % est_val)+100); //random number +100 is sub from value
            cout << "New offer: " << value1 << endl;
        }
    }
    if (bargain == 1) {
        cout << "You have sold " << item << " for $ " << value1 << endl;
        return value1;
    }
    else {
        cout << "You have sold " << item << " for $ " << value << endl;
        return value;
    }

}
```

```cpp
LuxuryStore::~LuxuryStore() {

}
```

LuxuryStore description: LuxuryStore is inherited from Store, it has a separate attribute of feature which os a special quality about the luxury item since it is better than regular item. It has polymorphism for printing the inventory. 20% Tax for buying items in luxury store, and when you sell a luxury item it subtract 100 automatically as the minimum selling price, uses customer estimated price as a modulus.

Discussion:

- Did you achieve your goal?
- Problems that you face during this project and what you did to overcome it
- Suggestion/Future work (If any)

I achieved all the goals I set for this final project, impressing myself by using the list STL properly and making the negotiation feature and implementing the polymorphism of adding luxury items with prefix and post fix of (feature).

Using the list on the stack was very challenging and I had to do additional research on how to call items on the stack from a list. However, it was very similar, and I would use the arrow to do so. Another challenge was selling items, I had to make a temporary item and then fill it in with the user input variables then push back to list.