بسم الله الرحمن الرحیم

گزارش پروژه : مدیریت فروشگاه

درس : برنامه سازی پیشرفته

برنامه نویس : رضا دهقانی

استاد مربوطه : جناب آقای دکتر اسماعیلی

تابستان ۱۳۹۹

شرح پروژه:

پروژه مدیریت فروشگاه به جهت سهولت مدیریت فروش سوپر مارکت و هایپر مارکت ها و با هدف جمع آوری اطلاعات فروش و چاپ فاکتور و ثبت اطلاعات مشتریان با رویکرد شی گرا طراحی گردید.

زبان برنامه نویسی مورد استفاده در این پروژه سی پلاس پلاس است و با نرم افزار ویژوال استودیو ورژن ۲۰۱۵ نوشته شده است.

ذکر این نکته خالی از لطف نیست که این پروژه قابلیت این را دارد که در تمامی فروشگاه هایی که محصولی برای فروش دارند مانند نمایشگاه خودرو و ... استفاده شود.

از دیگر قابلیت های این نرم افزار می توان به ذخیره شناسه محصول (Qrcode) تا یازده رقم با فرمت int، عدم قبول مقادیر منفی و صفر، ذخیره قیمت و تعداد محصول تا 18,446,744,073,709,551,615 که عدد بسیار بالایی است و توانایی پشتیبانی قیمت ها و تعداد بسیار بالا را دارا می باشد.

یکی از قابلیت های منحصر به فرد دیگری که این نرم افزار دارا می باشد صحت سنجی تلفن همراه مشتری هنگام وارد کردن اطلاعات مشتری می باشد به طوری که حتما شماره تلفن وارد شده باید ۱۱ رقم داشته باشد و با ۰ آغاز شود. سعی شده است تا این صحت سنجی ها در سرتاسر نرم افزار گنجانده شود تا از ورود

و ذخیره اطلاعات نادرست و غیرصحیح به نرم افزار تا حد امکان جلوگیری شود.

همانند شناسه محصول برای شماره تلفن های مشتری هم این قابلیت اضافه شده است که شماره تلفن همراه مشتری تکراری نباشد و در صورت تکراری بودن این مورد با یک پیغام به کاربر نمایش داده شود.

همچنین سعی شده است تا نمایش اطلاعات در این برنامه با نظم و ترتیب و با رعایت فاصله بین عناصر در صفحه نمایشگر انجام شود.

یک نکته مهم که در طول نوشتن برنامه به آن برخوردم مشاهده ارور هایی بود که در طول اجرای برنامه خصوصا در زمان ارتباط کلاس ها با هم به طور اتفاقی به آن برمی خوردم؛ رفع این ارور و تلاش برای اینکه نرم افزار تجربه کاربری مطلوبی داشته باشد تا حد امکان پایدار باشد و در حین اجرا دچار کرش نشود؛ موجب شد تا تک تک این ارور ها و دلیل رخ دادن آن ها را بررسی و سپس به رفع آن ها بپردازم که بعضا برای برخی از آن ها با توجه به اینکه هیچ پیش زمینه ای نداشتم با آزمون و خطا جلو می رفتم تا زمانی که ارور به طور کامل رفع می شد.

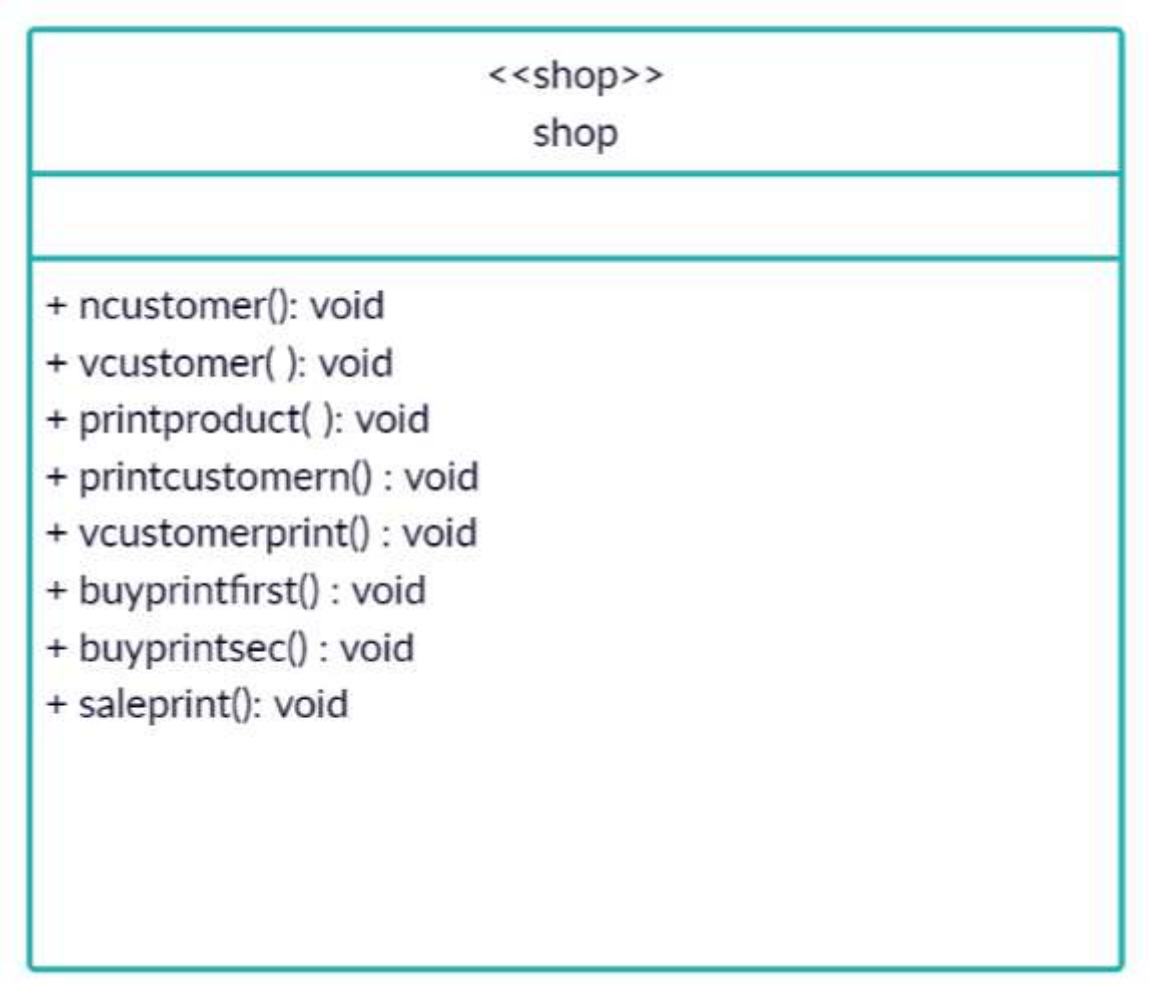
پیاده سازی این ویژگی ها بعضا دارای پیچیدگی ها و ترفند هایی بود که پیاده سازی آن ها برای اولین بار زمان زیادی می گرفت که در شرح جزئیات به این ترفند ها پرداخته می شود.

جزئیات پروژه:

در این پروژه از ۶ کلاس فروشگاه، محصولات، مشتری ، خرید، مشتری معمولی و وی آی پی تشکیل شده است که در ادامه جزئیات هر یک از کلاس ها و ارتباط بین آن ها شرح داده می شود.

کلاس فروشگاه (shop):

UML



در این کلاس از ۱۰ متود مختلف استفاده شده است که ۸ مورد آن در UML آمده است و ۲ مورد سازنده نیز برای این کلاس تعریف شده است.

کد کلاس:

فایل shop.h

```
#pragma once
#include <vector>
#include "product.h"
#include "person.h"
#include "buy.h"
vector <product> products;
vector <normal> normals;
vector <vip> vips;
class shop {
	public:
		shop();
		shop(vector<product> p){ products = p; }
		void ncustomer(vector <normal> n) { normals = n; }
		void vcustomer(vector <vip> v) { vips = v; }
		void printproduct();
		void printcustomern(vector <normal> n);
		void vcustomerprint(vector <vip> v);
		void buyprintfirst(vector<pair<buy, vector<product>>> b);
		void buyprintsec(vector<pair<buy, vector<product>>> b, int  bc);
		void saleprint(vector <vector <product>> sl);
};
```

فایل shop.cpp

```
#include "shop.h"

void shop::printproduct(){
	for (int i = 0; i < products.size(); i++)
	products[i].print();
}
void shop::printcustomern(vector <normal> n) {
	for (int i = 0; i < n.size(); i++)
		n[i].print();
}
void shop::vcustomerprint(vector <vip> v) {
	for (int i = 0; i < v.size(); i++)
		v[i].print();
}
void shop::buyprintfirst(vector<pair<buy, vector<product>>> b) {
	for (int i = 0; i < b.size(); i++)
	{
		b[i].first.printbuy();
	}
}
void shop::buyprintsec(vector<pair<buy, vector<product>>> b,int  bc) {
	for (int i = 0; i < b[bc].second.size(); i++)
		b[bc].second[i].print();
}
void shop::saleprint(vector <vector <product>> sl) {
	for (int i = 0; i < sl.size(); i++)
		for (int j = 0; j<sl[i].size(); j++)
			sl[i][j].print();
}

shop::shop() {}
```

شرح فایل shop.h:

`#pragma once` برای این استفاده شده است که کلاس یک مرتبه باز شود.

متود های اول و دوم هر دو سازنده کلاس فروشگاه هستند که اولی امکان تعریف آبجکت ها و دومی امکان تعریف شی از فروشگاه با استفاده از وکتور محصولات را فراهم می کند.

متود های ncustomer و vcustomer دو متود ستر setter هستند که وکتور مشتریان را گرفته و به عنوان شی در کلاس امکان ثبت شدن می دهند.

متود هایی که در زیر معرفی می شوند برای قسمت گزارشات پروژه و چاپ اطلاعات بکار می روند:

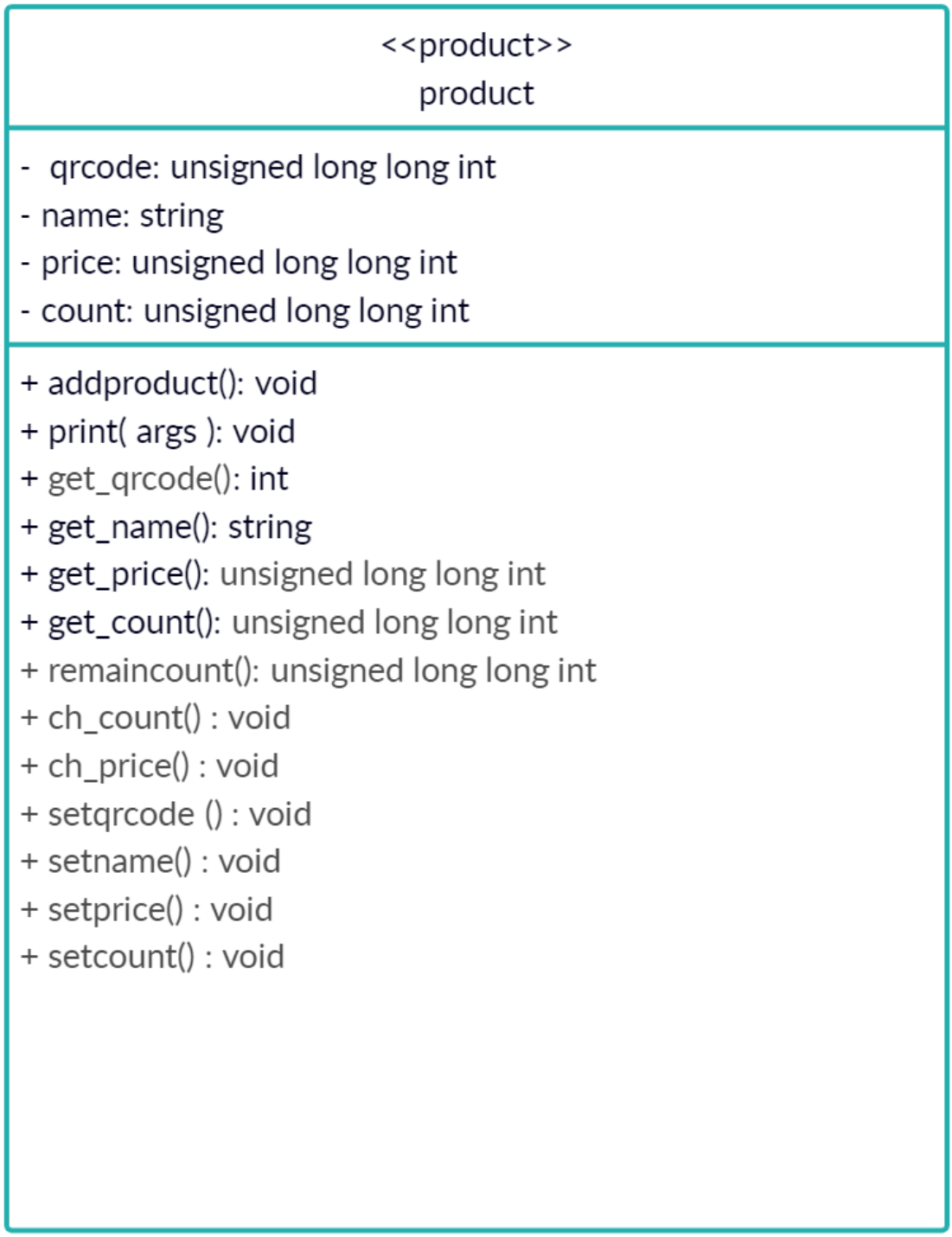
Printcustomern و vcustomerprint: برای نمایش اطلاعات مشتریان معمولی و وی آی پی تعریف شده است. این دو متد لیست مشتریان را در قالب وکتور در ورودی گرفته و اطلاعات مشتریان را شامل نام و نام خانوادگی و شماره تلفن همراه به ترتیب عضویت در فروشگاه چاپ می کنند.

Buyprintfirst و buyprintsec: برای چاپ اطلاعات خریدار و جزئیات خرید استفاده شده است. این دو متود یک جفت وکتور را به عنوان ورودی می گیرند و اطلاعات خرید را چاپ می کنند.

Saleprint: برای چاپ اطلاعات کالاهای فروخته شده بکار می رود. ورودی آن هم یک وکتور دو بعدی است که با حلقه تمام خانه های آن را چاپ می کند.

کلاس product:

UML



فایل product.h

```
#pragma once
#include <iostream>
#include <string>
using namespace std;
class product {
public:
	void addproduct(unsigned long long int qr, string n, unsigned long long int p,
unsigned long long int c);
	void print();
	int get_qrcode(){ return qrcode; }
	string get_name(){ return name; }
	unsigned long long int get_price() { return price; }
	unsigned long long int remaincount(unsigned long long int qrcode, unsigned long
long int count, int number);
	unsigned long long int get_count() { return count; }
	void ch_count(int c);
	void ch_price(int p);
	void setqrcode(int);
	void setname(string);
	void setprice(int);
	void setcount(int);
private:
	unsigned long long int qrcode;
	string name;
	unsigned long long int price;
	unsigned long long int count;
                                                                                    };
```

فایل product.cpp

```
#include "product.h"
#include <iostream>
#include <string>
#include <iomanip>
using namespace std;
void product::addproduct(unsigned long long int qr, string n, unsigned long long int
p, unsigned long long int c){
	unsigned long long int def = 1;
	qrcode = (qr<0 || qr==0) ? def : qr;
	name = n;
	price = (p<0 || qr==0) ? def : p;
	count = (c<0) ? def : c;
}
/*
product::product(long long qrcode, char name, long long price, long long count) {
	addproduct(qrcode, name,price, count);
}

ostream &operator<<(ostream& output, const product &p) {
*/
void product::print()
{
	cout << left << "QrCode: "<< setw(12) << qrcode << "Name: " << setw(30) << name
<< setw(7) << "Price: " << setw(12) << price << setw(7) << "Number: " << setw(12) <<
count
		<< endl;
}
void product::ch_count(int c) {
```

```
		count = c;
}
void product::ch_price(int p) {
	price = p;
}
unsigned long long int product::remaincount(unsigned long long int qrcode, unsigned
long long int count, int number)
{
	if (count >= number) {
		count = count - number;
		return count;
	}
	if (count < number) {
		cout <<"\nRemein count of product: " << count<<"\n";
	}
}
void product::setqrcode(int qr){
	qrcode = qr;
}
void product::setname(string n) {
	name = n;
}
void product::setprice(int p) {
	price = p;
}
void product::setcount(int c) {
	count = c;
}
```

جزئیات:

این کلاس از ۴ فیلد و ویژگی تشکیل شده است که برای اطلاعات کالا مورد استفاده قرار می گیرند.

شناسه محصول نام محصول قیمت و تعداد کالاهای موجود در فروشگاه.

متود های این کلاس:

متود addproduct: ورودی نداشته و برای اضافه کردن ویژگی های کالا به کلاس مورد استفاده قرار می گیرد.

متود print: برای پرینت اطلاعات کالا شامل شناسه محصول نام محصول قیمت و تعداد کالاهای موجود در فروشگاه. این متود از کتابخانه iomanip برای نظم دادن به چاپ اطلاعات کالا استفاده می کند.

متودهای getter شامل:

متود get_name ، get_qrcode ، get_price ، get_count که هر کدام فیلد های کلاس را به عنوان خروجی برگشت می دهند و این امکان را می دهند تا در خارج از کلاس به فیلد های private دسترسی داشت.

متود های setter شامل:

متود setname ، setqrcode ، setprice، setcount که مقادیری را به عنوان ورودی گرفته و تحت عنوان ویژگی های کلاس در داخل کلاس ست می کنند.

متود های بالا در قسمت های مختلف برنامه به خصوص قسمت هایی که با کالا درگیر بوده اند استفاده شده است.

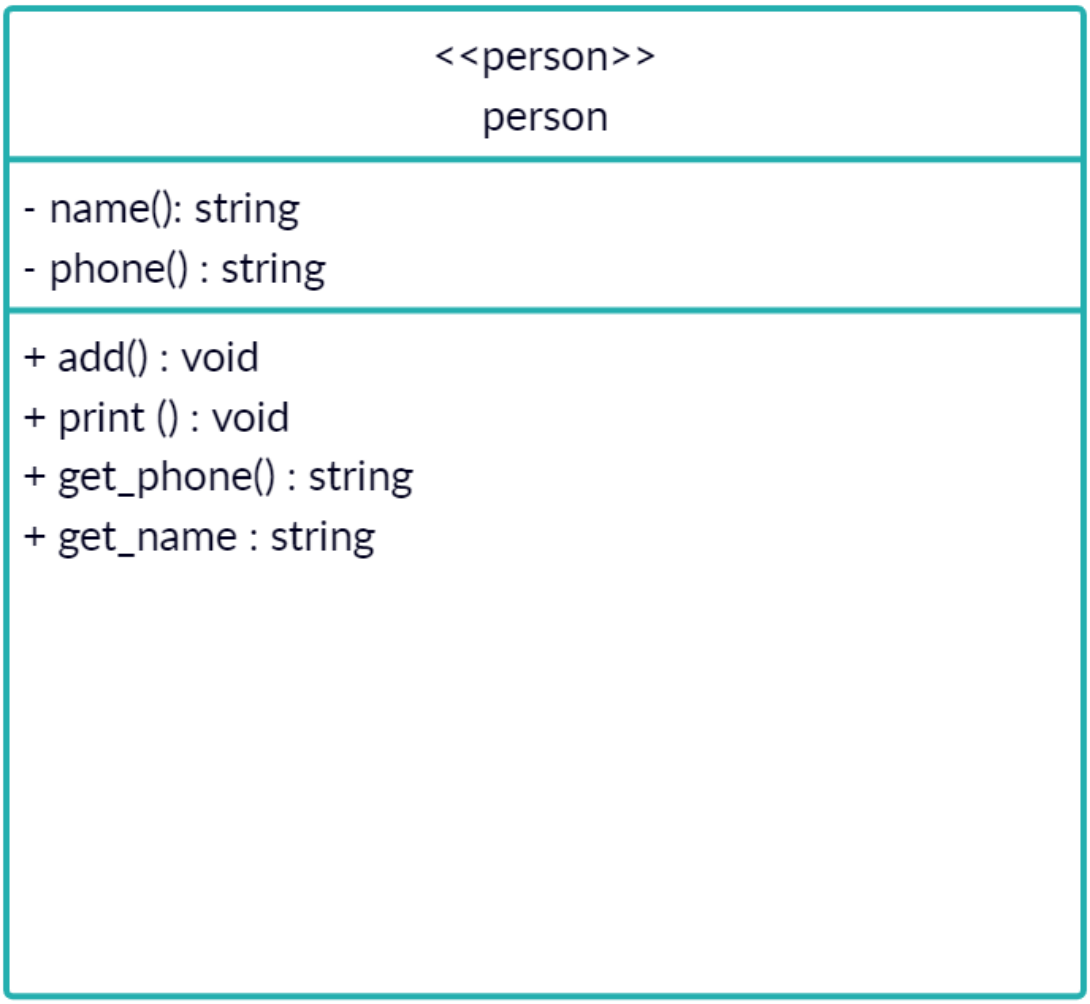
در متود ch_count و ch_price به ترتیب مقادیر جدید تعداد کالا و قیمت کالا را گرفته و آن را در فیلد های مربوطه در کلاس تغییر می دهند. از این متود در قسمت دوم منو فروشگاه برای تغییر قیمت و تعداد کالاهای فروشگاه استفاده شده است.

متود remaincount: این متود برای نمایش مقدار باقیمانده محصول در قسمت ششم یعنی قسمت خرید مورد استفاده قرار گرفته است، بطوریکه تعداد کالای درخواستی برای خرید را از کاربر گرفته و در صورتی که تعداد کالای درخواستی برای خرید از تعداد کالای موجود در فروشگاه بیشتر باشد تعداد باقیمانده را به کاربر نمایش می دهد، در صورتیکه تعداد کالا برای خرید کمتر از تعداد کالای موجود در فروشگاه باشد صرفا تعداد کالا را برمی گرداند.

کلاس مشتری (Person):

این کلاس یک کلاس پدر است که دو فرزند مشتری معمولی و مشتری وی آی پی دارد و در آن از وراثت کلاس ها استفاده شده است.

UML



فایل person.h:

```
#pragma once
#include <iostream>
#include <string>
using namespace std;

class person
{
public:
```

```
        void add(string n, string ph);
        void print();
        string get_phone() { return phone; }
        string get_name() { return name; }
private:
        string name;
        string phone;
};

class normal : public person
{
public:
        void setnorm(bool a);
private:
        bool isnorm;
};

class vip : public person
{
public:
        void setvip(bool a);
private:
        bool isvip;
                                                                        };
```

فایل person.cpp

```
#include "person.h"
#include <iostream>
#include <string>
#include <iomanip>
using namespace std;
void person::add(string n, string ph) {
        name = n;
        phone = ph;
        }
void person::print(){
        cout << left << "Name: " << setw(16) << name << "\tPhone number: " << setw(12)
<< phone << endl;
}
void normal::setnorm(bool a) {
        isnorm = a;
}
void vip::setvip(bool a) {
        isvip = a;
                                                                        }
```

فیلد ها:

دو ویژگی نام و تلفن همراه برای این کلاس تعریف شده است که هر دو از نوع استرینگ هستند و برای بخش های تعریف مشتری و حذف مشتری و همچنین ذخیره و چاپ اطلاعات مشتری از این کلاس استفاده شده است.

متود ها:

متود های getter:

دو متود get_phone() و get_name() این دو متود شماره تلفن و نام مشتری را بصورت استرینگ در خارج از کلاس به خروجی می برند.

متود add() : این متود برای اضافه کردن اطلاعات مشتری و مقدار دادن به ویژگی های کلاس مورد استفاده قرار می گیرند.

متود print() : این متود هم برای چاپ اطلاعات مشتری است. در این متود از کتابخانه iomanip برای سازماندهی و نمایش اطلاعات مشتری استفاده شده است.

کلاس normal:

ویژگی isnorm بصورت private تعریف شده است و مشخص می کند که مشتری معمولی است.

متود setnorm یک مقدار بولین را می گیرد و در صورتیکه در این کلاس به عنوان ورودی باشد مقدار صفر را بر می گرداند.

کلاس vip:

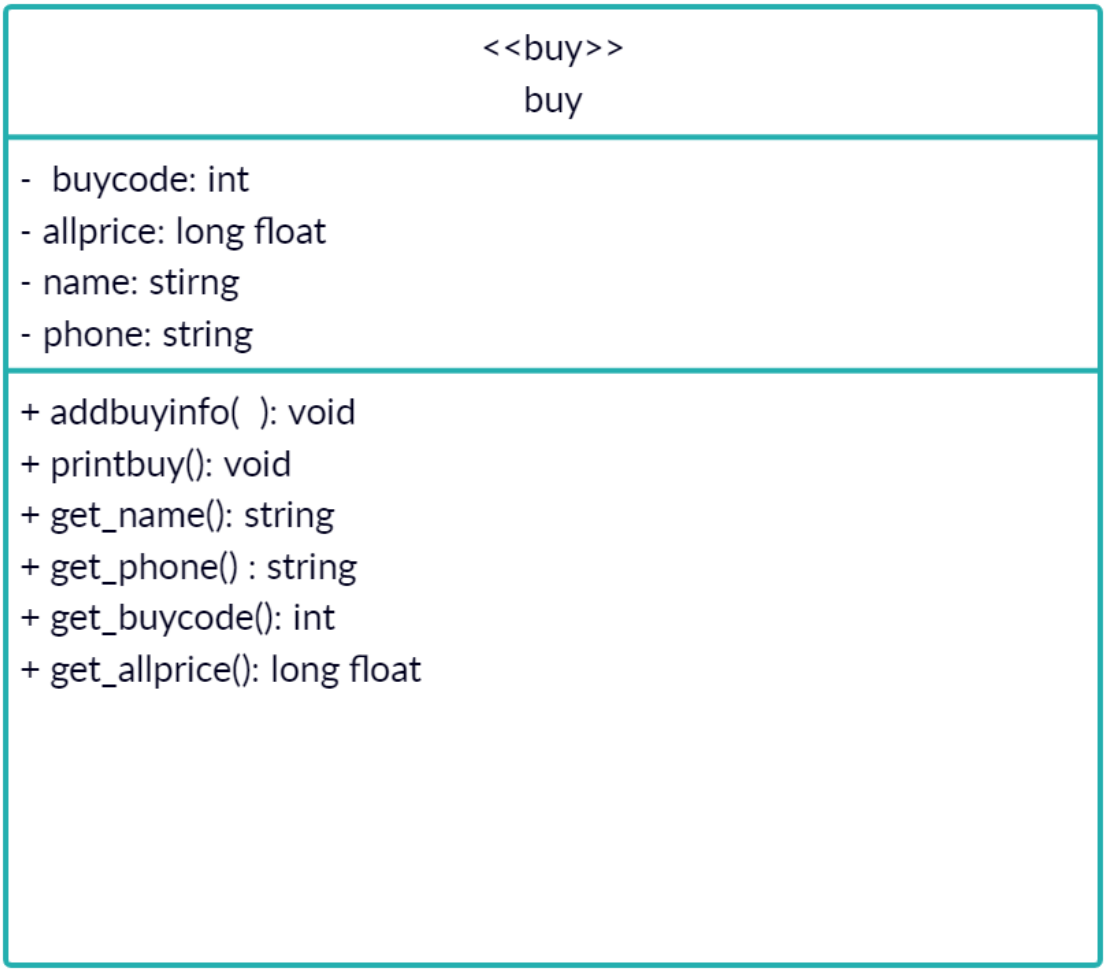
ویژگی isvip بصورت private تعریف شده است و مشخص می کند که مشتری وی آی پی است.

متود setvip یک مقدار بولین را می گیرد و در صورتیکه در این کلاس به عنوان ورودی باشد مقدار یک را بر می گرداند.

کلاس buy:

این کلاس برای خرید کاربر مورد استفاده قرار می گیرد.

UML



فایل buy.h

```
#pragma once
#include <string>
using namespace std;
class buy {
private:
	string name;
	string phone;
	int buycode;
	long float allprice;
public:
	void addbuyinfo(int bc, string n, string ph, long float pall);
	void printbuy();
	string get_name() { return name; };
```

```
        string get_phone() { return phone; };
        int get_buycode() { return buycode; };
        long float get_allprice() { return allprice; };

};
```

فایل buy.cpp

```
#include "buy.h"
#include "product.h"
#include <iostream>
#include <iomanip>
using namespace std;
void buy::addbuyinfo(int bc, string n, string ph, long float pall) {
        buycode = bc;
        name = n;
        phone = ph;
        allprice = pall;
}
void buy::printbuy(){
        cout << left << "BuyCode= " << setw(10) << buycode << "\tCustomer Name= " <<
setw(15) << name << "\tCustomer phone= " << setw(12) << phone << "\tAll Price= " <<
setw(20) << allprice << endl;
}
```

فیلد ها:

این فیلدها برای مشخصات خرید و خریدار مورد استفاده قرار می گیرد.

فیلد شناسه محصول برای گرفتن شناسه خرید قیمت که قیمت کل کالا ها را برای مشتری معمولی بدون تخفیف و برای مشتری وی آی پی با ۲۰ درصد تخفیف محاسبه می کند.

فیلدهای نام و phone برای گرفتن نام و شماره تلفن همراه مشتری.

در فرآیند خرید ابتدا شماره تلفن مشتری گرفته می شود و در لیست تمامی مشتریان جستجو می شود در صورتیکه مشتری موجود باشد فرآیند خرید ادامه می یابد. همچنین برای تک تک محصولات شناسه محصول جستجو می شود و در صورت موجود بودن محصول کالا به سبدخرید اضافه می شود.

متودها:

Addbuyinfo(): اطلاعات خرید را تنظیم می کند. ۴ مقدار را به عنوان ورودی می گیرد شامل تمام ویژگی ها کلاس که در بالا شرح داده شد.

Printbuy(): اطلاعات خرید را با استفاده از کتابخانه iomanip به صورت مرتب چاپ می کند.

Getter:

چهار getter نام، تلفن همراه، شناسه خرید و قیمت کل به ترتیب استرینگ، استرینگ، اینت و لانگ فلوت را به عنوان خروجی می دهد.

ارتباط بین کلاس ها:

کلاس محصولات هر یک از محصولات را به عنوان یک شی می گیرد و آن را در وکتوری با همین تایپ قرار می دهد.

رابطه بین کلاس فروشگاه و محصول shop & product یک رابطه association است زیرا یک مالک محصولات است.همچنین است رابطه بین فروشگاه و مشتری ها.

کلاس خرید و فروشگاه رابطه Composition دارند. زیرا اشیای کلاس فروشگاه که همان محصولات باشند از کلاس خرید برای ثبت خرید استفاده می شود. همچنین است رابطه بین کلاس های خرید و مشتری. زیرا کلاس خرید از لیست مشتریان استفاده می کند.

رابطه بین کلاس های person و normal و vip : یک رابطه وراثت است. همانطور که ذکر شد کلاس پرسون پدر دو کلاس نرمال و وی آی پی است.

در این نرم افزار از چهار فایل برای ذخیره اطلاعات محصولات فروشگاه، مشتری ها، سبد خریدها، خریداران استفاده شد.

با تشکر فراوان