 МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Компьютерный практикум №2**

з дисципліни «Об'єктно-орієнтоване програмування»

# на тему: «Робота з об'єктами, використання покажчика this»

Варіант №7

**Виконав:**

студент гр. БС-03

Затуловский Г. А.

**Перевірив:**

ас. каф. БМК Бабенко В.О.

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                      (підпис викладача)

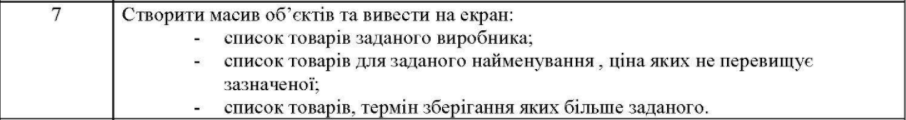
Київ-2022

**Завдання:**

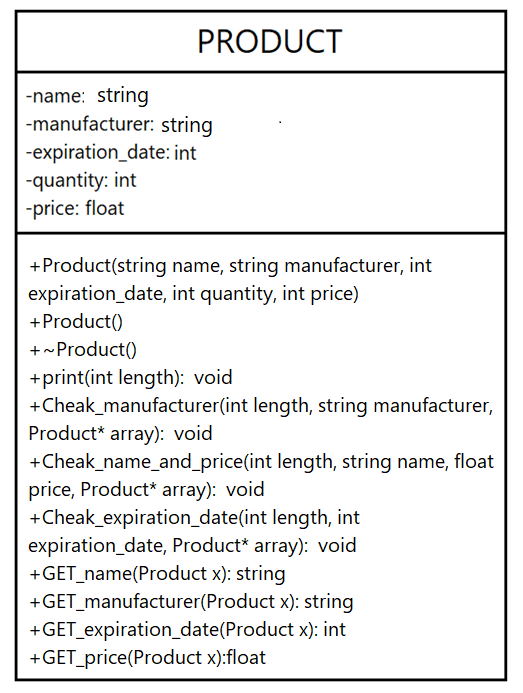
1. Вивчити теоретичні основи роботи з об'єктами та використання покажчика this.

2. Скористатися розробленим в попередній лабораторній роботі класом для розширення його функціональності (нові методи класу мають працювати з покажчиком this).

3. Вiдповідно до свого варіант розробити діаграму класів в нотації UML та написати програму, яка містить клас для роботи із запитами користувача (методи класу дозволяють відповідно до вибору користувача вивести на екран задані в завданнi списки) та iнкапсулює в собі масив об'єктів розробленого в попередній лабораторній роботі класу (визначення класів та їх реалізації мають бути розташовані у файлах \* h та \* .cpp вiдповiдно):



**UML діаграма:**



**Код програми:**

**PRODUCT.H**

#pragma once

#include <iostream>

#include "Product.h"

using namespace std;

class Product

{

private:

string name;

string manufacturer;

int expiration\_date;

int quantity;

int price;

public:

Product(string name, string manufacturer, int expiration\_date, int quantity, int price);

Product();

~Product();

void print(int length);

void Cheak\_manufacturer(int length, string manufacturer, Product\* array);

void Cheak\_name\_and\_price(int length, string name, float price, Product\* array);

void Cheak\_expiration\_date(int length, int expiration\_date, Product\* array);

string GET\_name(Product x);

string GET\_manufacturer(Product x);

int GET\_expiration\_date(Product x);

float GET\_price(Product x);

};

**OOP\_2.cpp**

#include <iostream>

#include "Product.h"

using namespace std;

int main()

{

int length = 0;

string name;

string manufacturer;

int expiration\_date;

float price;

cout << "Value of objects: ";

cin >> length;

Product\* array = new Product[length];

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

{

array[i].print(i);

}

cout << "Input manufacturer: ";

cin >> manufacturer;

array->Cheak\_manufacturer(length, manufacturer,array);

cout << "Input name: ";

cin >> name;

cout << "Input price: ";

cin >> price;

array->Cheak\_name\_and\_price(length, name, price, array);

cout << "Input expiration date: ";

cin >> expiration\_date;

array->Cheak\_expiration\_date(length, expiration\_date, array);

delete[] array;

return 0;

}

**Product.cpp**

#include "Product.h"

#include <iostream>

#include <string>

using namespace std;

Product::Product(string name, string manufacturer, int expiration\_date, int quantity, int price)

{

this->name = name;

this->manufacturer = manufacturer;

this->expiration\_date = expiration\_date;

this->quantity = quantity;

this->price = price;

}

Product::Product()

{

string choise\_name[5] = { "Licorice","Chocolate","Hard\_Candies","Lollipops","Caramels" };

string choise\_manufacturer[5] = { "Roshen","Ferrero","Hershey","Nestle","Meiji" };

name = choise\_name[rand() % 5];

manufacturer = choise\_manufacturer[rand() % 5];

expiration\_date= 15 + rand() % 345;

quantity=rand() % 1000;

price= 14 + rand() % 100;

}

Product::~Product()

{

}

void Product::print(int number)

{

int cheak\_number = 0;

cout << "#" << number+1 << endl

<< "Name: " << name

<< "\nManufacturer: " << manufacturer

<< "\nPrice: " << to\_string(price)

<< "\nExpiration\_date: " << to\_string(expiration\_date)

<< "\nQuantity: " << to\_string(quantity) << endl << endl;

}

void Product::Cheak\_manufacturer(int length, string manufacturer, Product \* array)

{

for (int j = 0; j < length; j++)

{

if (manufacturer == GET\_manufacturer(array[j]))

{

cout << endl;

array[j].print(j);

}

}

}

void Product::Cheak\_name\_and\_price(int length, string name, float price, Product\* array)

{

for (int j = 0; j < length; j++)

{

if (name == GET\_name(array[j]) && price >= GET\_price(array[j]))

{

cout << endl;

array[j].print(j);

}

}

}

void Product::Cheak\_expiration\_date(int length, int expiration\_date, Product\* array)

{

for (int j = 0; j < length; j++)

{

if (expiration\_date <= GET\_expiration\_date(array[j]))

{

cout << endl;

array[j].print(j);

}

}

}

string Product::GET\_name(Product x)

{

return x.name;

}

string Product::GET\_manufacturer(Product x)

{

return x.manufacturer;

}

int Product::GET\_expiration\_date(Product x)

{

return x.expiration\_date;

}

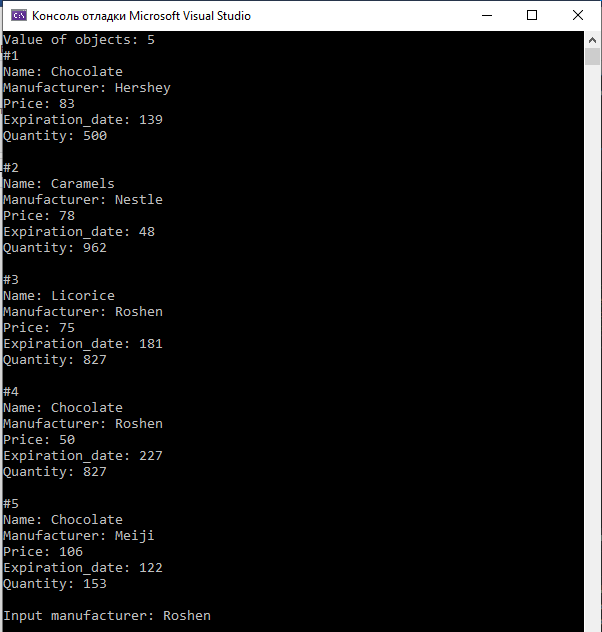
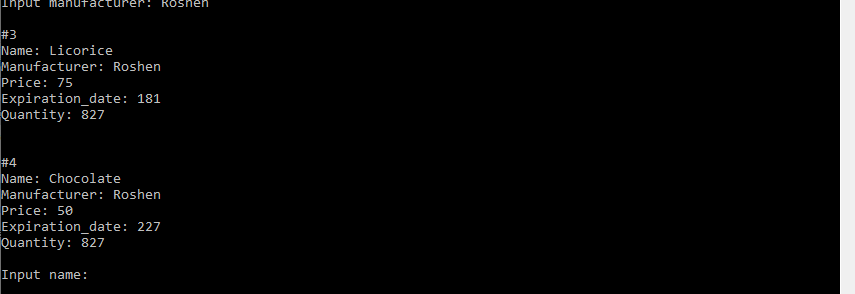
float Product::GET\_price(Product x)

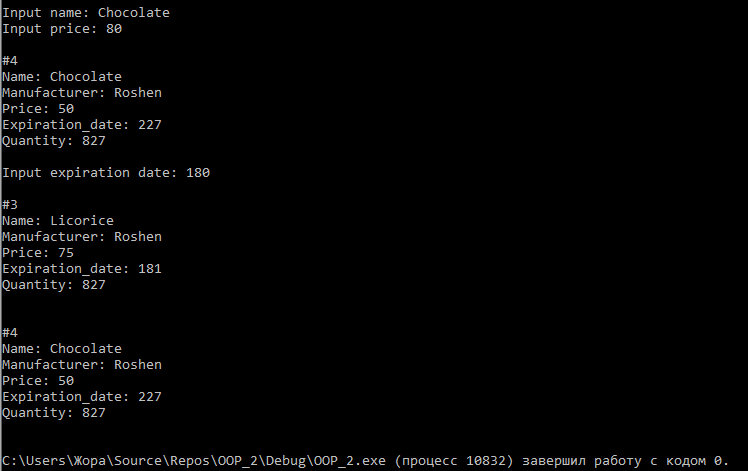
{

return x.price;

}

**Результати:**



**Контрольні запитання:**

**1. Що таке покажчик this?**

Вказівник цього зберігає адресу певного об'єкта класу

**2. Навести приклад, в якому не можна організувати код без використання покажчика this.**

Class Product:: GET\_name()

{

return \*this;

}

**3. Два способи iнiцiалiзації об'єктів у програмі.**

1.Конструктор за замовчуванням,

2.Конструктор копіювання.

**4. Як ініціалізувати масив об'єктів?**

Class Pixel

{

Public:

{

Pixel(int r, int g,int b)

{

This -> r = r;

This -> g = g;

This -> b = b;

}

Private:

Int r;

Int g;

Int b;

}

Imt main()

{

Int len = 5

Pixel arr[len]

{

Pixel(1,2,3),

Pixel(16,4,34),

Pixel(42,12,33),

Pixel(12,26,36),

Pixel(12,12,53),

};

}

**5. Чому не можна ініціалізувати поле, що є масивом, під час визначення класу?**

Виділяти пам'ять можна лише при створенні об'єкта, а не при його оголошенні

**6. Перелічити випадки, коли один клас використовує інший.**

Під час спідкування

**7. Які є види вiдношень мiж класами?**

Відношення типу has-a (клас містить інший клас). У цьому випадку в класі оголошується один або декілька екземплярів іншого класу. При даному відношенні можливі два випадки взаємодії. Перший випадок, це коли об’єкт (екземпляр), що оголошений в класі, не є складовою частиною класу (агрегація) і його використання не впливає на функціональну роботу класу. Другий випадок, коли об’єкт, що оголошений в класі, є складовою частиною цього класу (композиція).

**8. Які види вiдношень мiж класами визначені в мові моделювання UML?**

Агрегація, композиція, залежність, асоціація,

**9. Який тип відношення встановлено мiж класами в даній лабораторній роботі?**

Відношення композиції використовується у даній роботі.

**10. Що означає оператор (::)?**

(::) – це оператор, який показує поле видимості змінної до якої він застосован