 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Комп’ютерний практикум №2**

з дисципліни «Об’єктно-орієнтоване програмування»

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

Варіант №7

**Виконав:**

студент гр. БС-82

Грініх К. А.

**Перевірив:**

ас. каф. БМК

Рисін С. В.

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

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

(підпис викладача)

Київ-2020

**🞏 Практичне заняття без зауважень**

**🞏 Практичне заняття має зауваження:**

**🞏 несвоєчасний захист**

**🞏 присутні зауваження до UML діаграми:**

**🞏 діаграма класу не відповідає коду**

**🞏 виконані не за стандартом:**

**🞏 атрибути**

**🞏 відношення**

**🞏 потужність**

**🞏 інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🞏 присутні зауваження до коду:**

**🞏 задача завдання вирішена хибно**

**🞏 код програми не компілюється**

**🞏 хибно задані специфікатори доступу**

**🞏 помилки у визначенні конструкторів / деструкторів**

**🞏 відсутні списки ініціалізації в конструкторах**

**🞏 константні методи**

**🞏 використано глобальні змінні**

**🞏 статичні змінні при роботі з масивами**

**🞏 оформлення коду**

**🞏 присутні зайві символи «{» та «}»**

**🞏 інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🞏 невірні відповіді на запитання:**

**🞏 №1 🞏 №2 🞏 №3 🞏 №4 🞏 №5**

**🞏 №6 🞏 №7 🞏 №8 🞏 №9 🞏 №10**

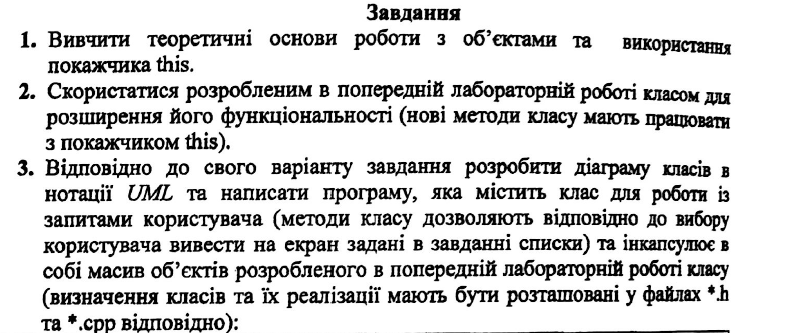
**🞏 маються інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Завдання:**



7. Створити масив об’єктів та вивести на екран:

-список товарів заданого виробника;

-список товарів для заданого найменування, ціна яких не перевищує зазначеної;

-список товарів, термін зберігання яких більше заданого.

**Діаграма:**



**Лістинг програми:**

Файл Product.h:

#include <iostream>

#include <cstring>

using namespace std;

class Product

{

private:

char\* m\_name, \* m\_producer;

unsigned int m\_price, m\_quantity;

unsigned short m\_term;

public:

Product(const char\* name, const char\* producer, unsigned int price, unsigned short term, unsigned int quantity);

~Product();

void set\_name(const char\* name);

void set\_producer(const char\* producer);

void set\_price(unsigned int price);

void set\_term(unsigned short term);

void set\_quantity(unsigned int quantity);

const char\* get\_name() const { return m\_name; }

const char\* get\_producer() const { return m\_producer; }

unsigned int get\_price(void) const { return m\_price; }

unsigned short get\_term() const { return m\_term; }

unsigned int get\_quantity(void) const { return m\_quantity; }

};

Файл Product.cpp:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Product.h"

#include <cstdio>

#include <iostream>

using namespace std;

Product::Product(const char\* name, const char\* producer, unsigned int price, unsigned short term, unsigned int quantity) :

m\_name(), m\_producer(), m\_price(price), m\_term(term), m\_quantity(quantity)

{

set\_name(name);

set\_producer(producer);

}

Product::~Product()

{

delete[] m\_name;

delete[] m\_producer;

}

void Product::set\_name(const char\* name)

{

delete[] m\_name;

this->m\_name = new char[strlen(name) + 1];

strcpy(m\_name, name);

}

void Product::set\_producer(const char\* producer)

{

delete[] m\_producer;

this->m\_producer = new char[strlen(producer) + 1];

strcpy(m\_producer, producer);

}

void Product::set\_price(unsigned int price)

{

this->m\_price = price;

}

void Product::set\_term(unsigned short term)

{

this->m\_term = term;

}

void Product::set\_quantity(unsigned int quantity)

{

this->m\_quantity = quantity;

}

Файл Product\_Array.h:

#pragma once

#include "Product.h"

struct Info

{

const char\* m\_name;

const char\* m\_producer;

unsigned int m\_price;

unsigned short m\_term;

unsigned int m\_quantity;

};

class Product\_Array

{

private:

Product\*\* Arr;

unsigned int Size;

public:

Product\_Array(const Info\* inf, unsigned int size);

~Product\_Array();

void p\_name(const char\* producer);

void p\_price(unsigned int price, const char\* name);

void p\_date(unsigned short term);

};

Файл Product\_Array.cpp:

#include <string.h>

#include <stdio.h>

#include <iostream>

#include "Product\_Array.h"

using namespace std;

Product\_Array::Product\_Array(const Info \*inf, int size) : Size(size)

{

Arr = new Product \*[Size];

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

Arr[i] = new Product

(inf[i].m\_name,

inf[i].m\_producer,

inf[i].m\_price,

inf[i].m\_term,

inf[i].m\_quantity);

}

Product\_Array::~Product\_Array()

{

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

delete Arr[i];

delete[] Arr;

}

void Product\_Array::p\_name(const char \*producer)

{

cout << "The list of the products from your producer: " << endl;

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

{

if ((strcmp(Arr[i]->get\_producer(), producer)==0)

{

cout << Arr[i]->get\_name() << endl;

}

}

}

void Product\_Array::p\_price(unsigned int price, const char \*name)

{

cout << "The list of the products with lower or the same price: " << endl;

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

{

if ((strcmp(Arr[i]->get\_name(), name)==0) && (Arr[i]->get\_price() <= price))

{

cout << Arr[i]->get\_name()<< " '"<<Arr[i]->get\_producer() << "' (" << Arr[i]->get\_price() << " grn)" << endl;

}

}

}

void Product\_Array::p\_date(unsigned short term)

{

cout << "The list of the products with longer term: " << endl;

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

{

if (Arr[i]->get\_term() > term)

{

cout << Arr[i]->get\_name() << " (" << Arr[i]->get\_term()<<" days)"<< endl;

}

}

}

Файл main.cpp:

#include <iostream>

#include <conio.h>

#include "Product\_Array.h"

using namespace std;

int main()

{

Info arr[] =

{

{"Milk", "Galychyna", 26, 12, 47},

{"Russian Cheese", "Ferma", 138, 72, 98},

{"Condensed Milk", "Myhalych", 27, 234, 125},

{"Cottage Cheese", "Shostka", 43, 7, 16},

{"Yogurt", "Galychyna", 11, 28, 51},

{"Yogurt", "Zlagoda", 9, 26, 49},

{"Yogurt", "Molokia", 13, 50, 39}

};

Product\_Array\* Input = new Product\_Array(arr, sizeof(arr) / sizeof(arr[0]));

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

// The list of the products from your producer:

const char producer[20]= " ";

cout << "Print a producer: ";

cin >> producer;

cout << " " << endl;

Input->p\_name(producer);

cout << " " << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " " << endl;

// The list of the products with lower or the same price:

unsigned int price=0;

const char name[20] = " ";;

cout << "Print a name of product: ";

cin >> name;

cout << "Print a price: ";

cin >> price;

cout << " " << endl;

Input->p\_price(price, name);

cout << " " << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " " << endl;

//The list of the products with longer expiration date:

unsigned short days=0;

cout << "Print the term: ";

cin >> days;

cout << " " << endl;

Input->p\_date(days);

cout << " " << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

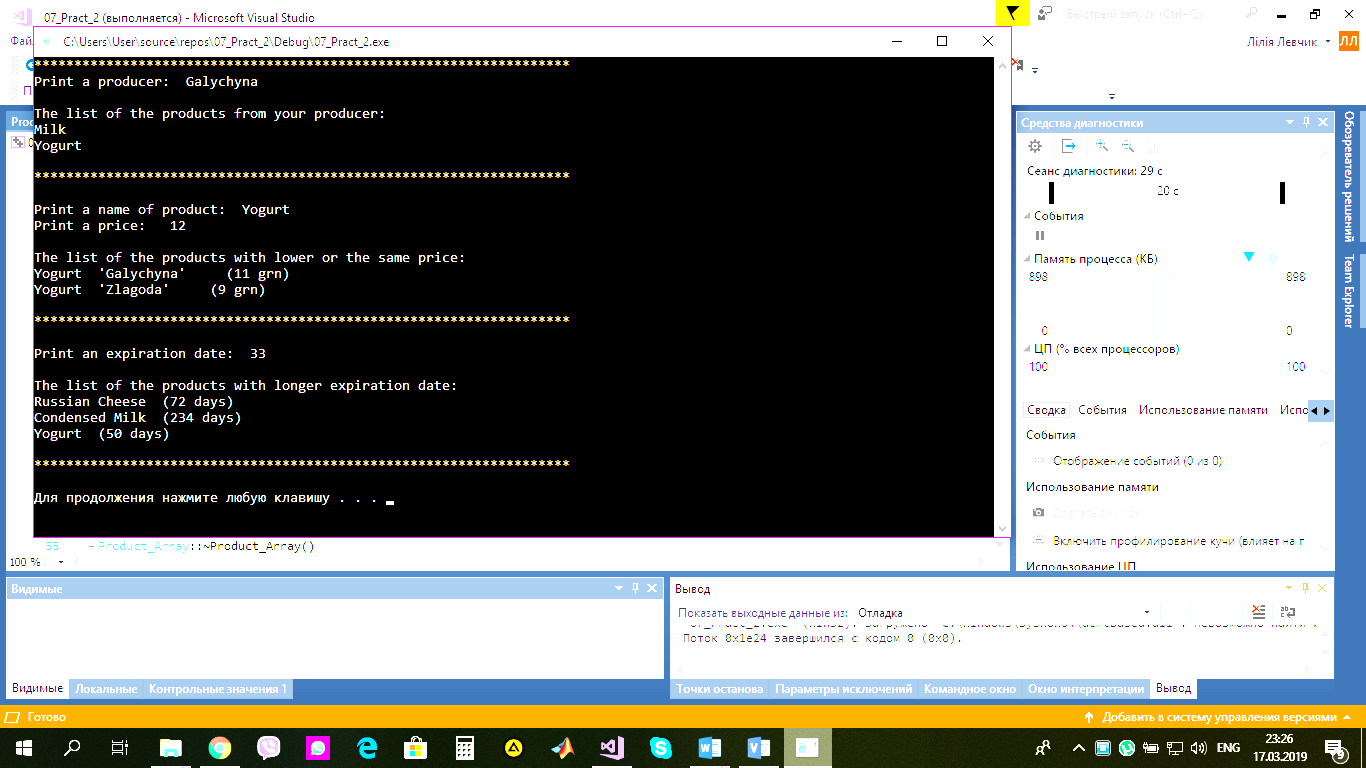
cout << " " << endl;

delete Input;

system("pause");

}

**Скріншот результатів виконання:**



**Відповіді на контрольні запитання:**

1. Що таке покажчик this?

this - це покажчик, який зберігає в собі адресу об'єкта класу. Коли функція, що належить класу, викликається для обробки даних конкретного об'єкта, цій функції [автоматично](http://ua-referat.com/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) і неявно передається покажчик (this) на той об'єкт, для якого функція викликана.

1. Навести приклад, в якому не можна організовувати код без використання покажчика this?

MyClass MyClass::getMe() { return \*this; }

1. Два способи ініціалізації об’єктів у програмі.

Створення об’єкта за допомогою конструктора за замовчуванням:

SomeClass SomeObject;

Створення об’єкта за допомогою конструктора з параметрами:

SomeClass SomeObject (список параметрів);

1. Як ініціалізувати масив об’єктів?

  Product\* array[7];

  for(int i=0;i<7; i++)

   array[i] = new Product(i);

Or:

  Product array[7];

Or:

  Product array[]= new   Product[n];

Or:

  Product array[2]={   Product (7,2), Product (9,4)};

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

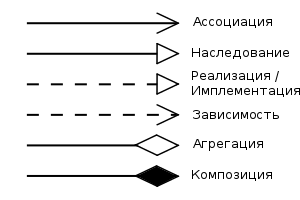
Ініціалізація передбачає виділення пам'яті, а пам’ять виділяється при створенні, а не при визначенні класу.

1. Перелічити випадки, коли один клас використовує інший.

* наслідування (є абстрактний клас – «людина» і більш спеціалізований (його нащадок) «робітник»),
* асоціація (показує відношення між об’єктами класу)
  + бінарна: одному екземпляру одного класу відповідає лише один екземпляр іншого класу (наприклад, «робітник» - «трудова книжка»)
  + н-арна: : одному екземпляру одного класу може відповідати один і більше екземплярів іншого класу («робітник» - «мобільний телефон»(може бути робочий і для особистих дзвінків)
* агрегація («назва робочого відділу» на кожному з відділів може бути від одного та більше робітників, але є і такий випадок що робітник фірми не належить жодному відділу (наприклад, начальник фірми))
* комопзиція (це створення об'єктів існуючих класів як елементів інших класів. Про композицію також часто говорять як про «відношення приналежності» за принципом у «у машини є корпус, колеса і двигун».)
* залежність (коли в сигнатурі операції одного класу є аргумент іншого класу, або коли в методі одного класу є локальний об’єкт іншого класу, або коли результатом операції одного класу є екземпляр іншого класу)

1. Які є види відношень між класами?
   * Залежності
   * Узагальнення (успадкування, наслідування)
   * Асоціації
   * Реалізації
   * Композиції
   * Агрегації
2. Які види відношень між класами визначені в мові моделювання UML?

Успадкування, асоціація, що показує відношення між об’єктами-екземплярами класу (бінарна – потужність зв’язку між класами 1 до 1, н-арна – потужність зв’язку один до багатьох), агрегація, композиція, залежність (зміни в одному класі потребують змін в іншому), реалізація – оголошення інтерфейсу та можливість його реалізації будь-яким класом.



1. Який тип відношення встановлено між класами в даній лабораторній роботі?

Відношення агрегації та залежності.

1. Що означає оператор (::)?

Такий оператор уточнює область видимості в якій знаходиться ім’я. Зліва від оператора повинна бути вказана область видимості, справа - ім’я, що уточнюється.

Використовується для визначення методів класу поза header файлом, де клас був оголошений.