МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Комп’ютерний практикум №4**

з дисципліни «Об’єктно-орієнтоване програмування»

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

Варіант №5

**Виконав:**

студент гр. БС-71

Воробйов І.О.

**Перевірив:**

асистент каф. БМК

Рисін С.В.

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

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

(підпис викладача)

Київ-2019

**🞏 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:**

1. Вивчити теоретичні основи перевантаження операторів в мові С++.
2. Розробити діаграму класів в нотації UML та програмний застосунок для тестування всіх перевантажених операцій класу відповідно до свого варіанту (визначення класів та їх реалізації мають бути розташовані в у файлах \*.h та \*.cpp відповідно; для ініціалізації даних класів використовувати введення з клавіатури; кожен клас, окрім зазначених у варіанті операцій, має містити конструктор з параметрами, конструктор копіювання, деструктор, методи встановлення і отримання значень всіх полів класу):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер варіанту | Клас | Поля | Методи класу | Операції для перевантаження |
| 5 | Парабола | Коефіцієнти ***a, b, c*** | Методи виведення рівняння параболи на екран, обчислення екстремуму функції (***min*** або ***max***) | Перевірка збігу двох парабол (| |), перевірка перетину двох парабол (/) |

1. Скласти і захистити звіт по роботі.

**UML діаграма**



**Код програми:**

**parabola.h:**

#pragma once

#include <iostream>

#include <string>

using namespace std;

class parabola

{

private:

double a, b, c;

public:

parabola();

~parabola();

parabola(double, double, double);

parabola(const parabola&);

void show() const;

void set\_a(double);

void set\_b(double);

void set\_c(double);

double get\_a() const;

double get\_b() const;

double get\_c() const;

double find\_maxORmin(double&);

friend bool operator || (parabola&, parabola&);

friend bool operator / (parabola&, parabola&);

};

**parabola.cpp:**

#include "parabola.h"

parabola::parabola() : a(1), b(1), c(1) {}

parabola::~parabola() {}

parabola::parabola(double x1, double x2, double x3): a(x1), b(x2), c(x3) {}

parabola::parabola(const parabola& tmp) : a(tmp.a), b(tmp.b), c(tmp.c) {}

void parabola::show() const

{

cout << "y = ";

if (a != 0 || b != 0 || c != 0)

{

if (a != 0)

if (a == 1)

cout << "x^2";

else

cout << a << "\*x^2";

if (b != 0)

{

if (b == 1)

if (a != 0)

cout << "+x";

else

cout << "x";

else if (a == 0)

cout << b << "\*x";

else

cout << (b > 0 ? "+" : "") << b << "\*x";

}

if (c != 0)

cout << (c > 0 ? "+" : "") << c << endl;

else

cout << endl;

}

else

cout << "0" << endl;

}

double parabola::get\_a() const

{

return a;

}

double parabola::get\_b() const

{

return b;

}

double parabola::get\_c() const

{

return c;

}

void parabola::set\_a(double x)

{

a = x;

}

void parabola::set\_b(double x)

{

b = x;

}

void parabola::set\_c(double x)

{

c = x;

}

double parabola::find\_maxORmin(double &x)

{

x = (-b / (2 \* a));

double y = a \* pow(x, 2) + b \* x + c;

return y;

}

bool operator || (parabola &one, parabola &two)

{

double oneX = (-one.get\_b() / (2 \* one.get\_a())), twoX = (-two.get\_b() / (2 \* two.get\_a()));

double oneY = one.get\_a()\*pow(oneX, 2) + one.get\_b()\*oneX + one.get\_c(),

twoY = two.get\_a()\*pow(twoX, 2) + two.get\_b()\*twoX + two.get\_c();

if (oneX == twoX && oneY == twoY)

return true;

else

return false;

}

bool operator / (parabola &one, parabola &two)

{

double newa = (one.get\_a() - two.get\_a()), newb = (one.get\_b() - two.get\_b()), newc = (one.get\_c() - two.get\_c());

if ((pow(newb, 2) - 4 \* newa\*newc) < 0)

return false;

else

return true;

}

**Source.cpp:**

#include "parabola.h"

int main()

{

double a, b, c;

cout << "Input obj1:\nInput a = "; cin >> a;

cout << "Input b = "; cin >> b;

cout << "Input c = "; cin >> c;

parabola\* obj1 = new parabola(a, b, c);

cout << "Input obj2:\nInput a = "; cin >> a;

cout << "Input b = "; cin >> b;

cout << "Input c = "; cin >> c;

parabola\* obj2 = new parabola(a, b, c);

parabola obj3(\*obj1);

cout << "obj1:\n";

obj1->show();

cout << "obj2:\n";

obj2->show();

cout << "obj3:\n";

obj3.show();

cout << "parabolas come together - is " << ((\*obj1 || \*obj2) ? "true" : "false") << endl;

cout << "parabolas intersect - is " << ((\*obj1 / \*obj2) ? "true" : "false") << endl;

if (obj1->get\_a() == 0)

cout << "It's not parabola." << endl;

else

{

double x = 0;

double y = obj1->find\_maxORmin(x);

cout << (obj1->get\_a() ? "minimum" : "maximum") << " in obj1 = (" << y << "," << x << ")\n";

}

if (obj2->get\_a() == 0)

cout << "It's not parabola." << endl;

else

{

double x = 0;

double y = obj2->find\_maxORmin(x);

cout << (obj2->get\_a() ? "minimum" : "maximum") << " in obj2 = (" << y << "," << x << ")\n";

}

delete obj1;

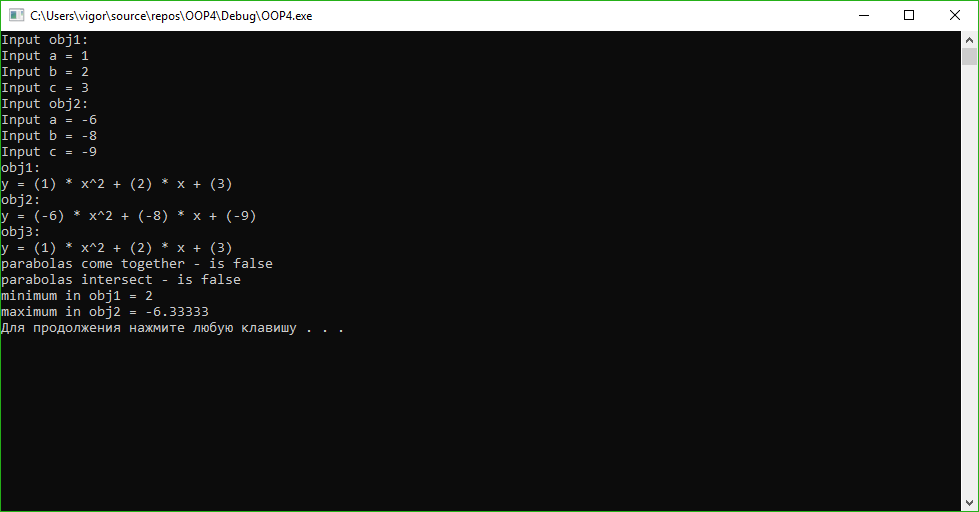
delete obj2;

system("pause");

return 0;

}

**Результат роботи програми:**



**Контрольні запитання**

1. Навіщо потрібне перевантаження операторів?

Позволяет записывать операции с пользовательскими классами в общепринятом виде

1. Якими двома різними способами визначаються перевантажені операції?

Методами класса и глобальными функциями.

1. Які оператори мови С++ не можуть бути перевантажені?

Нельзя перезагрузить: оператор доступа к членам класса(.),операция доступа к членам класса через разименования(.\*), оператор размещения области видимости(::), тернарный оператор условия(?:)

1. Які операції не можна перевантажити за допомогою глобальної дружньої функції? Відповідь пояснити.

За допомогою глобальної дружньої функції не можна перевантажувати такі оператори : =, (), [] і ->. Перевантаження оператора для класу можна реалізувати і з використанням функції, що не є членом цього класу(функції - «друзі» класу). Функції-"друзі" не мають покажчика this. Отже, якщо для перевантаження бінарного оператора використовується функція-"друг", явно передаються обидва операнда. Якщо ж за допомогою функції- "друга" перевантажується унарний оператор, операторної функції передається один оператор.

1. У яких випадках операції можна перевантажити тільки за допомогою глобальної функції? Когда левый оперант нне является объектом класса, для которого выполняется перегрузка
2. У яких випадках глобальна операція – функція повинна бути дружньою?

Дружні функції необхідні тоді, коли операція проводиться над об'єктами різних класів і для того, щоб ми мали доступ до приватних членів класу.

1. Чи обов’язково у функції оператор параметр типу «клас» або «посилання на клас»? Відповідь поясніть

Не обов’язково. Якщо оператор є членом класа , то об’єкт до якого він застосовується передається не явно через this і якщо серед аргументів немає класів , то передача такого параметру не потрібна.

1. Чи можна успадкувати або повторно перевантажити в похідному класі операцію, що була перевантажена у базовому класі?

Перевантажити все, унаслідується все окрім присвоєння.

1. Чому не можна перевантажувати деструктори?

Деструктори не можна перевантажувати через відсутність у них аргументів.

1. Скільки необхідно передати параметрів функції-методу класу , що реалізує перевантаження бінарної операції?

1 параметр , що задає другий операнд.