Міністерство освіти і науки України

Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатика та програмної інженерії

Звіт

З лабораторної роботи № 4 з дисципліни “Основи програмування-2” “Перегрузка операторів”

Варіант 2

Виконав: ІП-13 Бабашев Олексій

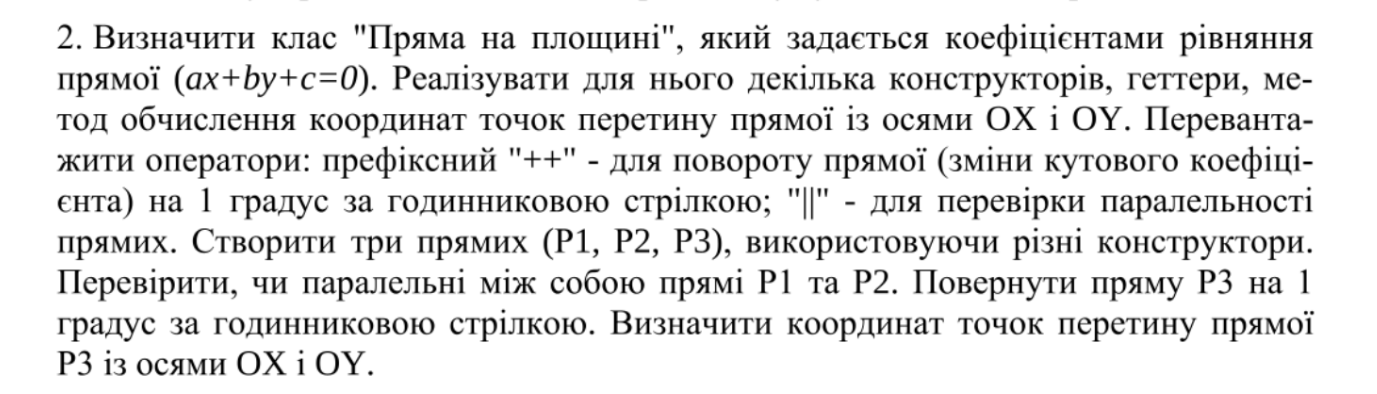
Перевірила: Вєчерковська А. С.

Київ - 2022

**Лабораторна робота 4**

**Класи та об’єкти**

**Індивідуальне завдання:**



**Виконання програми в терміналі та код на С++:**

**Вихідний код:**

**Lab4.cpp**

#include "Functions.h"

int main()

{

Line P1;

cout << "1) 1x - 1y + 1 = 0" << '\n' << '\n';

double a, b, c; cout << "a = "; cin >> a; cout << "b = "; cin >> b; cout << "c = "; cin >> c;

Line P2(a, b, c);

cout << "2) "<< a <<"x + "<<b << "y + " << c <<" = 0" << '\n';

if (P1 || P2)

{

cout << "P1 and P2 is paralel" << '\n' << '\n';

}

else

{

cout << "P1 and P2 is NOT paralel" << '\n' << '\n';

}

Line P3(P1);

cout << "3) 1x - 1y + 1 = 0" << '\n' << '\n';

cout << "line3 degrees = " << P3.get\_Degrees() << '\n';

cout << "k = " << P3.get\_k() << '\n' ;

cout << "Ox: (" << P3.Ox() << ";0) " << " Oy: (0; " << P3.Oy() << ")" << '\n' << '\n';

++P3;

cout << "line3 degrees = " << P3.get\_Degrees() << '\n';

cout << "k = " << P3.get\_k() << '\n' ;

cout << "Ox: (" << P3.Ox() << ";0) " << " Oy: (0; " << P3.Oy() << ")" << '\n';

}

**Functions.cpp**

#include "Functions.h"

double Line::RadianToDegrees()

{

return Radian \* (180.0 / M\_PI);

}

double Line::DegreesToRadian()

{

return Degrees \* (M\_PI / 180.0);

}

Line::Line()

{

a = 1;

b = -1;

c = 1;

Radian = M\_PI / 4;

Degrees = 45;

k = tan(Radian);

}

Line::Line(Line& copyClass)

{

this->a = copyClass.get\_a();

this->b = copyClass.get\_b();

this->c = copyClass.get\_c();

this->Radian = copyClass.get\_Radian();

this ->Degrees = copyClass.get\_Degrees();

this->k = copyClass.get\_k();

}

Line::Line(double a, double b, double c)

{

this->a = a;

this->b = b;

this->c = c;

if (b==0)

{

this->Radian = M\_PI / 2;

this->Degrees = 90;

this->k = 1;

}

else {

Radian = atan((double)(-a) / (double)b);

Degrees = RadianToDegrees();

this->k = tan(Radian);

}

}

double Line::Ox() {

if (c == 0) {

return 0;

}

else {

return -(c / a);

}

}

double Line::Oy() {

if (c == 0) {

return 0;

}

else {

return -(c / b);

}

}

void Line::operator++()

{

this->Degrees -= 1;

Radian = DegreesToRadian();

this->k = tan(Radian);

if (c!=0)

{

this->a = c / k;

}

}

bool Line::operator||(Line lineToCheckWith)

{

if (this->a/lineToCheckWith.get\_a() == this->b/lineToCheckWith.get\_b())

{

return true;

}

else {

return false;

}

}

double Line::get\_a() {

return a;

}

double Line::get\_b() {

return b;

}

double Line::get\_c() {

return c;

}

double Line::get\_k() {

return k;

}

double Line::get\_Degrees() {

return Degrees;

}

double Line::get\_Radian() {

return Radian;

}

**Functions.h**

#pragma once

#define \_USE\_MATH\_DEFINES

#define TO\_DEGREES (180 / M\_PI)

#include <iostream>

#include<cmath>

#include<vector>

using namespace std;

class Line

{

public:

Line();

Line(Line& a);

Line(double, double, double);

double get\_a();

double get\_b();

double get\_c();

double get\_k();

double get\_Degrees();

double get\_Radian();

double Ox();

double Oy();

void operator++();

bool operator||(Line lineToCheckWith);

private:

double a, b, c;

double k;

double Radian, Degrees;

double RadianToDegrees();

double DegreesToRadian();

};

**Висновок:** Вивчив особливості перегрузки операторів класів. Застосував отриманні знання на практиці.