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

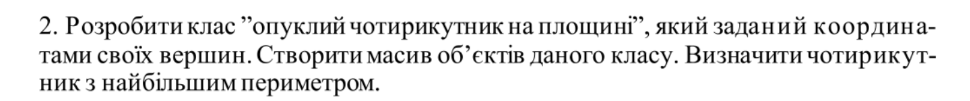
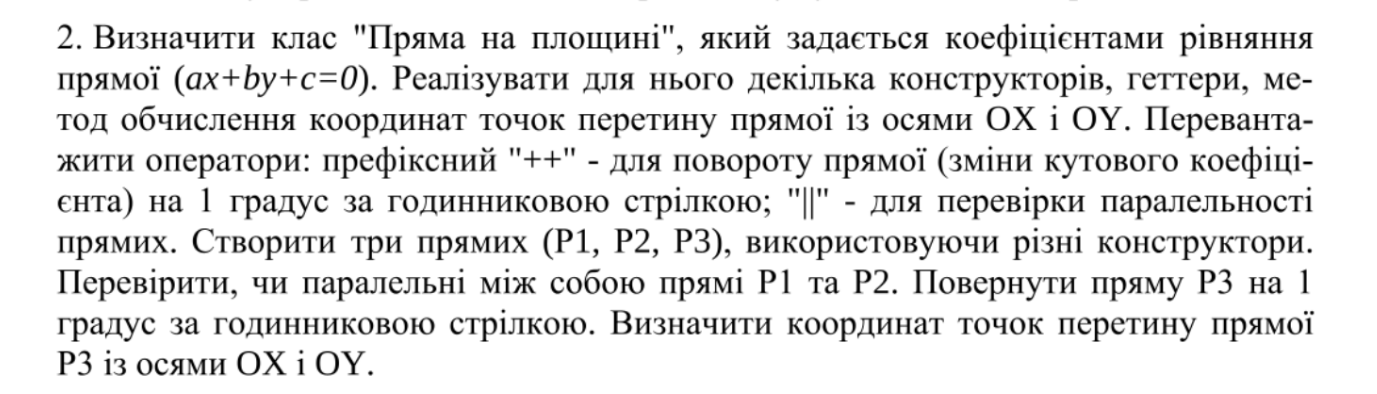
Перевірила: Вєчерковська А. С.

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 line1;

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

cout << "Oy: x = 0 " << " y = " << line1.Ox() << " Ox: x = " << line1.Oy() << " y = 0 " << '\n';

int c1; cout << "c = "; cin >> c1;

cout << "2) 1x + 1y + "<<c1<<" = 0" << '\n';

Line line2(c1);

cout << "Oy: x = 0 " << " y = " << line2.Ox() << " Ox: x = " << line2.Oy() << " y = 0 " << '\n';

int a, b, c2; cout << "a = "; cin >> a; cout << "b = "; cin >> b; cout << "c = "; cin >> c2;

cout << "2) "<< a <<"x + "<<b << "y + " << c2 <<" = 0" << '\n';

Line line3(a, b ,c2);

cout << "Oy: x = 0 " << " y = " << line3.Ox() << " Ox: x = " << line3.Oy() << " y = 0 " << '\n' << '\n';

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

cout << "k = " << line3.get\_k() << '\n';

++line3;

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

cout << "k = " << line3.get\_k() << '\n' << '\n';

if (line1 || line2)

{

cout << "1 and 2 is paralel" << '\n';

}

else if(line2 || line3)

{

cout << "2 and 3 is paralel" << '\n';

}

else if (line3 || line1)

{

cout << "3 and 1 is paralel" << '\n';

}

else

{

cout << "noone is paralel" << '\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 = 0;

Radian = M\_PI / 4;

Degrees = 45;

this->k = tan(Radian);

}

Line::Line(int c)

{

a = 1;

b = 1;

this->c = c;

Radian = 3 \* M\_PI / 4;

Degrees = 135;

this->k = tan(Radian);

}

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

{

this->a = a;

this->b = b;

this->c = c;

if (b==0)

{

Radian = M\_PI / 2;

Degrees = 90;

this->k = 0;

}

else {

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

Degrees = RadianToDegrees();

this->k = tan(Radian);

}

}

int Line::Ox() {

int x;

if (a==0)

{

cout << " the line lies on Ox or parallel to the axis! "<<'\n';

return 0;

}

else {

x = -(c/a);

return x;

}

}

int Line::Oy() {

int y;

if (b==0)

{

cout << " the line lies on Oy or parallel to the axis! " << '\n';

return 0;

}

else {

y = -(c / b);

return y;

}

}

void Line::operator++()

{

this->Degrees += 1;

Radian = DegreesToRadian();

this->k = tan(Radian);

}

bool Line::operator||(Line lineToCheckWith)

{

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

{

return true;

}

else {

return false;

}

}

int Line::get\_a() {

return a;

}

int Line::get\_b() {

return b;

}

int 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(int);

Line(int, int, int);

int get\_a();

int get\_b();

int get\_c();

double get\_k();

double get\_Degrees();

double get\_Radian();

int Ox();

int Oy();

void operator++();

bool operator||(Line lineToCheckWith);

private:

int a, b, c;

double k;

double Radian, Degrees;

double RadianToDegrees();

double DegreesToRadian();

};

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