Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №1 з дисципліни

«Основи програмування»

Варіант 2

Виконав студент Боровков Іван Ігорович

( прізвище, ім'я, по батькові)

Перевірив викладач Вітковська Ірина Іванівна

( прізвище, ім'я, по батькові)

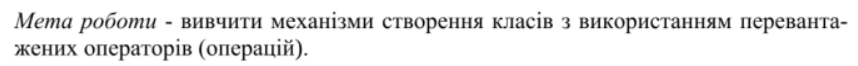
Київ 2022

Лабораторна робота №2

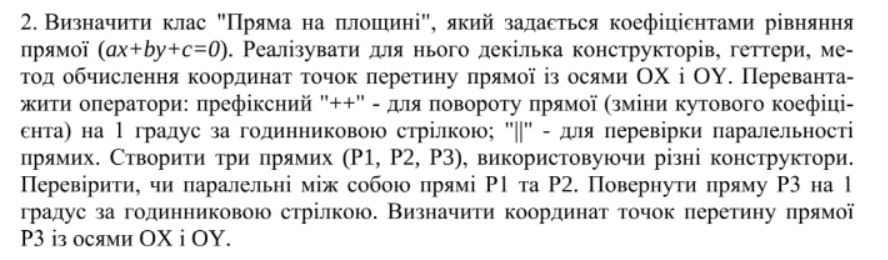
[ПЕРЕВАНТАЖЕННЯ ОПЕРАТОРІВ](http://ipi.kpi.ua/study/ii-semestr/laboratorni-roboti/lpboratorna-robota-3-perevantazhennya-operatoriv/)

Лабораторна робота №4

[ПЕРЕВАНТАЖЕННЯ ОПЕРАТОРІВ](http://ipi.kpi.ua/study/ii-semestr/laboratorni-roboti/lpboratorna-robota-3-perevantazhennya-operatoriv/)



Умова задачі:



Код задачі:

Line.hpp

#include <vector>

using namespace std;

struct Coords

{

    double x, y;

};

class Line

{

public:

    struct TAngle

    {

        double rad, deg;

    };

    Line();

    Line(int c);

    Line(int a, int b, int c);

    void operator++();

    bool operator||(Line SecondLine);

    double GetAngle();

    Coords FindIntersection(Line SecondLine);

private:

    vector<int> Coefs;

    TAngle Angle;

};

Line.cpp

#include "Line.hpp"

#include <math.h>

#include <iostream>

static double RadToDeg(double Rad)

{

    return Rad \* (180.0 / M\_PI);

}

static double DegToRad(double Deg)

{

    return Deg \* (M\_PI / 180.0);

}

Line::Line()

{

    std::cout<<"Line P1 y = x  created;\n";

    Coefs = {1, -1, 0};

    Angle = { M\_PI / 4, 45};

}

Line::Line(int c)

{

    Coefs = {0, -1, c};

    Angle = {0, 0};

}

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

{

    Coefs = {a, b, c};

    if (b != 0)

    {

        Angle.rad =atan(static\_cast<double>(-a) / b);

        Angle.deg = RadToDeg(Angle.rad);

    }

    else

    {

        Angle = { M\_PI/2, 90};

    }

}

void Line::operator++()

{

    Angle.deg += 1;

    Angle.rad += RadToDeg(1);

    Coefs[1] = tan(Angle.rad);

}

bool Line::operator||(Line SecondLine)

{

    if (this->Angle.rad == SecondLine.Angle.rad)

        return true;

    else

        return false;

}

double Line::GetAngle()

{

    return Angle.deg;

}

Coords Line::FindIntersection(Line SecondLine)

{

    Coords Intersect;

    int divider, multiplier;

    double a[2][3];

    if (this->Coefs[0] != 0)

    {

        for (size\_t i = 0; i < 3; i++)

        {

            a[0][i] = this->Coefs[i];

            if (i == 3)

                a[0][i] \*= -1;

        }

        for (size\_t i = 0; i < 3; i++)

        {

            a[1][i] = SecondLine.Coefs[i];

            if (i == 3)

                a[1][i] \*= -1;

        }

    }

    else

    {

        for (size\_t i = 0; i < 3; i++)

        {

            a[1][i] = this->Coefs[i];

            if (i == 3)

                a[1][i] \*= -1;

        }

        for (size\_t i = 0; i < 3; i++)

        {

            a[0][i] = SecondLine.Coefs[i];

            if (i == 3)

                a[0][i] \*= -1;

        }

    }

    double a\_clone[2][3];

    for (size\_t i = 0; i < 2; i++)

    {

        for (size\_t j = 0; j < 3; j++)

        {

            a\_clone[i][j] = a[i][j];

        }

    }

    for (size\_t i = 0; i < 3; i++)

        a\_clone[0][i] /= a[0][0];

    for (size\_t i = 0; i < 3; i++)

        a\_clone[1][i] -= a[1][0] \* a\_clone[0][i];

    Intersect.y = -1 \* a\_clone[1][2] / a\_clone[1][1];

    Intersect.x = a\_clone[0][2] - Intersect.y \* a\_clone[0][1];

    return Intersect;

}

main.cpp

#include "Line.hpp"

#include <iostream>

using namespace std;

int main(int argc, char const \*argv[])

{

    int a, b, c;

    Line P1;

    cout << "Enter C coef to create the P2 line parallel to the X axis: \n";

    cin >> c;

    Line P2(c);

    cout << "Enter 3 coef to create the P3 line: \n";

    cin >> a >> b >> c;

    Line P3(a, b, c);

    if (P1 || P2)

        cout << "Lines P1 and P2 are parallel.\n";

    else

        cout << "Lines P1 and P2 aren't parallel.\n";

    if (!(P2 || P3))

        {

            Coords answer = P2.FindIntersection(P3);

            cout << "Point of intersection P2 and P3 is: " << answer.x << ' ' << answer.y <<"\n\n";

        }

        else

            cout << "Point of intersection P2 and P3 doesn't exist\n\n";

    cout << "P3 angle is: " << P3.GetAngle() << '\n';

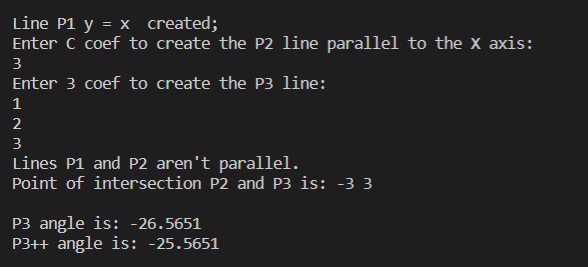
    ++P3;

    cout << "P3++ angle is: " << P3.GetAngle() << '\n\n';

    return 0;

}

Скріншоти виконання програми:



*Висновок*

В ході лабораторної роботи я вивчив механізми створення класів з використанням перевантажених операторів.