

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

# ЛАБОРАТОРНАЯ РАБОТА №1

по курсу  
объектно-ориентированное программирование I семестр, 2021/22 уч. год

Студент Меджидли Махмуд Ибрагим оглы, группа М80-208Б-20

Преподаватель Дорохов Евгений Павлович

## **Цель:**

- Изучение системы сборки на языке C++, изучение систем контроля версии.
- Изучение основ работы с классами в C++;

## **Порядок выполнения работы**

1. Ознакомиться с теоретическим материалом.
2. Получить у преподавателя вариант задания.
3. Реализовать задание своего варианта в соответствии с поставленными требованиями.
4. Подготовить тестовые наборы данных.
5. Создать репозиторий на GitHub.
6. Отправить файлы лабораторной работы в репозиторий.
7. Отчитаться по выполненной работе путём демонстрации работающей программы на тестовых наборах данных (как подготовленных самостоятельно, так и предложенных преподавателем) и ответов на вопросы преподавателя (как из числа контрольных, так и по реализации программы).

## **Требования к программе**

Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.

Необходимо настроить сборку лабораторной работы с помощью CMake. Собранная программа должна называться **oop\_exercise\_01** (в случае использования Windows **oop\_exercise\_01.exe**)

Необходимо зарегистрироваться на GitHub (если студент уже имеет регистрацию на GitHub то можно использовать ее) и создать репозиторий для задания лабораторной работы.

Преподавателю необходимо предъявить ссылку на публичный репозиторий на Github. Имя репозитория должно быть [https://github.com/login/oop\\_exercise\\_01](https://github.com/login/oop_exercise_01)

Где login – логин, выбранный студентом для своего репозитория на Github.

Репозиторий должен содержать файлы:

- main.cpp // файл с заданием работы
- CMakeLists.txt // файл с конфигурацией CMake
- test\_xx.txt // файл с тестовыми данными. Где xx – номер тестового набора 01, 02 , ... Тестовых наборов должно быть больше 1.
- report.doc // отчет о лабораторной работе

## Описание программы

Исходный код лежит в 3 файлах:

1. main.cpp - исполняемый код.
2. Long.h - специальный файл .h.
3. CMakeLists.txt - специальный дополнительный файл типа CMakeLists.

Пользователь вводя символы 1-9 и =, вызывает соответствующую каждой задаче функцию, которая выполняет требуемое.

1 – В в о д д а н н ы х

2 – В ы в о д

3 – П е р е м е щ е н и е

4 – И з м е н е н и е р а з м е р а

5 – М и н и м а л ь н ы й п р я м о у г о л ь н и к

6 – П р я м о у г о л ь н и к п о л у ч и в ш и й с я н а п е р  
е с е ч е н и и

7 – В ы в о д п л о щ а д и

8 – В ы в о д п е р и м е т р а

9 – С р а в н е н и е п л о щ а д и

= – С р а в н е н и е п е р и м е т р о в

0 – В ы х о д

**Дневник отладки**

Во время выполнения лабораторной работы программа не нуждалась в отладке, все ошибки компиляции были исправлены с первой попытки. После их исправления программа работала так, как было задумано изначально.

## **Недочёты**

Недочётов не было обнаружено.

## **Выводы**

Данная лабораторная работа помогла мне использовать полученные на лекциях теоретические знания на практике, и я написал простенький полностью работающий класс.

## Исходный код

### Long.h

```
#pragma once
```

```
#include <iostream>
```

### Main.cpp

```
// CMakeProject1.cpp : Defines the entry point for the application.
```

```
//
```

```
#include "CMakeProject1.h"
```

```
#include <iostream>
```

```
#include <Windows.h>
```

```
#include <algorithm>
```

```
using namespace std;
```

```
class rectangle
```

```
{
```

```
private:
```

```
    double x1, x2, y1, y2, dx, dy;
```

public:

rectangle(double a, double b, double c, double d) : x1(a), x2(b), y1(c), y2(d) {} // конструктор  
с параметрами

rectangle() : x1(2), x2(5), y1(2), y2(0) {} // конструктор по умолчанию

~rectangle() {} // деструктор

void Input();

void Result();

void Peremeshenie();

void Size();

void Minimal();

void Cross();

void Square();

void Perimeter();

void SquareComparison();

void PerimeterComparison();

};

void rectangle::Square()

{

double A = abs(x2 - x1); // длина стороны A

double B = abs(y2 - y1); // длина стороны B

cout << "П л о щ а д ь = " << (A \* B) << endl; // П л о щ а д ь

};

void rectangle::Perimeter()

{

double A = abs(x2 - x1); // длина стороны A

double B = abs(y2 - y1); // длина стороны B

cout << "П е р и м е т р = " << 2 \* (A + B) << endl; // П е р и м е т р

};

void rectangle::SquareComparison()

{

```

double x3, x4, y3, y4, S1, S2;

cout << "2 прямоугольник:" << endl;
cout << "Введите координату левой точки" << endl; cin >> x3;
cout << "Введите координату верхней точки" << endl; cin >> y3;
cout << "Введите координату правой точки" << endl; cin >> x4;
cout << "Введите координату нижней точки" << endl; cin >> y4;


double A = abs(x2 - x1); // длина стороны А
double B = abs(y2 - y1); // длина стороны В
S1 = A * B;
cout << "Площадь 1-го =" << S1 << endl; //Площадь 1-го


double Q = abs(x4 - x3); //длина стороны А
double W = abs(y4 - y3); //длина стороны В
S2 = Q * W;
cout << "Площадь 2-го =" << S2 << endl; //Площадь 2-го


if (S1 == S2)
{
    cout << "Площадь 1-го = Площадь 2-го " << endl;
}
else if (S1 > S2)
{
    cout << "Площадь 1-го > Площадь 2-го " << endl;
}
else if (S1 < S2)
{
    cout << "Площадь 1-го < Площадь 2-го " << endl;
}
}

```



```

void rectangle::PerimeterComparison()
{
    double x3, x4, y3, y4, S1, S2;

    cout << "2 прямоугольник:" << endl;
    cout << "Введите координату х левой точки" << endl; cin >> x3;
    cout << "Введите координату у верхней точки" << endl; cin >> y3;
    cout << "Введите координату х правой точки" << endl; cin >> x4;
    cout << "Введите координату у нижней точки" << endl; cin >> y4;

    double A = abs(x2 - x1); //длина стороны А
    double B = abs(y2 - y1); //длина стороны В
    S1 = 2 * (A + B); //Периметр
    cout << "Периметр 1-го =" << S1 << endl; //Периметр

    double Q = abs(x4 - x3); //длина стороны А
    double W = abs(y4 - y3); //длина стороны В
    S2 = 2 * (Q + W); //Периметр
    cout << "Периметр 2-го =" << S2 << endl; //Периметр

    if (S1 == S2)
    {
        cout << "Периметр 1-го = Периметр 2-го " << endl;
    }
    else if (S1 > S2)
    {
        cout << "Периметр 1-го > Периметр 2-го " << endl;
    }
    else if (S1 < S2)
    {
        cout << "Периметр 1-го < Периметр 2-го " << endl;
    }
}

```

```
}
```

```
void rectangle::Input()
```

```
{
```

```
    cout << "Введите координату х левой нижней точки" << endl;  
    cin >> x1;
```

```
    cout << "Введите координату у левой нижней точки" << endl; cin  
>> y2;
```

```
    cout << "Введите координату х правой верхней точки" <<  
endl; cin >> x2;
```

```
    cout << "Введите координату у правой верхней точки" <<  
endl; cin >> y1;
```

```
}
```

```
void rectangle::Result()
```

```
{
```

```
    cout << " A:" << "(" << x1 << "," << y1 << ")" << endl;
```

```
    cout << " B:" << "(" << x2 << "," << y1 << ")" << endl;
```

```
    cout << " C:" << "(" << x2 << "," << y2 << ")" << endl;
```

```
    cout << " D:" << "(" << x1 << "," << y2 << ")" << endl;
```

```
}
```

```
void rectangle::Peremeshenie()
```

```
{
```

```
    cout << "На сколько переместить по X?" << endl;
```

```
    cin >> dx;
```

```
    x1 += dx; x2 += dx;
```

```
    cout << "На сколько переместить по Y?" << endl;
```

```
    cin >> dy;
```

```
    y1 += dy; y2 += dy;
```

```
    cout << " A:" << "(" << x1 << "," << y1 << ")" << endl;
```

```
    cout << " B:" << "(" << x2 << "," << y1 << ")" << endl;
```

```
    cout << " C:" << "(" << x2 << "," << y2 << ")" << endl;
```

```

        cout << " D:" << "(" << x1 << "," << y2 << ")" << endl;
    }

```

```

void rectangle::Size()
{
    cout << "Н а с к о л ь к о  и з м е н и т ь  X?" << endl;
    cin >> dx;
    cout << "Н а с к о л ь к о  и з м е н и т ь  Y?" << endl;
    cin >> dy;
    x2 += dx; y2 += dy;
    Result();
}

```

```

void rectangle::Minimal()
{
    double x3, x4, y3, y4, x_min = x1, x_max = x1, y_min = y1, y_max = y1, AB, BC, AC;
    cout << " 2 п р я м о у г о л ь н и к : " << endl;
    cout << "В в е д и т е  к о о р д и н а т у  х  л е в о й  т о ч к и " << endl; cin >> x3;
    cout << "В в е д и т е  к о о р д и н а т у  у в е р х н е й  т о ч к и " << endl; cin >> y3;
    cout << "В в е д и т е  к о о р д и н а т у  х  п р а в о й  т о ч к и " << endl; cin >> x4;
    cout << "В в е д и т е  к о о р д и н а т у  у н и ж н е й  т о ч к и " << endl; cin >> y4;
    cout << endl << "В ы  в в е л и  с л е д у ю щ и е  к о о р д и н а т ы  1  п р я м о у г о л ь н и к а : " << endl;
    cout << " A:" << "(" << x1 << "," << y1 << ")" << endl;
    cout << " B:" << "(" << x2 << "," << y1 << ")" << endl;
    cout << " C:" << "(" << x2 << "," << y2 << ")" << endl;
    cout << " D:" << "(" << x1 << "," << y2 << ")" << endl;
    cout << endl;
    cout << "В ы  в в е л и  с л е д у ю щ и е  к о о р д и н а т ы  2  п р я м о у г о л ь н и к а : " << endl;
    cout << " A:" << "(" << x3 << "," << y3 << ")" << endl;
    cout << " B:" << "(" << x4 << "," << y3 << ")" << endl;
    cout << " C:" << "(" << x4 << "," << y4 << ")" << endl;
}

```

```

cout << " D:" << "(" << x3 << "," << y4 << ")" << endl;

cout << endl;

double A[4] = { x1,x2,x3,x4 }, B[4] = { y1,y2,y3,y4 };

for (int i = 0; i < 4; i++)
{
    if (x_max < A[i])
    {
        x_max = A[i];
    }

    if (y_max < B[i])
    {
        y_max = B[i];
    }

    if (x_min > A[i])
    {
        x_min = A[i];
    }

    if (y_min > B[i])
    {
        y_min = B[i];
    }
}

cout << " На и м е н ь ш и й п р я м о у г о л ь н и к с о д е р ж а щ и й о б а п р е
д ы д у щ и х и м е е т с л е д у ю щ и е к о о р д и н а т ы: " << endl;

cout << " A:" << "(" << x_min << "," << y_max << ")" << endl;

cout << " B:" << "(" << x_max << "," << y_max << ")" << endl;

cout << " C:" << "(" << x_max << "," << y_min << ")" << endl;

cout << " D:" << "(" << x_min << "," << y_min << ")" << endl;
}

```

```

void rectangle::Cross()
{

```

```
double x3, x4, y3, y4, x_min = x1, x_max = x1, y_min = y1, y_max = y1, x1_sred = 0, x2_sred = 0, y1_sred = 0, y2_sred = 0;
```

```
cout << "2 прямоугольник:" << endl;
```

```
cout << "Введите координату левой точки" << endl; cin >> x3;
```

```
cout << "Введите координату верхней точки" << endl; cin >> y3;
```

```
cout << "Введите координату правой точки" << endl; cin >> x4;
```

```
cout << "Введите координату нижней точки" << endl; cin >> y4;
```

```
cout << endl;
```

```
cout << "Вы ввели следующие координаты 1 прямоугольника:" << endl;
```

```
cout << " A:" << "(" << x1 << "," << y1 << ")" << endl;
```

```
cout << " B:" << "(" << x2 << "," << y1 << ")" << endl;
```

```
cout << " C:" << "(" << x2 << "," << y2 << ")" << endl;
```

```
cout << " D:" << "(" << x1 << "," << y2 << ")" << endl;
```

```
cout << endl;
```

```
cout << "Вы ввели следующие координаты 2 прямоугольника:" << endl;
```

```
cout << " A:" << "(" << x3 << "," << y3 << ")" << endl;
```

```
cout << " B:" << "(" << x4 << "," << y3 << ")" << endl;
```

```
cout << " C:" << "(" << x4 << "," << y4 << ")" << endl;
```

```
cout << " D:" << "(" << x3 << "," << y4 << ")" << endl;
```

```
cout << endl;
```

```
double X[4] = { x1,x2,x3,x4 }, Y[4] = { y1,y2,y3,y4 };
```

```
int n = sizeof(X) / sizeof(X[0]);
```

```
int m = sizeof(Y) / sizeof(Y[0]);
```

```
/*Here we take two parameters, the beginning of the  
array and the length n upto which we want the array to  
be sorted*/
```

```
sort(X, X + n);
```

```
sort(Y, Y + n);
```

```
if (X[1] == X[2] || Y[1] == Y[2])
```

```
{
```

```

        cout << "Пересечения нет!!!:" << endl;
    }
    else
    {
        cout << "Координаты прямоугольника получившегося на пересечении:" << endl;
        cout << " A:" << "(" << X[1] << "," << Y[1] << ")" << endl;
        cout << " B:" << "(" << X[2] << "," << Y[1] << ")" << endl;
        cout << " C:" << "(" << X[2] << "," << Y[2] << ")" << endl;
        cout << " D:" << "(" << X[1] << "," << Y[2] << ")" << endl;
    }
}

```

```

int main()
{
    setlocale(LC_ALL, "Russian");

    cout << " 1 – Ввод данных /n 2 – Вывод ¥n 3 – Перемещение /n 4 – Изменение размера /n"
    " 5 – Минимальный прямоугольник /n 6 – Прямоугольник получившийся на пересечении /n 7 – Вывод площади /n 8 – Вывод периметра /n 9 – Сравнение площади /n = – Сравнение периметров /n 0 – Выход /n" << endl;

    rectangle A;

    char Control;

    cout << "> ";

    cin >> Control;

    while (Control != '0')
    {
        switch (Control)
        {
            case '1':
                A.Input();

                cout << "> ";

```

```
    cin >> Control;  
  
    break;  
  
case '2':  
  
    A.Result();  
  
    cout << "> ";  
  
    cin >> Control;  
  
    break;  
  
case '3':  
  
    A.Peremeshenie();  
  
    cout << "> ";  
  
    cin >> Control;  
  
    break;  
  
case '4':  
  
    A.Size();  
  
    cout << "> ";  
  
    cin >> Control;  
  
    break;  
  
case '5':  
  
    A.Minimal();  
  
    cout << ">";  
  
    cin >> Control;  
  
    break;  
  
case '6':  
  
    A.Cross();  
  
    cout << "> ";  
  
    cin >> Control;  
  
    break;  
  
case '7':  
  
    A.Square();  
  
    cout << "> ";  
  
    cin >> Control;
```

```

        break;
    case '8':
        A.Perimeter();
        cout << "> ";
        cin >> Control;
        break;
    case '9':
        A.SquareComparison();
        cout << "> ";
        cin >> Control;
        break;
    case '=':
        A.PerimeterComparison();
        cout << "> ";
        cin >> Control;
        break;
    case '0':
        cout << "Control = " << Control << endl;
        break;
}

return 0;
}

```