МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСТИТЕТ)

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 С++;

## 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 GitHubто можно использовтаь ее) и создать репозитарий для задания лабораторной работы.

Преподавателю необходимо предъявить ссылку на публичный репозиторий на Github. Имя репозитория должно быть [https://github.com/](https://github.com/login/oop_exercise_01)*[login](https://github.com/login/oop_exercise_01)*[/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 конфигураций 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 - Ввод данных

2 - Вывод

3 - Перемещение

4 - Изменение размера

5 - Минимальный прямоугольник

6 - Прямоугольник получившийся на пересечении

7 - Вывод площади

8 - Вывод периметра

9 - Сравнение площади

= - Сравнение периметров

0 - Выход

**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 Сross();

void Square();

void Perimeter();

void SquareComparison();

void PerimeterComparison();

};

void rectangle::Square()

{

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

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

cout << "Площадь = " << (A \* B) << endl; //Площадь

};

void rectangle::Perimeter()

{

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

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

cout << "Периметр = " << 2 \* (A + B) << endl; //Периметр

};

void rectangle::SquareComparison()

{

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

cout << " 2 прямоугольник:" << endl;

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

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

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

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

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

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

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 B

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 y верхней точки" << endl; cin >> y3;

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

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

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

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

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 B

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 y левой нижней точки" << endl; cin >> y2;

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

cout << "Введите координату y 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 << "На сколько изменить Х?" << endl;

cin >> dx;

cout << "На сколько изменить 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 прямоугольник:" << endl;

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

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

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

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

cout << endl << "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 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 << " Наименьший прямоугольник содержащий оба предыдущих имеет следующие координаты: " << 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::Сross()

{

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 х левой точки" << endl; cin >> x3;

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

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

cout << "Введите координату y 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.Сross();

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;

}