# **Лабораторная работа № 02**

**Студент**: Сазонов Вадим Кириллович , м8О-201Б-20

# **Тема: Перегрузка операций. Литералы**

**Цель:**

* Изучение основ работы с классами в С++;
* Перегрузка операций и создание литералов

## **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.

Реализовать над объектами реализовать в виде перегрузки операторов.

Реализовать пользовательский литерал для работы с константами объектов созданного класса.

**Вариант 12:**

**Разработать класс Rectangle**, представляющий собой прямоугольник со сторонами, параллельными осям координат. Поля – координаты левого нижнего и правого верхнего угла. Требуется реализовать следующие методы: вычисление площади и периметра, перемещения вдоль осей, изменение размеров, сравнение по площади и по периметру. Реализовать метод получения прямоугольника, представляющего общую часть (пересечение) двух прямоугольников. Реализовать метод объединения двух прямоугольников: наименьший прямоугольник, включающего оба заданных прямоугольника.

Код программы:

#include <iostream>

class Rectangle

{

private:

int x1, y1, x2, y2;

public:

Rectangle():x1(0),x2(0),y1(0),y2(0){};

~Rectangle(){};

Rectangle(int a,int b,int c,int d):x1(a),x2(c),y1(b),y2(d){};

int area() const{

return (x2-x1)\*(y2-y1);

}

int perimeter() const{

return 2\*(x2-x1)+2\*(y2-y1);

}

void change\_size(int a){

x2\*=a;

y2\*=a;

}

void change\_x(int a){

x1-=a;

x2-=a;

}

void change\_y(int a){

y1-=a;

y2-=a;

}

void change\_xy(int a,int b){

x1-=a;

x2-=a;

y1-=b;

y2-=b;

}

friend bool area\_compare(const Rectangle &a,const Rectangle &b){

return a.area()>b.area();

}

friend bool perimeter\_compare(const Rectangle &a,const Rectangle &b){

return a.perimeter()>b.perimeter();

}

friend Rectangle intersection(const Rectangle &a,const Rectangle&b){

Rectangle res;

if(a.x1<b.x1) res.x1=b.x1;

else res.x1=a.x1;

if(a.y1<b.y1) res.y1=b.y1;

else res.y1=a.y1;

if(a.x2>b.x2) res.x2=b.x2;

else res.x2=a.x2;

if(a.y2>b.y2) res.y2=b.y2;

else res.y2=a.y2;

return res;

}

friend Rectangle rec\_union(const Rectangle &a,const Rectangle &b){

Rectangle res;

if(a.x1<b.x1) res.x1=a.x1;

else res.x1=b.x1;

if(a.y1<b.y1) res.y1=a.y1;

else res.y1=b.y1;

if(a.x2>b.x2) res.x2=a.x2;

else res.x2=b.x2;

if(a.y2>b.y2) res.y2=a.y2;

else res.y2=b.y2;

return res;

}

friend std::ostream &operator<<(std::ostream &out,const Rectangle &a){

out<<"( "<<a.x1<<" , "<<a.y1<<" )"<<", ( "<<a.x2<<" , "<<a.y2<<" )\n";

return out;

}

};

int main(){

int a,b,c,d;

std::cin>>a>>b>>c>>d;

Rectangle first(a,b,c,d);

std::cin>>a>>b>>c>>d;

Rectangle second(a,b,c,d);

std::cout<<"First " <<first<<"Area is: "<<first.area()<<" Perimeter is: "<<first.perimeter()<<std::endl;

std::cout<<"Second "<<second<<"Area is: "<<second.area()<<" Perimeter is: "<<second.perimeter()<<std::endl;

Rectangle res=rec\_union(first,second);

std::cout<<"Union is: "<<res;

res=intersection(first,second);

std::cout<<"Intersection is: "<<res;

return 0;

}

Тесты:

1. 0 0 100 100

12 12 37 84

2. -1 -1 2 2

0 0 0 0

Результаты тестов:

1.

First ( 0 , 0 ), ( 100 , 100 )

Area is: 10000 Perimeter is: 400

Second ( 12 , 12 ), ( 37 , 84 )

Area is: 1800 Perimeter is: 194

Union is: ( 0 , 0 ), ( 100 , 100 )

Intersection is: ( 12 , 12 ), ( 37 , 84 )

2.

First ( -1 , -1 ), ( 2 , 2 )

Area is: 9 Perimeter is: 12

Second ( 0 , 0 ), ( 0 , 0 )

Area is: 0 Perimeter is: 0

Union is: ( -1 , -1 ), ( 2 , 2 )

Intersection is: ( 0 , 0 ), ( 0 , 0 )