

ФГАОУВО «Южно-Уральский государственный университет (НИУ)»

Институт естественных и точных наук

Кафедра «Прикладная математика и программирование»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

по дисциплине «Объектно-ориентированное программирование»

Автор работы

студент группы ЕТ-212

_____ Гайнулина Э.Д.

_____ 2022 г.

Работа зачтена с оценкой

_____ А.К.Демидов

_____ 2022 г.

Челябинск, 2017

1 Постановка задачи

I. Реализовать класс

5. Прямоугольник [l,t,r,b] Rect

координаты целые

Конструктор: Rect(l,t,r,b)

Операции:

x*y (пересечение, функция),
x+y ("объединение", охватывающий прямоугольник минимального размера, функция),
x*=y (заменить пересечением)
x+=y (расширить x, заменив охватывающим)
x-=y, x!=y,
x<=y (x полностью лежит в y), x>=y (наоборот, y в x),
!x (x является результатом пересечения прямоугольников, не имеющих общих точек)

где x,y - прямоугольники

вывод, ввод в виде [0,5,40,20]

Методы:

```
int getLeft();  
int getTop();  
int getRight();  
int getBottom();
```

Операции (если есть в задании) =, [], +=, -=, *=, /=, префиксные ++, -- определять как методы.

Для ввода переопределить >>, для вывода - <<. Формат ввода-вывода объектов делать так, как указано в задании.

Запись [текст] означает, что текст может отсутствовать, например, конструктор ИмяКласса(a[,b[,c]]) может быть вызван с 1, 2 или 3 аргументами.

Пример:

```
Vector a(0,0),b(1.5,0.3);  
cout<<"Введите вектор a:"<<endl;  
cin>>a;  
//нужно ввести (1.2,3.2) вместе со скобками и запятой  
cout<<a<<" + "<<b<<" = "<<(a+b)<<endl;  
// печатается (1.2,3.2) + (1.5,0.3) = (2.7,3.5)
```

II. Реализовать main с тестами (создание объектов и выполнение действий с ними)

2 Описание интерфейса класса

```
class Rect {
    long l, t, // левый верхний угол
           r, b; // правый нижний угол
    bool real;
public:
    Rect(long l=0, long t=0, long r=0, long b=0): l(l), t(t),
r(r), b(b), real(true){} //конструктор
    Rect &operator+=(const Rect&); //перегрузка операции +=
    Rect &operator*=(const Rect&); //перегрузка операции *=
    friend bool operator==(const Rect &, const Rect
&); //перегрузка операции ==
    friend bool operator!=(const Rect &, const Rect
&); //перегрузка операции !=
    friend bool operator<=(const Rect &, const Rect
&); //перегрузка операции <=
    friend bool operator>=(const Rect &, const Rect
&); //перегрузка операции >=
    bool operator!() const {return (!real);} //перегрузка
операции !
    friend istream &operator>>(istream &s, Rect
&r); //перегрузка операции >>
    friend ostream &operator<<(ostream &s, const Rect &r); //
перегрузка операции <<
    long getLeft() const { return l; } // правый x
    long getTop() const { return t; } // верхний y
    long getRight() const { return r; } // левый x
    long getBottom() const { return b; } // нижний y
};
```

3 Описание тестов для проверки классов

```
int main()

{
    Rect r1(0, 0, 10, 10), r2(3, 6, 15, 15);

    cout<<"Вывод\n";

    cout<<"r1="<<r1<<" "<<"r2="<<r2<<endl;

    cout<<"Ввод r1\n";

    cin>>r1;

    cout<<"r1="<<r1 <<endl;


    cout<<"Проверка операции +\n";

    cout<<r1<<" + "<<r2<<" = "<<(r1+r2)<<endl;


    cout<<"Проверка операции *\n";

    cout<<r1<<" * "<<r2<<" = "<<(r1*r2)<<endl;


    cout<<"Проверка операции +=\n";

    r1+=r2;

    cout<<"r1 += r2, r1 ="<<r1<<endl;


    cout<<"Проверка операции *=\n";

    r1*=r2;

    cout<<"r1 *= r2, r1 ="<<r1<<endl;


    cout<<"Проверка операции ==\n";

    cout<<r1<<" == "<<r2<<" = "<<(r1==r2)<<endl;


    cout<<"Проверка операции !=\n";

    cout<<r1<<" != "<<r2<<" = "<<(r1!=r2)<<endl;
```

```
cout<<"Проверка операции !()\n";  
cout<<"! ("<<r1<<' ' '<<(! (r1))<<endl;  
return 0;  
}
```

Полученные результаты

Вывод

$r1=[0,0,10,10]$ $r2=[3,6,15,15]$

Ввод $r1$

$[2,7,8,9]$

$r1=[2,7,8,9]$

Проверка операции $+$

$[2,7,8,9] + [3,6,15,15] = [2,6,15,15]$

Проверка операции $*$

$[2,7,8,9] * [3,6,15,15] = [3,7,8,9]$

Проверка операции $+=$

$r1 += r2, r1 = [2,6,15,15]$

Проверка операции $*=$

$r1 *= r2, r1 = [3,6,15,15]$

Проверка операции $==$

$[3,6,15,15] == [3,6,15,15] = 1$

Проверка операции $!=$

$[3,6,15,15] != [3,6,15,15] = 0$

Проверка операции $!()$

$!([3,6,15,15]) 0$

4 Листинг реализации класса

```
Rect& Rect::operator+=(const Rect& rect) {
    if(!real || !rect.real) return *this;
    l = fmin(l, rect.l);
    t = fmin(t, rect.t);
    r = fmax(r, rect.r);
    b = fmax(b, rect.b);
    return *this;
}

Rect operator+(Rect r1, Rect r2) {
    return r1 += r2;
}

Rect &Rect::operator*=(const Rect&rect) {
    if(!real || !rect.real) return *this;
    l = fmax(l, rect.l);
    r = fmin(r, rect.r);
    t = fmax(t, rect.t);
    b = fmin(b, rect.b);
    if(r-l <= 0 || b-t <= 0)
    {
        real = false;
        r = 0;
        t = 0;
        l = 0;
        b = 0;
    }
    return *this;
}

Rect operator*(Rect r1, Rect r2) {
    return r1 *= r2;
}

bool operator==(const Rect &r1, const Rect &r2) {
    return (r1.r == r2.r and r1.t == r2.t and r1.l == r2.l and
    r1.b == r2.b and r1.real == r2.real);
}

bool operator!=(const Rect &r1, const Rect &r2) {
    return !(r1==r2);
}

bool operator<=(const Rect &x, const Rect &y) {
    return (x.l >= y.l and x.t >= y.t and x.r <= y.r and x.b <=
    y.b and x.real);
}

bool operator>=(const Rect &x, const Rect &y) {
    return (y<=x);
}
```

```
istream &operator>>(istream &s, Rect &r) {
    char c[5];
    return s>>c[0]>>r.l>>c[1]>>r.t>>c[2]>>r.r>>c[3]>>r.b>>c[4];
}

ostream &operator<<(ostream &s, const Rect &r) {
    return s<<'['<<r.l <<","<<r.t<<','<<r.r<<','<<r.b<<']';
}
```