

# Методические указания для СРС по выполнению лабораторных работ по дисциплине Объектно-ориентированное программирование

Тема: **Перегрузка операций**

1. Реализовать класс с заданным интерфейсом на языке C++ согласно варианту.

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)
```

2. Реализовать main с тестами для проверки класса (создание объекта и выполнение действий с ним)

3. Написать отчет

- Постановка задачи
- Описание интерфейса класса (class {} и комментарии ко всем полям, методам и функциям)
- Описание тестов для проверки классов (main с комментариями, какие действия выполнялись, полученные результаты)
- Листинг реализации класса (реализация методов и функций, отступы, без комментариев)

Шаблон отчета и пример заполнения отчета в приложенных файлах

4. Отправить отчет

## Пример решения задания 2

Класс день недели

Операции:

конструктор(номер дня 1..7), метод day() возвращает номер дня

префиксный и постфиксный ++ и --

операции weekday+=int, weekday-=int, weekday+int, weekday-int

операции == и !=, где оба аргумента weekday, либо int и weekday

ввод-вывод как две буквы ПН, ВТ, ..., ВС

```
class weekday {
    int d; // номер дня недели от 1 до 7
```

```

    static char *names[7]; // обозначения дней недели для ввода-вывода
public:
    // методы
    weekday(int); // конструктор-преобразователь
    int day() const { return d; } // возвращает номер дня
    weekday &operator++(); // префиксный ++
    weekday &operator--(); // префиксный --
    weekday &operator+=(int n); // операция += (день недели через n дней)
    friend bool operator==(weekday w1, weekday w2) {return w1.d==w2.d; } //
сравнение на равенство
    friend istream& operator>>(istream&, weekday&); // ввод
    friend ostream& operator<<(ostream&, weekday); // вывод
};
// другие операции (не методы)
inline weekday operator++(weekday&w, int) { // постфиксная операция ++
    weekday t(w);
    ++w;
    return t;
}
inline weekday operator--(weekday&w, int) { // постфиксная операция --
    weekday t(w);
    --w;
    return t;
}
inline weekday &operator--(weekday&w, int n) { // операция -= (день недели n
дней назад)
    return w+=-n;
}
inline weekday operator+(weekday w, int n) { // операция + (день недели через n
дней)
    return w+=n;
}
inline weekday operator-(weekday w, int n) { // операция - (день недели n дней
назад)
    return w-=n;
}
inline bool operator!=(weekday w1, weekday w2) { // неравенство
    return !(w1==w2);
}
// реализация
weekday::weekday(int n):d(n) {
    if(n<1 || n>7) {
        cout<<"Неверный номер\n";
        exit(1);
    }
}
weekday &weekday::operator++() {
    ++d;
    if(d>7) d=1;
    return *this;
}
weekday &weekday::operator--() {
    --d;
    if(d<1) d=7;
    return *this;
}
weekday &weekday::operator+=(int n) {
    d=(d+n)%7;
    if(d<=0) d+=7;
    return *this;
}
static char *weekday::names[7]={"ПН", "ВТ", "СР", "ЧТ", "ПТ", "СБ", "ВС"};
istream& operator>>(istream&i, weekday&w) {
    char c1,c2;

```

```

i>>c1>>c2;
for(int d=0;d<7;++d)
    if(weekday::names[d][0]==c1 && weekday::names[d][1]==c2) {
        w.d=d+1;
        break;
    }
return i;
}
ostream& operator<<(ostream&o, weekday w) {
    return o<<weekday::names[w.d-1];
}

```

Пример проверки, задача - вызвать все операции

```

int main() {
    weekday w1(1),w2(1);
    ++w1;
    cout<<w1<<"\n";
    w1--;
    cout<<w1<<"\n";
    cout<<"Введите день недели:"
    cin>>w1;
    cout<<w1<<" == "<<w2<<" = "<<(w1==w2)<<"\n";
    cout<<w1<<" != "<<w2<<" = "<<(w1!=w2)<<"\n";
    cout<<w1<<" == "<<2<<" = "<<(w1==2)<<"\n";
    cout<<1<<" != "<<w1<<" = "<<(1!=w1)<<"\n";
    cout<<w1<<" + "<<100<<" = "<<(w1+100)<<"\n";
    cout<<w1<<" - "<<100<<" = "<<(w1-100)<<"\n";
}

```

## Варианты заданий

### Вариант 1

Вектор на плоскости (x,y) Vector

Конструктор: Vector(x,y)

Операции:

$a+b$ ,  $a-b$ ,

$a+=b$ ,  $a-=b$ ,

$a*b$  (скалярное произведение),

$a==b$ ,  $a!=b$

!a (a является нулевым вектором),

$a/z$ ,  $z*a$  ("масштабирование" вектора)

где a,b - вектора, z - double

вывод, ввод в виде (1.4, -2.5)

Методы:

double getX();

double getY();

### Вариант 2

Комплексное число  $a+bi$  Complex

Конструктор: Complex(a,b)

Операции:

$x+y$ ,  $x-y$ ,  $x*y$ ,  $x/y$ ,

$x+=y$ ,  $x-=y$ ,  $x*=y$ ,  $x/=y$ ,

$x==y$ ,  $x!=y$ ,

!x (x равно 0)

где x,y - комплексные числа или double

(определить преобразование double->Complex)

вывод, ввод в виде 1.5+2.2i

Методы:

double getReal(); (реальная часть)

double getImaginary(); (мнимая часть)

Остальные варианты <https://ipc.susu.ru/20768-3.html>

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  
\_\_\_\_\_ А.А.Александрова  
\_\_\_\_\_ 2019 г.

Работа зачтена с оценкой  
\_\_\_\_\_  
\_\_\_\_\_ А.К.Демидов  
\_\_\_\_\_ 2019 г.

Челябинск, 2019

# 1 Постановка задачи

## I. Реализовать класс

### 1. Рациональные числа (дроби) Rational

это числа вида  $a/b$ , где  $a$  и  $b$  не имеют общих делителей,  $b > 0$ ,  $|a|, |b| < 2^{31}$

Конструктор: Rational(a,b)

Операции:

$x+y$ ,  $x-y$ ,  $x*y$ ,  $x/y$ ,

$x+=y$ ,  $x-=y$ ,  $x*=y$ ,  $x/=y$ ,

$x>y$ ,  $x<y$ ,  $x>=y$ ,  $x<=y$ ,  $x==y$ ,  $x!=y$

!x (x равно нулю)

где  $x, y$  - рациональные числа или long

(определить преобразование long->Rational)

вывод, ввод в виде 3/5

Методы:

long getp(); (числитель)

long getq(); (знаменатель)

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.

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

## 2 Описание интерфейса класса

```
class Rational {  
    long a, // числитель  
        b; // знаменатель.  
public:  
    Rational(long=0, long=1); // конструктор  
    Rational &operator+=(const Rational&); // перегрузка операции +=  
    Rational &operator-=(const Rational&); // перегрузка операции -=  
    Rational &operator*=(const Rational&); // перегрузка операции *=  
    Rational &operator/=(const Rational&); // перегрузка операции /=  
    friend bool operator<(const Rational &, const Rational &); // перегрузка операции <  
    friend bool operator==(const Rational &, const Rational &); // перегрузка операции ==  
    bool operator!() const { return (a==0); } // перегрузка операции !  
    friend istream &operator>>(istream &, Rational &); // перегрузка операции >>  
    friend ostream &operator<<(ostream &, const Rational &); // перегрузка операции <<  
    long getp() const { return a; } // числитель  
    long getq() const { return b; } // знаменатель.  
};
```

### 3 Описание тестов для проверки классов

```
int main() {
    Rational a(1, 3), b(1,2), c(15);
    cout<<"Вывод\n";
    cout<<"a="<<a<<" "<<"b="<<b<<" "<<"c="<<c<<endl;
    cout<<"Ввод a\n";
    cin>>a;
    cout<<"a="<<a <<endl;
    cout<<"Проверка операции +\n";
    cout<<a<<" + "<<b<<" = "<<(a+b)<<endl;
    cout<<a<<" + "<<2<<" = "<<(a+2)<<endl;
    cout<<2<<" + "<<a<<" = "<<(2+a)<<endl;
    cout<<a+b<<endl;
    ...
    cout<<"Проверка операции +=\n";
    a+=b;
    cout<<"a="<<a<<endl;
    ...
    cout<<"Проверка операции >\n";
    cout<<a<<" > "<<b<<" = "<<(a>b)<<endl;
    cout<<a<<" > "<<2<<" = "<<(a>2)<<endl;
    cout<<2<<" > "<<a<<" = "<<(2>a)<<endl;
    ...
    return 0;
}
```

#### Полученные результаты.

Вывод

a=1/3 b=1/2 c=15/1

Ввод a

2/7

a=2/7

Проверка операции +

2/7 + 1/2 = 11/14

2/7 + 2 = 16/7

2 + 2/7 = 16/7

Проверка операции –

...

Проверка операции \*

...

Проверка операции /

...

Проверка операции +=

a=11/14

Проверка операции -=

a=2/7

Проверка операции >

11/14 > 1/2 = 1

11/14 > 2 = 0

2 > 11/14 = 1

Проверка операции <

...

Проверка операции >=



...

Проверка операции  $\leq$

...

Проверка операции  $=$

...

Проверка операции  $\neq$

...

## 4 Листинг реализации класса

```
long NOD(long a, long b){
    long c;
    if(a<0) a=-a;
    while(b!=0){
        c=a;
        a=b;
        b=c%b;
    }
    return a;
}

Rational::Rational(long a, long b){
    long tmp=NOD(a,b);
    this->a=a/tmp;
    this->b=b/tmp;
}

Rational &Rational::operator+=(const Rational&p) {
    a=a*p.b+p.a*b;
    b=b*p.b;
    long tmp=NOD(a,b);
    a/=tmp;
    b/=tmp;
    return *this;
}

Rational operator+(Rational x, Rational y) {
    return x+=y;
}

Rational operator/(Rational x, Rational y) {
    return x/=y;
}

bool operator<(const Rational &x, const Rational &y) {
    if ((x.a*y.b-y.a*x.b)<0) return 1;
    return 0;
}

bool operator>(const Rational &x, const Rational &y) {
    return y<x;
}

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

bool operator<=(const Rational &x, const Rational &y) {
    return !(y<x);
}

bool operator==(const Rational &x, const Rational &y) {
    return ((x.a==x.b&& y.a==y.b));
}

bool operator!=(const Rational &x, const Rational &y) {
    return !(x==y);
}
```

```
istream &operator>>(istream &s, Rational &p) {  
    char c;  
    return s>>p.a>>c>>p.b;  
}  
ostream &operator<<(ostream &s, const Rational &p) {  
    return s<<p.a <<"/"<<p.b;  
}
```