



C++

## Разбор домашней работы



# Перегрузки операций

# Теория



15 минут



**Вспомним, что было на  
предыдущем занятии**

# Дробь

```
class Fraction {
    long long x, y;
public:
    Fraction(long long x = 0, long long y = 1): x(x), y(y) {}

    void read() {
        char tmp;
        cin >> x >> tmp >> y;
    }
    void print() { cout << x << "/" << y << endl; }
    Fraction operator+(long long a) { // Сложение с целым числом
        return Fraction(x + a*y, y);
    }
};
```

**Что произойдет, если сложить  
число с дробью, а не  
наоборот?**

**Операции можно перегружать  
как внешние функции**



# Дробь

```
class Fraction {
    long long x, y;
public:
    Fraction(long long x = 0, long long y = 1): x(x), y(y) {}

    void read() {
        char tmp;
        cin >> x >> tmp >> y;
    }
    void print() { cout << x << "/" << y << endl; }
    Fraction operator+(long long a) { // Сложение с целым числом
        return Fraction(x + a*y, y);
    }
};

Fraction operator+(long long a, Fraction b) {
    return Fraction(a*b.y + b.x, b.y);
}
```

**В чем ошибка?**

# Дружественные функции

**Дружественные функции** - функции, имеющие доступ ко всем полям и методам класса, вне зависимости от модификаторов доступа.

Чтобы сделать дружественную функцию, необходимо указать прототип этой функции с ключевым словом **friend** внутри класса:

**friend** тип результата **имя функции** (параметры) { }

# Дробь

```
class Fraction {
    long long x, y;
public:
    Fraction(long long x = 0, long long y = 1): x(x), y(y) {}
    void read() {
        char tmp;
        cin >> x >> tmp >> y;
    }
    void print() { cout << x << "/" << y << endl; }
    Fraction operator+(long long a) { // Сложение с целым числом
        return Fraction(x + a*y, y);
    }
    friend Fraction operator+(long long, Fraction);
};

Fraction operator+(long long a, Fraction b) {
    return Fraction(a*b.y + b.x, b.y);
}
```

# Дробь

```
class Fraction {  
    long long x, y;  
public:  
    Fraction(long long x = 0, long long y = 1): x(x), y(y) {}  
    void read() {  
        char tmp;  
        cin >> x >> tmp >> y;  
    }  
    void print() { cout << x << "/" << y << endl; }  
    Fraction operator+(long long a) { // Сложение с целым числом  
        return Fraction(x + a*y, y);  
    }  
    friend Fraction operator+(long long, Fraction);  
};  
Fraction operator+(long long a, Fraction b) {  
    return Fraction(a*b.y + b.x, b.y);  
}
```

В реализации функции не нужно указывать ключевое слово friend!

# Практика



15 минут



# Теория



15 минут



**Можно ли перегрузить  
операции ввода и вывода?**



# Дробь

```
class Fraction {  
    long long x, y;  
public:  
    // Описание класса  
};
```

```
int main() {  
    Fraction a;  
    cin >> a;  
    cout << a;  
    return 0;  
}
```

# Дробь

```
class Fraction {  
    long long x, y;  
public:  
    // Описание класса  
};
```

```
int main() {  
    Fraction a;  
    cin >> a;  
    cout << a;  
    return 0;  
}
```

Что является левым операндом, а что правым для операции ввода?

# Дробь

```
class Fraction {  
    long long x, y;  
public:  
    // Описание класса  
};
```

```
int main() {  
    Fraction a;  
    cin >> a;  
    cout << a;  
    return 0;  
}
```

Что является левым операндом, а что правым для операции ввода?

Левый операнд - cin

cin имеет тип данных istream

# Дробь

```
class Fraction {
    long long x, y;
public:
    // Описание класса
    friend void operator>>(istream&, Fraction&);
};

void operator>>(istream& input, Fraction& a){
    char tmp;
    input >> a.x >> tmp >> a.y;
}

int main() {
    Fraction a;
    cin >> a;
    cout << a;
    return 0;
}
```

**Что произойдет, если считать  
несколько переменных?**

# Дробь

```
class Fraction {  
    long long x, y;  
public:  
    // Описание класса  
    friend void operator>>(istream&, Fraction&);  
};  
void operator>>(istream& input, Fraction& a){  
    char tmp;  
    input >> a.x >> tmp >> a.y;  
}  
int main() {  
    Fraction a, b;  
    cin >> a >> b;  
    cout << a;  
    return 0;  
}
```

**Что должно быть результатом  
ввода переменной?**

# Дробь

```
class Fraction {
    long long x, y;
public:
    // Описание класса
    friend istream& operator>>(istream&, Fraction&);
};

istream& operator>>(istream& input, Fraction& a){
    char tmp;
    input >> a.x >> tmp >> a.y;
    return input;
}

int main() {
    Fraction a, b;
    cin >> a >> b;
    cout << a;
    return 0;
}
```



# Дробь

```
class Fraction {  
    long long x, y;  
public:  
    // Описание класса  
    friend istream& operator>>(istream&, Fraction&);  
    friend ostream& operator<<(ostream&, Fraction);  
};  
istream& operator>>(istream& input, Fraction& a){  
    char tmp;  
    input >> a.x >> tmp >> a.y;  
    return input;  
}  
ostream& operator<<(ostream& output, Fraction a){  
    output << a.x << "/" << a.y;  
    return output;  
}
```

# Практика



25 минут



# Итоги урока

- 1) Операции не коммутативны
- 2) Перегрузка операций как внешних функций
- 3) Дружественные функции
- 4) Перегрузка операций ввода и вывода