



C++

Разбор домашней работы



Перегрузки операций

Теория



15 минут



Вспомним класс MyVector

Вектор

```
class MyVector {  
    int* data;  
    int len;  
public:  
    MyVector(int len = 0): len(len) {  
        data = new int[len];  
    }  
    ~MyVector() { delete[] data; }  
    int get(int i) { return data[i]; }  
    void set(int i, int x) { data[i] = x; }  
};
```

**Как обращаться к отдельным
элементам такого вектора?**

[]

Квадратные скобки перегружаются только как член класса и принимают как параметр ровно одно значение.

Как параметр может выступать значение любого типа данных, а не только целочисленное.

тип результата **operator[]** (параметр) {}

Вектор

```
class MyVector {  
    int* data;  
    int len;  
public:  
    MyVector(int len = 0): len(len) {  
        data = new int[len];  
    }  
    ~MyVector() { delete[] data; }  
    int get(int i) { return data[i]; }  
    void set(int i, int x) { data[i] = x; }  
    int operator[](int i) { return data[i]; }  
};
```

Вектор

```
class MyVector {  
    int* data;  
    int len;  
public:  
    MyVector(int len = 0): len(len) {  
        data = new int[len];  
    }  
    ~MyVector() { delete[] data; }  
    int get(int i) { return data[i]; }  
    void set(int i, int x) { data[i] = x; }  
    int operator[](int i) { return data[i]; }  
};
```

```
MyVector a(5);  
a[2] = 3;  
cout << a[2];
```

Что будет выведено на экран?

Вектор

```
class MyVector {  
    int* data;  
    int len;  
public:  
    MyVector(int len = 0): len(len) {  
        data = new int[len];  
    }  
    ~MyVector() { delete[] data; }  
    int get(int i) { return data[i]; }  
    void set(int i, int x) { data[i] = x; }  
    int& operator[](int i) { return data[i]; }  
    //Результатом должна быть ссылка  
};
```

```
MyVector a(5);  
a[2] = 3;  
cout << a[2];
```

Практика



20 минут



Теория



10 минут



()

Круглые скобки также перегружаются только как член класса, но в отличие от квадратных могут принимать любое количество параметров

тип результата **`operator()`** (параметры) `{ }`

**Зачем стоит перегружать
круглые скобки?**

Функтор

Функтор (или функциональный объект) - объект класса, который можно использовать как функцию.

Позволяет, например, сохранять состояние между запусками функции.

Пример

```
class MyVector {
    int* data;
    int len;
public:
    MyVector(int len = 0): len(len) {
        data = new int[len];
    }
    ~MyVector() { delete[] data; }
    int get(int i) { return data[i]; }
    void set(int i, int x) { data[i] = x; }
    int& operator[](int i) { return data[i]; }
    //Перегрузим скобки так, чтобы находить сумму элементов на отрезке
    int operator()(int l, int r) {
        int sum = 0;
        for (int i = l; i <= r; i++) sum += data[i];
        return sum;
    }
};
```

Практика



15 минут



Теория



5 минут



Приведение типов

Приведение типов также перегружается только как член класса, но, опять же, в отличие предыдущих перегрузок, не принимает никаких параметров и не нуждается в типе возвращаемого значения.

```
operator тип данных() {}
```

Пример

```
class Fraction{
    long long x, y;
public:
    Fraction(long long x = 0, long long y = 1): x(x), y(y) {}
    friend istream& operator>>(istream&, Fraction&);
    friend ostream& operator<<(ostream&, Fraction&);

    operator double() {
        return (double)x/y;
    }
};
```

Практика



10 минут



Итоги урока

- 1) Перегрузка индексации
- 2) Перегрузка круглых скобок
- 3) Функторы
- 4) Перегрузка приведения типов
- 5) Отличие этих перегрузок от изученных ранее