

# Объектно-ориентированное программирование

Перегрузка операций

Гришмановский Павел Валерьевич,  
кафедра автоматики и компьютерных систем, Политехнический институт, СурГУ

# Перегрузка операций

Перегрузка операций – реализация существующих в С++ операций для операндов, имеющих тип, определенный разработчиком (т.е. класс)

- Перегружаются только существующие в С++ операции
- Нельзя изменить приоритет, ассоциативность, синтаксис, количество операндов
- Не перегружаются операции: `?: sizeof :: . .*`
- Хотя бы один из операндов должен быть объектом класса (или ссылкой на объект класса)

Перегрузка любой операции @ выполняется в виде функции или метода с именем **operator@**

Любая операция @ может быть вызвана с использованием традиционного синтаксиса этой операции

1. Если все операнды имеют примитивные типы (не являются объектами классов), то вызывается встроенная операция
2. Если хотя бы один операнд – объект класса, то вызывается соответствующая перегруженная операция
3. Если подходящей перегруженной операции нет, то объект приводится к другому типу (если возможно), чтобы вызвать встроенную или перегруженную операцию

Любая перегруженная операция также может быть вызвана явно как функция или метод с именем **operator@**

# Перегрузка операций как функций

Перегрузка операции как функции – количество параметров соответствует количеству операндов:

`<Т.В.З.> operator@ (<ТИП> <ИМЯ>) ;` унарная

`<Т.В.З.> operator@ (<ТИП1> <ИМЯ1>, <ТИП2> <ИМЯ2>) ;` бинарная

Хотя бы один из операндов должен быть объектом класса или ссылкой на объект класса

Вызов операции:

`@ <оп1>` `operator@ (<оп1>)` унарная

`<оп1> @ <оп2>` `operator@ (<оп1>, <оп2>)` бинарная

Нельзя перегружать как функции:

`[]` `()` `(<ТИП>)` `->`

`& (адрес)` `* (разыменование)`

# Перегрузка операций как методов

Перегрузка операции как метода – количество параметров на 1 меньше, чем количество операндов:

`<Т.В.З.> operator@ ( ) ;` унарная

`<Т.В.З.> operator@ ( <ТИП> <ИМЯ> ) ;` бинарная

Левый или единственный операнд должен быть объектом класса или ссылкой на объект класса

Вызов операции извне объекта (`<оп1>` – объект класса):

`@ <оп1>` `<оп1>.operator@ ( )` унарная  
`<указатель> -> operator@ ( )`

`<оп1> @ <оп2>` `<оп1>.operator@ ( <оп2> )` бинарная  
`<указатель> -> operator@ ( <оп2> )`

Вызов операции в теле метода:

`@ *this` `this -> operator@ ( )` унарная  
`this -> operator@ ( )`

`( *this ) @ <оп2>` `this -> operator@ ( <оп2> )` бинарная  
`this -> operator@ ( <оп2> )`

# Общие принципы перегрузки операций

Как правило, операции возвращают значения, которые нельзя изменять (right-value – rvalue). Поэтому, если таковым является объект, то он должен быть константным.

Если объекты (в т.ч. по ссылке) являются операндами, то:

Операнд	Операция перегружена	
	как метод	как функция
<i>Операции без побочного действия</i>		
Операнд унарной операции, левый операнд бинарной операции	Метод должен быть константным	Параметр должен быть ссылкой на константный объект
Правый операнд бинарной операции	Параметр должен быть ссылкой на константный объект	Параметр должен быть ссылкой на константный объект
<i>Операции с побочным действием (присваивание, инкремент, декремент)</i>		
Операнд ++, -- Левый операнд =, @=	Метод должен быть <b>НЕ</b> константным	Параметр должен быть ссылкой на <b>НЕ</b> константный объект
Правый операнд =, @=	Параметр должен быть ссылкой на константный объект	Параметр должен быть ссылкой на константный объект

# Особенности перегрузки операций

Большинство операций могут быть перегружены с произвольными типами параметров и возвращаемых значений, но:

- Операции отношений и логические операции, как правило, возвращают **bool**
- Операции присваивания (все), префиксные инкремент и декремент, как правило, возвращают ссылку на константный объект – левый операнд, поэтому удобнее перегружаются как методы (с использованием **return \*this;** )
- Арифметические и поразрядные операции, постфиксные инкремент и декремент, как правило, возвращают константный объект (не ссылку) и удобнее реализуются с использованием перегруженной операции сложного присваивания

Особые правила и/или синтаксис перегрузки имеют:

- Присваивание объектов одного класса
- Постфиксные инкремент и декремент
- Приведение типа
- Индексирование
- Вызов функции
- Косвенный селектор и получение адреса объекта
- Операции **new** и **delete**

# Перегрузка операции приведения типа

# Перегрузка операции индексирования



# Перегрузка операции вызова функции

`<Т.В.З.> operator () (<сп.форм.парам.>) [const];`

Особенности перегрузки:

- Перегружается только как метод
- Список формальных параметров – **любой**
- Тип возвращаемого значения – **любой**
- Может быть перегружена многократно в одном классе – как по списку формальных параметров, так и по признаку константности

# Перегрузка операции косвенного селектора

`<T.B.Z.> operator -> () [const];`

Особенности перегрузки:

- Перегружается только как метод
- Формальные параметры **отсутствуют**
- Тип возвращаемого значения должен быть **одним из**:
  - указатель на структурный тип (class, struct, union) к которому будет применен косвенный селектор
  - объект или ссылка на объект класса, в котором также перегружен косвенный селектор
- Может быть перегружена одновременно как константная и неконстантная

# Перегрузка операции получения адреса

`<T.B.Z.> operator & ( ) [const];`

Особенности перегрузки:

- Перегружается только как метод
- Формальные параметры **отсутствуют**
- Тип возвращаемого значения – **любой** (обычно указатель на объект)
- Может быть перегружена одновременно как константная и неконстантная
- Получить адрес любого объекта можно при помощи `std::addressof()` без использования операции `&`

# Перегрузка операции разыменования (разадресации)

`<T.B.Z.> operator * () [const];`

Особенности перегрузки:

- Перегружается только как метод
- Формальные параметры **отсутствуют**
- Тип возвращаемого значения – любой (обычно объект или ссылка на объект любого типа, но в т.ч. может быть встроенным типом)
- Может быть перегружена одновременно как константная и неконстантная