С/С++: Лекция 2

Воробьев Д.В

11.09.2020

1/42

Воробьев Д.В С/С++: Лекция 2 11.09.2020

Операторы

### Precedence

Порядок вызова операторов.

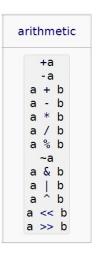
### Associativity

Свойство оператора, определяющее порядок расстановки скобок.

3/42

Воробьев Д.В С/С++: Лекция 2 11.09.2020

# Арифметические



## Арифметические

#### Правоассоциативные

```
+ (унарный) - (унарный)
```

#### Левоассоциативные

```
+ (бинарный) - (бинарный) * / % | ^« »
```

```
#include <iostream>
int main() {
    // ((7 - 7) - 7) = -7
    std::cout << 7 - 7 - 7;

    // ((24 / 4) / 2) = 3
    std::cout << 24 / 4 / 2;
    return 0;
}</pre>
```

5/42

# Икремент / декримент

increment decrement ++a - - a a++ a--

Воробьев Д.В С/С++: Лекция 2

6/42

# Икремент / декримент

### Префиксный

- 1. увеличить значение
- 2. вернуть значение

```
#include <iostream>
int main() {
    int x = 10;
    // 11
    std::cout << ++x;
    // 11
    std::cout << x;
}</pre>
```

# Икремент / декримент

## Постфиксный

- 1. сохранить значение
- 2. увеличить значение
- 3. вернуть сохраненное

```
#include <iostream>
int main() {
    int x = 10;
    // 10
    std::cout << x++;
    // 11
    std::cout << x;
}</pre>
```

## Скомбиниурем изученные операторы

```
#include <iostream>
int main() {
   int x = 10;
   std::cout << ---x;
   return 0;
}</pre>
```

### Видим - - и - с одинаковым приоритетом и правоассоциативные

```
Right-to-left
++a --a
+a -a
(type)
*a
&a
sizeof
co await
new new[]
delete delete[]
```

Воробьев Д.В С/С++: Лекция 2 11.09.2020 10/42

#### Получаем

```
#include <iostream>
int main() {
   int x = 10;
   std::cout << --(-x);
   return 0;
}</pre>
```

Да, тут ошибка. Но это демонстрирует, что порядок действий правильный. Пример про associativity, а не value category.

Воробьев Д.В С/С++: Лекция 2 11.09.2020 11/42

## Причина ошибки

- 1. префиксный принимает Ivalue expression
- 2. у нас у это -х –
- 3. -х это prvalue expression
- 4. не состыковочки

Воробьев Д.В С/С++: Лекция 2 11.09.2020 12/42

```
#include <iostream>

int main() {
    int x = 10;
    // mym sce or выведет 9
    std::cout << -(--x);
    return 0;
}
```

```
#include <iostream>
int main() {
    int x = 10;
   // доказывает
   // что скобки так -(x--)
   // m.\kappa. ecsu mak (-x)--
    // то была бы ошибка
    // но ее нет
    std::cout << -x--;
    return 0;
```

13 / 42

# Присваивание

#### assignment

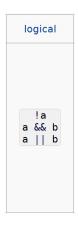
14 / 42

# Присваивание

```
#include <iostream>
int main() {
    int x = 10;
    int y = 6;
    x \&= y;
    // 2
    std::cout << x;
    return 0;
}
```

```
#include <iostream>
int main() {
    int x = 10;
    int y = 6;
    x = x \& y;
    // 2
    std::cout << x;
    return 0;
```

## Логические



Воробьев Д.В С/С++: Лекция 2 11.09.2020 16/42

# Сравнения

#### comparison

17 / 42

Воробьев Д.В С/С++: Лекция 2 11.09.2020

# Доступа

member access a[b] \*a &a a->b a.b a->\*b a.\*b

18 / 42

Воробьев Д.В С/С++: Лекция 2 11.09.2020

# Доступа

```
#include <iostream>
struct A {
    int x = 10;
};
int main() {
    A* p = new A;
    std::cout << p->x;
    std::cout << (*p).x;
    return 0;
```

```
#include <iostream>
int main() {
    int* p = new (5);
    // 5
    std::cout << *p;</pre>
    int x = 10;
    p = &x;
    // 10
    std::cout << p;
    return 0;
```

#### При переименовании member

#### 2 изменения

#### 4 изменения

```
#include <iostream>
struct S {
    int member = 10;
};
int main() {
    int S::* ptr = &S::member;
    Sa;
    std::cout << a.*ptr;
    S b;
    std::cout << b.*ptr;
    Sc;
    std::cout << c.*ptr;
    return 0;
```

```
#include <iostream>
struct S {
    int member = 10;
};
int main() {
    Sa:
    std::cout << a.member;</pre>
    Sb;
    std::cout << b.member;</pre>
    Sc;
    std::cout << c.member;</pre>
    return 0;
```

# Другие



Воробьев Д.В

# Тернарный

```
#include <iostream>
int main() {
   int x = 0;
   int y = 1;

   std::cout << a > b ? a : b;
   return 0;
}
```

### Запятая

```
#include <iostream>
int main() {
   int n = 1;
   int m = (++n, std::cout << "n = " << n << '\n', ++n, 2*n);
    std::cout << "m = " << (++m, m) << '\n';
   return 0;
```

23 / 42

## sizeof, alignof

- sizeof возвращает размер объекта
- alignof возвращает число байт требуемое для выравнивания

```
#include <iostream>
struct C {
    char x;
    int y;
};
int main() {
    // 1
    std::cout << alignof(char);</pre>
    1/4
    std::cout << alignof(int);</pre>
    // выравнивание no int
    std::cout << sizeof(C);</pre>
    return 0;
```

#### lvalue и rvalue

### Определение

Ivalue - expression, которому можно сделать присвоение rvalue - не Ivalue expression

#### Предупреждение

Это грубое определение. Более аккуратно во 2-ой части курса.

Воробьев Д.В С/С++: Лекция 2 11.09.2020 25/42

# Перегрузка функций

### Перегрузка функции

Определение более 2-ух функций в одинаковом scope с разным списком параметров.

#### Замечание

Определение не полное для методов структуры / класса требуется уточнение.

26 / 42

# Перегрузка функций

#### Можно

#### Нельзя

```
#include <iostream>

void func(double a) {}

void func(int a) {}

int main() {
    return 0;
}
```

```
#include <iostream>

void func(int a) {}
int func(int a) {}

int main() {
    return 0;
}
```

## Функции с аргументами по умолчанию

### Указываются последними в списки параметров

#### Можно

#### Нельзя

```
#include <iostream>
void func(int a, int b = 0) {}
int main() {
    return 0;
}
```

```
#include <iostream>
void func(int b = 0, int a) {}
int main() {
    return 0;
}
```

## Явное приведение типов

### Проблема

Слева дали double\* указатель на float. При чтении будет считываться 8 байт. Памяти выделяли под 4 байта

C-cast static\_cast

```
#include <iostream>
int main() {
   float x = 3.1;
   // UB
   double* y = (double*) &x;
   return 0;
}

#include <iostream>
int main() {
   float x = 3.1;
   // CE
   double* y = static_cast<double*>(&x);
   return 0;
}
```

29 / 42

### Вывод

Используйте static cast. static cast проверит совместимость типов

Воробьев Д.В С/С++: Лекция 2 11.09.2020 30/42

Управляющие конструкции

Воробьев Д.В С/С++: Лекция 2 11.09.2020 31/42

```
int main() {
    int x = 10;
    if (x) {
        int y = 20;
    }
    int z = 10;
    return 0;
}
```

Воробьев Д.В С/С++: Лекция 2 11.09.2020 32/42

# if, else

```
int main() {
    int x = 10;
    if (x) {
        int y = 20;
    } else {
        int y = 5;
    }
    int z = 10;
    return 0;
}
```

Воробьев Д.В С/С++: Лекция 2 11.09.2020 33/42

## while

```
int main() {
   int x = 0;
   while(x < 1) {x++;}
   return 0;
}</pre>
```

### do-while

```
int main() {
   int j = 0;
   do {
        j++;
   } while (j < 2);
   return 0;
}</pre>
```

Воробьев Д.В С/С++: Лекция 2 11.09.2020 35/42

### for

```
int main() {
   for (std::size_t x = 0; x < 1; x++) {}
   return 0;
}</pre>
```

Воробьев Д.В С/С++: Лекция 2 11.09.2020 36/42

### switch

```
int main() {
    int x = 0;
    switch(x) {
        case 0 : {
            std::cout << 1;
            break;
        case 1 : {
            std::cout << 2;
            break;
        default: {
            std::cout << 3;
    // 1
    return 0;
```

### switch

```
int main() {
    int x = 0;
    switch(x) {
        case 0 : {
            std::cout << 1;
        case 1 : {
            std::cout << 2;
            break;
        default: {
            std::cout << 3;
    // 12
   return 0;
```

### switch

```
int main() {
    int x = 2;
    switch(x) {
        case 0 : {
            std::cout << 1;
        case 1 : {
            std::cout << 2;
            break;
        default: {
            std::cout << 3;
   return 0;
```

### break

```
int main() {
    int x = 10;
    while( x < 10) {
        break;
    }
    return 0;
}</pre>
```

Воробьев Д.В С/С++: Лекция 2 11.09.2020 40/42

## continue

```
int main() {
    int x = 10;
    while( x < 10) {
        continue;
    }
    return 0;
}</pre>
```

## Return

```
int main() {
    return 0;
}
```

оробьев Д.В С/С++: Лекция 2 11.09.2020 42/42