



# Операции языка С

## Лекция 5

*Иллюстративный материал к лекциям по алгоритмизации и программированию*

Автор Саблина Н.Г.

2016 г.



# Содержание



Операторы

Операции языка C

Арифметические операции

Преобразование типов

Логические операции

Поразрядные операции

Прочие операции

Автор





# Операторы

***Операторы в языке*** - это синтаксические конструкции, предназначенные как для записи алгоритмических действий по преобразованию данных, так и для задания порядка выполнения других действий.





# Язык содержит следующие операторы:

## *Простые операторы:*

- оператор присваивания;
- оператор функции;
- оператор перехода;
- пустой оператор;
- оператор цикла с  
постусловием;
- оператор цикла с  
параметром.

## *Структурные операторы:*

- условный оператор;
- оператор варианта;
- оператор цикла с  
предусловием;





# Операции языка Си

**Знак операции** - это символ или комбинация символов, которые сообщают компилятору о необходимости произвести определенные арифметические, логические или другие действия.



# Знаки операций языка C

[ ]	( )	.	->	++	--
&	*	+	-	~	!
sizeof	/	%	<<	>>	
<	>	<=	>=	==	!=
^		&&		?:	=
*=	/=	%=	+=	-=	<<=
>>=	&=	^=	=	,	



# Арифметические операции

- вычитание и унарный минус;
- + сложение;
- \* умножение;
- / деление;
- % деление по модулю;
- ++ увеличение на единицу (increment);
- уменьшение на единицу (decrement).



# Особенности некоторых операций

## (1)



- Операция **деления по модулю** % дает остаток от целочисленного деления.
  - Операция % может применяться **только к целочисленным переменным**.
- ++ (инкремент ) прибавляет единицу к операнду;
- -- (декремент) вычитает единицу из операнда;





# Инкремент и декремент

- Обе операции унарные, могут следовать
  - перед операндом (префиксная форма)
  - после операнда (постфиксная форма)
- Три оператора дают один и тот же результат

$X = X + 1;$

$++X;$

$X++;$

- Различия возникают при использовании инкремента в выражениях





# Пример 1

```
#include <stdio.h>

main()
{ int x=5;
  int y=60;
  x++;
  ++y;
  printf("x=%d y=%d\n", x, y);
  printf("x=%d y=%d\n", x++, ++y);
}
```





# Результат работы этой программы

$x=6, y=61;$

$x=6, y=62.$

$x++$  - значение переменной  $x$  сначала используется в выражении,  
затем переменная увеличивается на единицу;

$++x$  - переменная  $x$  сначала увеличивается на единицу, а затем ее  
значение используется в выражении.





# Старшинство арифметических операций следующее:

++, --

- (унарный минус)

\*, /, %

+, -

Операции, одинаковые по старшинству, выполняются в порядке слева направо.

Для того, чтобы изменить порядок операций, могут использоваться круглые скобки.





# Приведение типов

- Если операнды имеют один тип, то результат арифметической операции имеет тот же тип.
- Поэтому, когда операция деления / применяется к целым переменным или символьным переменным, остаток отбрасывается.

Например:

$$11/3=3$$

$$1/2 =0$$





# Преобразование типов при вычислении выражений

- **Выражение** в языке C - это некоторая допустимая комбинация переменных, констант и операций.
- Если операнды операции принадлежат разным типам, то они приводятся к некоторому общему типу.
- Различаются **неявные** и **явные** преобразования типов.





# Неявные преобразования типов

Неявные преобразования транслятор выполняет без вмешательства программиста.

Они применяются всякий раз, когда смешиваются различные типы данных.

Такие преобразования выполняются согласно правилам, называемым стандартными преобразованиями.



# Правила автоматического приведения типов при вычислениях (1):



1. Все переменные типа **char** и **short int** преобразуются в **int**, все переменные типа **float** преобразуются в **double**.
2. Для любой пары операндов:
  - если один из операндов **long double**, то и другой преобразуется в **long double**;
  - если один из операндов **double**, то и другой преобразуется в **double**;



# Правила автоматического



## приведения типов при вычислениях (2):

- если один из операндов **long**, то и другой преобразуется в **long**;
- если один из операндов **unsigned**, то и другой преобразуется в **unsigned**.

3. В операторе присваивания конечный результат приводится к типу переменной в левой части оператора присваивания, при этом тип может как повышаться (расширение типа), так и понижаться (сужение типа).





# Пример неявного преобразования типа

```
int i = 3.14; // 3.14 преобразуется к int (i=3)
```

- константа 3.14 типа **double** неявно преобразуется транслятором в тип **int** - дробная часть отброшена.
- Компилятор C не выдаст предупреждения.



# Примеры приведения типов

- Чему равны значения переменных **y** и **y1** после выполнения операторов:

```
int i=5;
```

```
float y, y1, x;
```

```
y=i/4;
```

```
x=i;
```

```
y1=x/4;
```

- Чему равны значения переменных **x**, **y** и **y1** после выполнения операторов:

```
int t=9;    float y, y1, x;
```

```
y=t/4;      x=t;
```

```
y1=x/4;    x=t%4
```

# Явные преобразования типов

Тип результата вычисления выражения можно изменить, используя конструкцию «приведение» (casts), имеющую следующий вид:

(тип) выражение

Здесь «тип» - один из стандартных типов данных языка С.



# Операции отношения

используются для сравнения

< меньше,

<= меньше или равно,

> больше,

>= больше или равно,

== равно,

!= не равно.





# Логические операции:

Бинарные операции

&& и (AND),

|| или (OR),

Унарная операция

! не (NOT).

Операнды – логического типа; принимают значения:

истинно ("**true**") или ложно ("**false**").





## Пример 2

```
#include <stdio.h>

main()

{ int tr, fal;

tr = (101<=105); /*выражение "истинно" */
fal = (101>105); /*выражение "ложно" */
printf("true - %d , false - %d\n", tr, fal);

}
```

Программа выведет на экран: **true – 1 , false - 0**





# Таблица истинности

X	Y	X AND Y	X OR Y	NOT X	X XOR Y
1	1	1	1	0	0
1	0	0	1	0	1
0	1	0	1	1	1
0	0	0	0	1	0





# Старшинство логических операций и операций отношения

Старшая !

> < >= <=

== !=

&&

Младшая ||



# Операция присваивания (1)

обозначается =

Пример:

```
if ((f=x-y)>0) printf ("Число x, больше чем y)
```

Порядок выполнения:

- вычисляется величина  $x-y$ ,
- результат присваивается переменной  $f$ ,
- сравнивается ее значение с нулем.



# Операция присваивания (2)

Многократное присваивание выполняется

справа налево :

$$a = b = c = x * y;$$

- Сначала вычисляется значение  $x * y$
- затем это значение присваивается  $c$ ,

потом  $b$  и затем  $a$





# Операция присваивания (3)

дополнительные операции присваивания:

$+=$ ,  $-=$ ,  $/=$   $*=$  и  $\%=$ .

**Например:**

$m-=20$  то же самое, что и  $m=m-20$

$m\%=10$  то же самое, что и  $m=m\%10$





# Поразрядные операции (побитовые операции)

Поразрядные операции можно проводить с любыми целочисленными переменными и константами.

Нельзя использовать эти операции с переменными типа **float**, **double** и **long double**.

Результатом побитовой операции будет целочисленное значение.





# Поразрядные операции

& AND,

| OR,

^ XOR,

~ NOT,

<< СДВИГ влево,

>> СДВИГ вправо.





# Пример 3

Если надо установить значение старшего разряда переменной типа `char` равным нулю, то удобно применить операцию `&` (AND):

```
ch=ch&127;
```

Пусть `ch='A'`

```
'A'  11000001
```

```
127  01111111
```

```
-----
```

```
'A'&127 01000001
```





# Пример 4

Если же мы хотим установить старший разряд равным единице, то удобна операция OR:

```
ch = ch | 128;
```

```
'A'    11000001
```

```
128    10000000
```

-----

```
'A'|128 11000001
```





# Операция условие

**Операция условие** - единственная операция языка С, имеющая три операнда.

Эта операция имеет вид:

$(\text{выр1})?(\text{выр2}):(\text{выр3})$

**Например:**

$\text{Max} = X > Y ? X : Y;$





# Операция запятая

Операция запятая имеет самый низкий приоритет из всех операций языков С и С++.

Операция запятая выполняется слева направо, и ее значением является значение правого операнда.

В выражении (выр1), (выр2) сначала вычислится значение (выр1), затем - значение (выр2).

**Например:** `int x, y=5;`

`x=(y+2, y-3); //результат: x=2`

Скобки необходимы, поскольку оператор «запятая» имеет более низкий приоритет по сравнению с оператором присваивания (результат без скобок `x=7`)





# Операция sizeof

Имеет две формы:

- sizeof (тип)
- sizeof (выражение).

Результат - целочисленное значение длины типа или выражения в байтах.

При использовании второй формы значение выражения не вычисляется, а лишь определяется его тип.

**Например:** `int x=2, y=3, z; float b=5.5;`

`z=sizeof(x); z=sizeof(int); z=sizeof(b); z=sizeof(float);`





# Итоги

## Рассмотренные вопросы:

- Операторы
- Преобразование типов
- Операции
  - Логические
  - Поразрядные
  - Арифметические
  - Прочие операции
  - Старшинство операций





# Определение некоторых понятий

**Операнд** – вложенное выражение в выражении C++, воздействующее на некоторый оператор.

**Оператор** – лексема в выражении C++, которая приводит к значению данного типа, и возможно имеющая побочные эффекты. Оператору дается от одного до трех выражений в качестве операндов

**Определение типа** – декларация, которая даёт типу имя

**Преобразование типа** – представление значения одного типа как значение его допустимого представления, имеющего другой  
ТИП.





# Библиографический список

- Подбельский В.В., Фомин С.С. Курс программирования на языке Си: учебник. М.: ДМК Пресс, 2012. – 384 с.
- Павловская Т.А. С/С++. Программирование на языке высокого уровня: учебник для студентов вузов, обучающихся по направлению "Информатика и вычисл. техника" СПб.: Питер, 2005. - 461 с.
- Павловская Т. А., Щупак Ю. А. С++. Объектно-ориентированное программирование. Практикум. Практикум. — СПб.: Питер, 2006. — 265 с: ил.
- Березин Б.И. Начальный курс С и С++ / Б.И. Березин, С.Б. Березин. - М.: ДИАЛОГ-МИФИ, 2001. - 288 с
- Каширин И.Ю., Новичков В.С. От С к С++. Учебное пособие для вузов. – М.: Горячая линия – Телеком, 2012. – 334 с.





Автор:

Саблина Наталья Григорьевна

Ст. преподаватель

каф. РТС УрФУ

