

## Лекция 5

# Логические выражения. Функции форматного ввода и вывода

---

Преподаватель Палехова Ольга Александровна,  
кафедра О7 БГТУ «Военмех»

# Логические операции

---

Логические операции:

- **!** (НЕ, отрицание, инверсия)
- **&&** (И, логическое умножение, конъюнкция)
- **||** (ИЛИ, логическое сложение, дизъюнкция)

$x \in (a, b)$

$a < x \ \&\& \ x < b$

$x \notin (a, b)$

$! ( a < x \ \&\& \ x < b )$

$x \leq a \ || \ x \geq b$



Результатом логического выражения будет число:  
1 – выражение истинно, 0 – выражение ложно.

# Сравнение с нулем



Любое ненулевое значение считается истинным.

выражение	значение выражения	
	$a = 0$	$a \neq 0$
$a \neq 0$	0 (ложь)	1 (истина)
$a$	0 (ложь)	$a$ (истина)
$a == 0$	1 (истина)	0 (ложь)
$!a$	1 (истина)	0 (ложь)

$a \neq 0 \ \&\& \ b \neq 0 \rightarrow a \ \&\& \ b$



Чему будет равно значение этого выражения, если  $a=5$ ,  $b=2$ ?

# Вычисление сложных логических выражений

Выражения вычисляются слева направо в соответствии с приоритетом операций

2 1 6 3 5 4  
!(a > b) || c != d && b == a

**!** Оптимизация вычислений логических выражений -  
**Вычисление логического выражения прекращается, как только становится известным результат.**

y && x < 0 && sin(x/y) > 0

вычисляется  
всегда

вычисляется  
только если  
y≠0

вычисляется только  
если истинны два  
первых условия

# Топ-5 ошибок в записи условий

ошибка	примеры (a=2, b=5, x=10)	
	выражения	значения
= вместо ==	$a == b$ $a = b$	0 (ложь) 5 (истина)
математическая запись двойного неравенства	$a < x \ \&\& \ x < b$ $a < x < b$	0 (ложь) 1 (истина)
, вместо \&\&	$a > b \ \&\& \ x > b$ $a > b \ , \ x > b$	0 (ложь) 1 (истина)
\& вместо \&\&	$a \ \&\& \ b$ $a \ \& \ b$	1 (истина) 0 (ложь)
потеря сравнения в сложном условии	$x < a \    \ x < b$ $x < a \    \ b$	0 (ложь) 1 (истина)

# Инструкции

---

**Инструкция** (оператор, команда) – наименьшая автономная часть программы, выполняющая какое-то действие.

Инструкции языка Си:

- **инструкция-выражение** – любое выражение, заканчивающееся точкой с запятой;
- **составная инструкция (блок)** – последовательность инструкций, заключенная в фигурные скобки **{}**;
- операторы выбора:
  - инструкция ***if-else*** (условный оператор);
  - переключатель ***switch***;
- операторы цикла:
  - циклы с предусловием ***while*** и ***for***;
  - цикл с постусловием ***do ... while***;
- операторы перехода: ***goto*** (безусловный переход), ***break***, ***continue***, ***return*** (выход из функции).

# Функции форматного ввода и вывода

---

Для использования стандартного ввода-вывода в тексте программы пишем

```
#include <stdio.h>
```

Функция форматного ввода

**scanf (формат ввода, адреса)**

**Формат ввода** – символьная строка, которая показывает, как воспринимать содержимое потока ввода.

Функция форматного вывода

**printf (формат вывода, значения)**

**Формат вывода** – символьная строка, которая показывает, как должен выглядеть выводимый текст.

Функции *scanf()* и *printf()* являются универсальными и могут использоваться для ввода-вывода данных всех стандартных типов

# Функция форматного ввода *scanf()*

---

***scanf*** (формат ввода, адреса)

Функция ***scanf()*** преобразует последовательность символов, находящихся в потоке ввода, во внутреннее представление, соответствующее форматной строке, и записывает полученный код по указанным адресам. Возвращаемый **результат** – количество преобразованных значений

Форматная строка содержит символы, которые должны быть введены, и спецификации преобразования. Общий вид спецификации преобразования:

**%**\* ширина\_поля\_ввода модификатор **спецификатор\_типа**

Обязательно



# Ввод целых чисел с клавиатуры

scanf –  
форматный ввод

формат ввода

адреса ячеек, куда  
записать введенные  
числа

```
scanf ("%d%d", &a, &b);
```

ждать ввода с клавиатуры двух целых чисел (через пробел или *Enter*), первое из них записать в переменную a, второе – в b

входная строка	результат функции	значения переменных
3 5	2	a=3, b=5
3,5	1	a=3, b не изменяется
три пять	0	a и b не изменяются
a 5	0	a и b не изменяются

# Спецификаторы и модификаторы

	СИМВОЛ	ТИП ВВОДИМОГО ДАННОГО
спецификаторы	<b>d</b>	десятичное целое: <i>int</i>
	<b>i</b>	целое: <i>int</i> . Целое может быть восьмеричным (с 0 слева) или шестнадцатеричным (с 0x или 0X слева)
	<b>u</b>	беззнаковое десятичное целое: <i>unsigned int</i>
	<b>c</b>	символы: <i>char</i> . Считывается любой символ, в том числе разделитель
	<b>s</b>	строка
	<b>e, f, g</b>	число с плавающей точкой: <i>float</i>
модификаторы	<b>L</b>	<i>long double</i>
	<b>l</b>	<i>long, unsigned long, double</i>
	<b>h</b>	<i>short</i>

# Ширина поля ввода

---

**Ширина поля ввода** – максимально возможное число символов, подлежащее преобразованию. Число символов будет меньше, если раньше встретится пробел или символ, который не может быть преобразован по данной спецификации.

```
scanf ("%3d%2d%4d", &a, &b, &c);
```

На входе:

123|45|678

Значения переменных:

a = 123, b = 45, c = 678

```
scanf ("%2d.%2d.%4d", &dd, &mm, &yy);
```

# «\*» в спецификации ввода

---

«\*» запрещает запись значения

Ввод времени

```
scanf ( "%2d:%2d" , &h , &m ) ;
```

разделителем может быть только двоеточие

А если на входе:

12-37 ?

```
scanf ( "%2d%*c%2d" , &h , &m ) ;
```

пропускаем один любой символ

# Функция форматного вывода *printf()*

---

**printf** (форматная строка, значения)

Функция преобразует внутреннее представление данных в последовательность символов, соответствующую форматной строке, и записывает ее в поток вывода. Возвращаемый **результат** – количество выведенных символов.

Форматная строка содержит произвольный текст и спецификации преобразования. Общий вид спецификации преобразования:

**%** флаги ширина\_поля\_вывода . точность модификатор **спецификатор**



Обязательно

Спецификации преобразования должны соответствовать типу указанных аргументов.

# Вывод чисел на экран

здесь вывести  
целое число

это число взять  
из ячейки C

```
printf ("%d", c);
```

```
printf ("Результат: %d", c);
```

```
printf ("%d+%d=%d", a, b, c );
```

формат вывода

список значений

```
printf ("%d+%d=%d", a, b, a+b );
```

арифметическое  
выражение

# Спецификации формата *printf()*

СИМВОЛ	тип аргумента, вид печати
<b>d, i</b>	<b>int</b> ; десятичное целое
<b>u</b>	<b>unsigned int</b> ; беззнаковое десятичное целое
<b>c</b>	<b>char, int</b> ; один символ
<b>s</b>	строка
<b>f</b>	<b>double</b> ; десятичная дробь, количество знаков после точки задается точностью (по умолчанию равно 6)
<b>e, E</b>	<b>double</b> ; вещественное число в экспоненциальной форме записи, количество знаков после точки в мантиссе задается точностью (по умолчанию равно 6)
<b>g, G</b>	<b>double</b> ; использует %e или %E, если порядок меньше, чем -4, или больше или равен точности; в противном случае использует %f. Завершающие нули и завершающая десятичная точка не печатаются

# Ширина поля вывода и точность

---

**Ширина поля вывода** – минимальное количество знакомест, в которое помещается выводимое значение. Если количество цифр в числе меньше ширины поля, недостающее количество символов дополняется пробелами. Если число символов в изображении значения больше ширины поля, то ширина поля игнорируется.

**Точность** указывается с помощью точки и числа после него. Точность задает минимальное количество цифр при выводе целого числа, число цифр после точки при выводе вещественного числа с помощью %f и %e или максимальное число значащих цифр при спецификации %g.



# Вывод целых чисел

```
int x = 1234;  
printf ("%d", x);
```

или "%i"

1234

минимальное  
число позиций

или "%9i"

```
printf ("%9d", x);
```

1234

всего 9 позиций

5

4

# Вывод вещественных чисел

```
double x = 123.4567;  
printf ("%f", x);
```

123.456700

минимальное число  
позиций, 6 цифр в  
дробной части

```
printf ("%9.3f", x);
```

123.457

всего 9 позиций,  
3 цифры в дробной  
части. Округление!

```
printf ("%e", x);
```

1.234560e+02

стандартный вид:  
 $1,23456 \cdot 10^2$

```
printf ("%10.2e", x);
```

1.23e+02

всего 10 позиций,  
2 цифры в дробной  
части мантииссы

# Точность при выводе целых чисел

```
int x=1234, y=12;  
printf (".3d .4d", x, y);
```

Точность  
игнорируется

1234 0012

Недостающее число знаков  
дополняется нулями

```
int dd = 5, mm = 10, yy = 2020;  
printf ("%4.2d.%.2d.%.2d", dd, mm, yy);
```

Ширина  
поля

Вывести  
2 знака

Вывести  
точку

Вывести  
2 знака

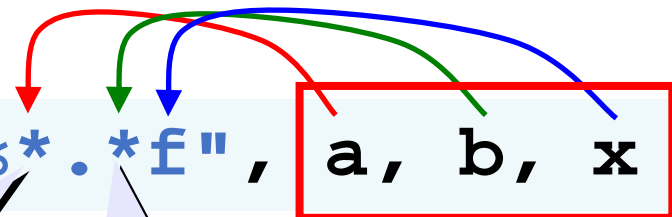
Вывести  
точку

05.10.2020

# Вычисляемые ширина поля и точность

```
int a=10, b=2;  
double x=123.4567;
```

```
printf ( "x=%*.*f", a, b, x );
```



ширина  
поля

точность

список значений

```
x= 123.46
```

```
printf ( "x=%*.*f", a-b, a/3, x*b );
```

```
x= 246.913
```

арифметические  
выражения

# Флаги

флаг	назначение
-	выравнивание по левой границе поля вывода
+	принудительный вывод знака числа
пробел	пробел вместо знака числа перед положительными числами
0	дополнение числа до ширины поля нулями (не с флагом "-")
#	со спецификаторами o, x, X – вывод префикса (0, 0x, 0X); со спецификаторами f, g, G – вывод точки при нулевой ширине поля

```
double m = 123.4567;  
printf ("m = %+ -#6.fg", m);
```

вывести +

по левому краю

вывести .

только целая  
часть

```
m = +123. g
```