

Информатика

Язык С. Выражения, операции, операторы

Гирик Алексей Валерьевич

Университет ИТМО

2022





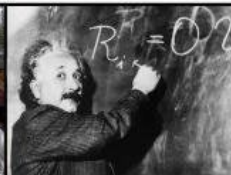



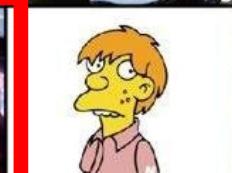




















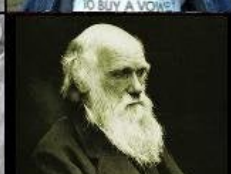


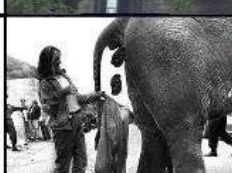



Материалы курса

- Презентации, материалы к лекциям, литература, задания на лабораторные работы
 - shorturl.at/jqRZ6



Общие сведения о языке С

Язык С

Java	C	PHP	Ruby	Haskell	Lisp	as seen by...
						Java fans
						C fans
						PHP fans
						Ruby fans
						Haskell fans
						Lisp fans

Язык не очень высокого уровня

■ 1970 – C, Деннис Ритчи



1970 - Niklaus Wirth creates Pascal, a procedural language. Critics immediately denounce Pascal because it uses " $x := x + y$ " syntax instead of the more familiar C-like " $x = x + y$ ". This criticism happens in spite of the fact that C has not yet been invented.

1972 - Dennis Ritchie invents a powerful gun that shoots both forward and backward simultaneously. Not satisfied with the number of deaths and permanent maimings from that invention he invents C and Unix.

1972 - Alain Colmerauer designs the logic language Prolog. His goal is to create a language with the intelligence of a two year old. He proves he has reached his goal by showing a Prolog session that says "No." to every query.

1973 - Robin Milner creates ML, a language based on the M&M type theory. ML begets SML which has a formally specified semantics. When asked for a formal semantics of the formal semantics Milner's head explodes. Other well known languages in the ML family include OCaml, F#, and Visual Basic.

1980 - Alan Kay creates Smalltalk and invents the term "object oriented." When asked what that means he replies, "Smalltalk programs are just objects." When asked what objects are made of he replies, "objects." When asked again he says "look, it's all objects all the way down. Until you reach turtles."

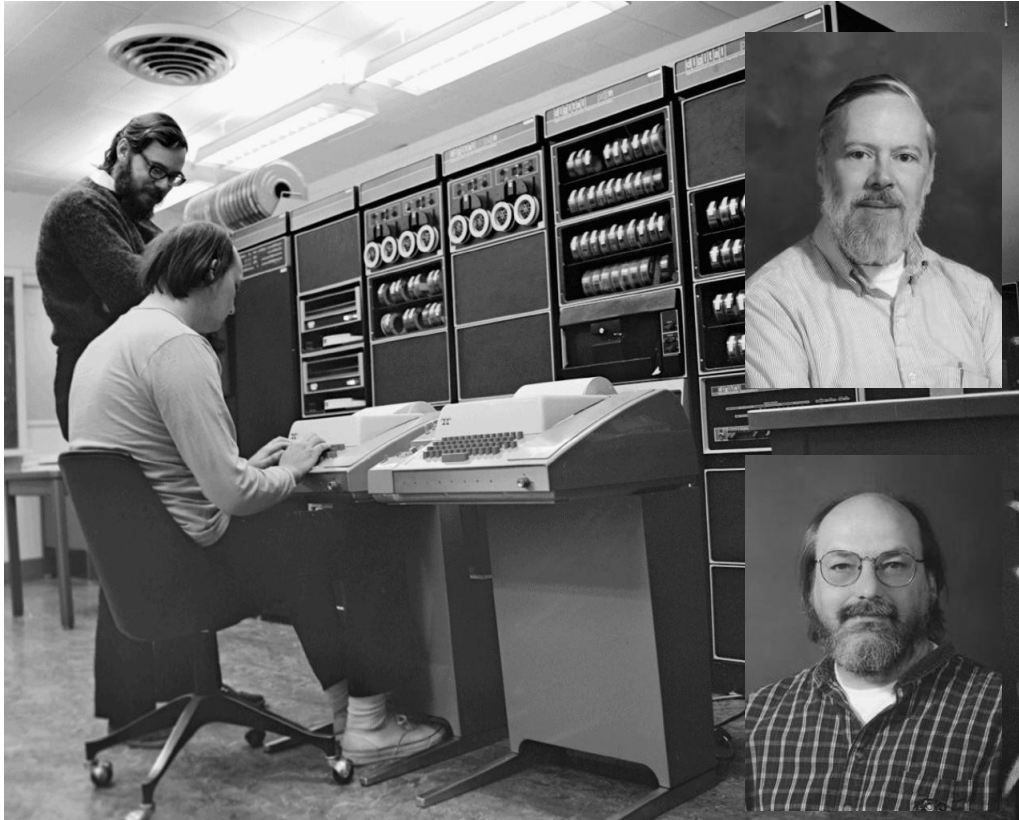
[A brief, incomplete, and Mostly Wrong History of Programming Languages](#)

by James Iry

История языка C

- 1963 – **CPL** (Combined Programming Language)
- 1966 – **BCPL** (Basic CPL), Martin Richards
- 1968 – **Bon**, Ken Thompson
- 1969 – **B**, Ken Thompson
- 1971 – **NB**, Dennis Ritchie
- 1972 – **C**, Dennis Ritchie
- **1973 – C используется для написания ядра UNIX**
- 1978 – **K&R C**, Brian Kernighan and Dennis Ritchie
- 1989 – **ANSI C**, ANSI X3.159-1989
- 1990 – **C90**, ISO/IEC 9899:1990
- 1999 – **C99**, ISO/IEC 9899:1999
- 2011 – **C11**, ISO/IEC 9899:1999

C

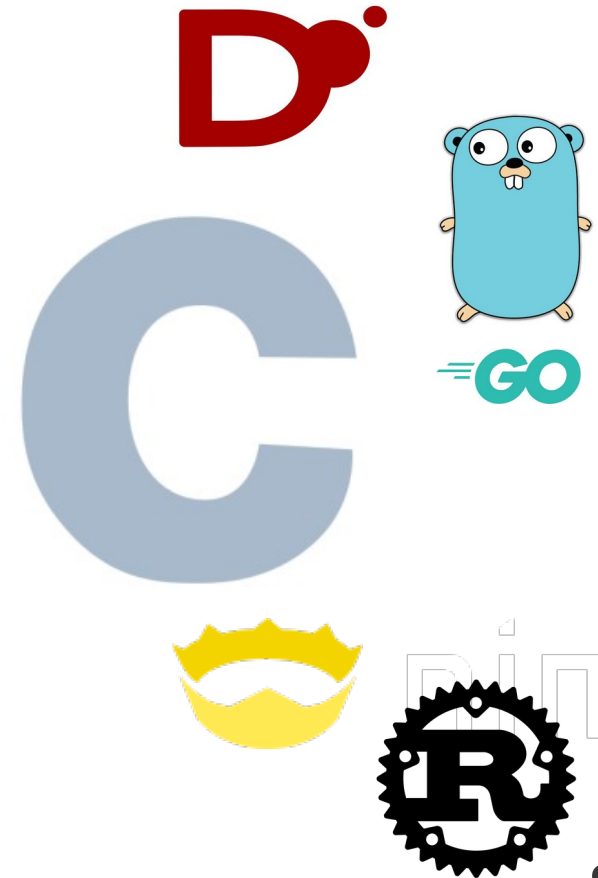


■ Деннис Ритчи и Кен Томпсон (Dennis Ritchie and Ken Thompson)

- 1969 – 1972 – первые версии
- 1978 – «K&R C»
- 1989 – ANSI C
- 1999 – ANSI C99
- 2011 – ISO C11
- 2017 – ISO C17
- 2023 – ISO C23

Языки системного программирования

- ассемблеры
 - разные для разных архитектур
- старые языки, которые потеряли актуальность
 - BLISS
 - Ada (?)
 - ...
- "неудачники"
 - Pascal/Oberon/Modula-2/...
 - D
 - Go (?)
- **актуальные**
 - **C**
 - **C++**
- wannabes
 - Rust
 - Nim (? controlled gc)
 - V



Почему С?

- **простой синтаксис**
 - легко освоить за короткое время
- **удовлетворяет всем требованиям к языкам системного программирования**
 - а также стандартизован ISO
- **один из самых популярных языков**
 - высокая вероятность того, что пригодится в дальнейшей карьере
 - хороший фундамент при освоении "похожих" на С языков

Хеллоуворлд

```
// Compile with: gcc -o hello hello.c
```

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

```
int main() {
```

```
    printf("Hello, ugly and unforgiving world!\n");
```

```
    return 0;
```

```
}
```

Компиляция из командной строки

- GNU Compiler Collection

- cc

- gcc



```
> cc -o example example.c
```

```
> ./example
```

```
> gcc -o example example.c
```

```
> ./example
```

Компиляция из командной строки

- LLVM
 - ▣ clang



```
> clang -o example example.c  
> ./example
```

Литература по С

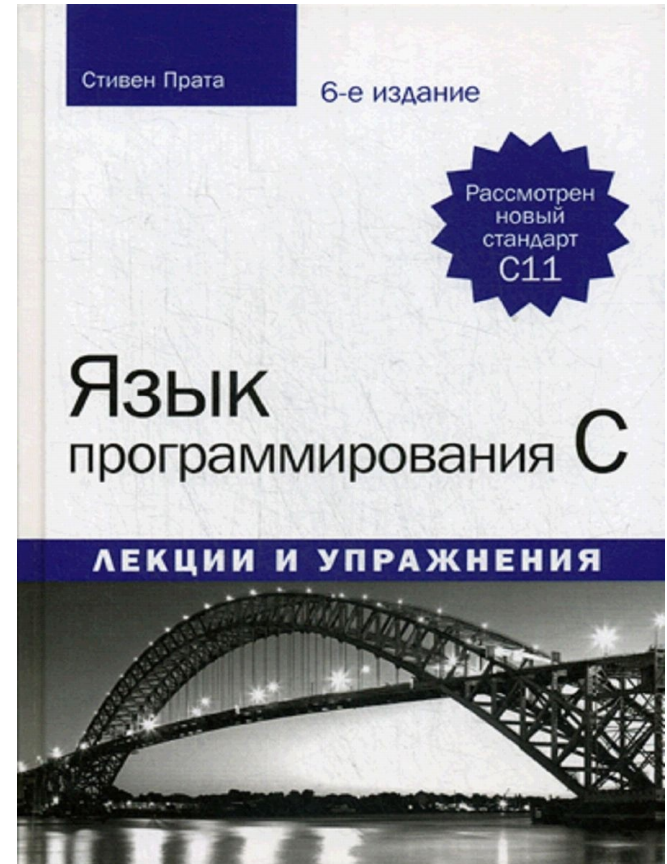
Прата С.

Язык программирования С.
Лекции и упражнения.

Шестое издание

2014 (2015), 928 с.

Подробная книга, хороша для тех, кто совсем не знает С, но "многобукаф".



Литература по С

Керниган Б., Ритчи Д.

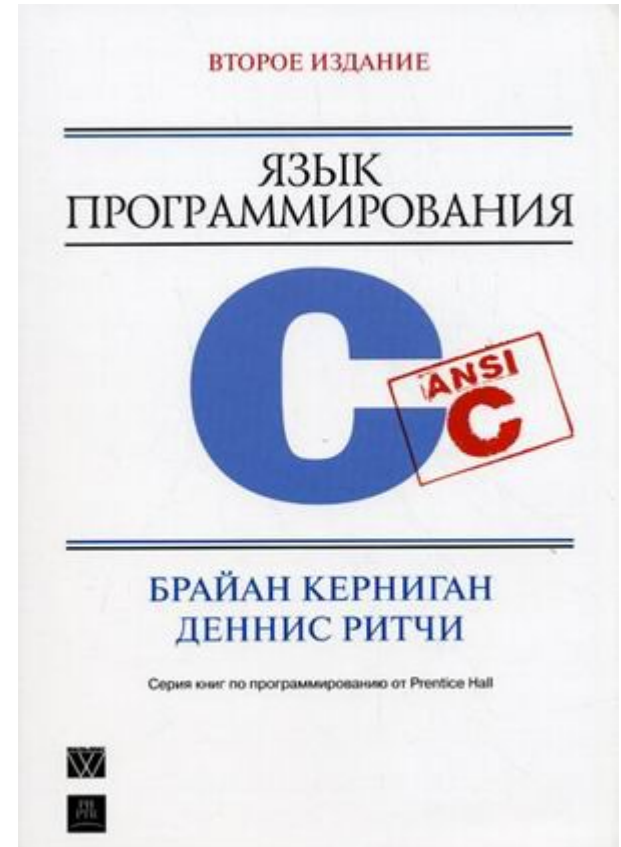
Язык программирования С.

Второе издание

1988 (2009), 304 с.

Классика!

Обязательна к прочтению!



Литература по С

Подбельский В.В., Фомин С.С.

Курс программирования на
языке Си

2012, 385 с.

Тоже вполне неплохой учебник.



Литература по С

Сикорд Р.

Эффективный С

2020 (2022), 304 с.

Посложнее предыдущих.



Литература по C

Jens Gustedt

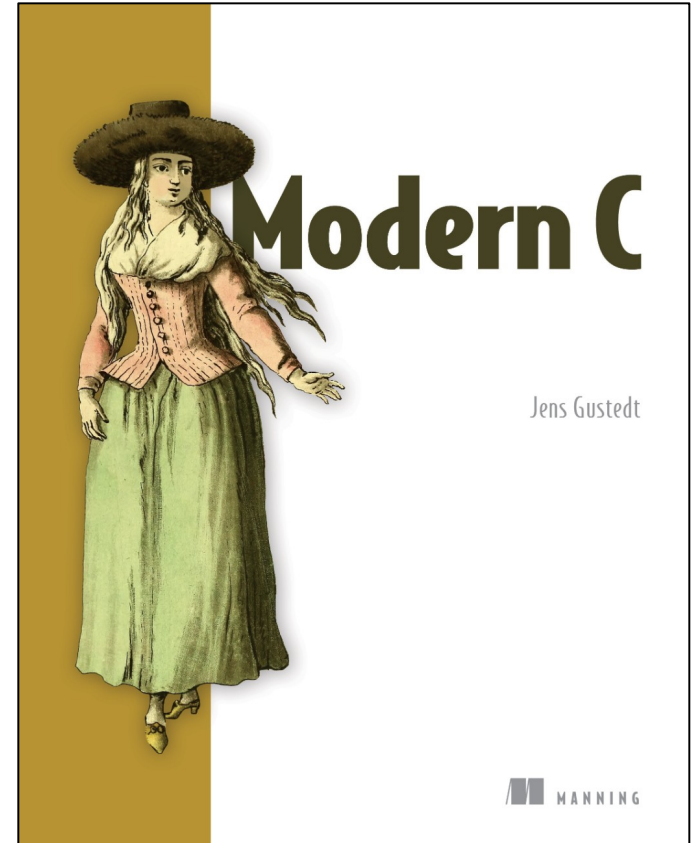
Modern C

2019, 408 pages

Фор зоус ху кен рид ин инглиш.

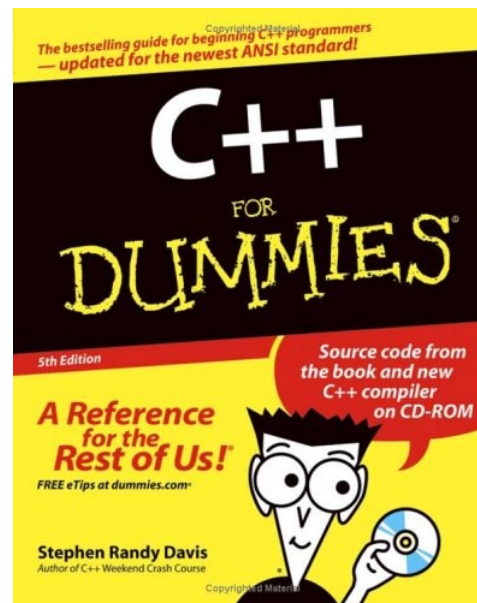
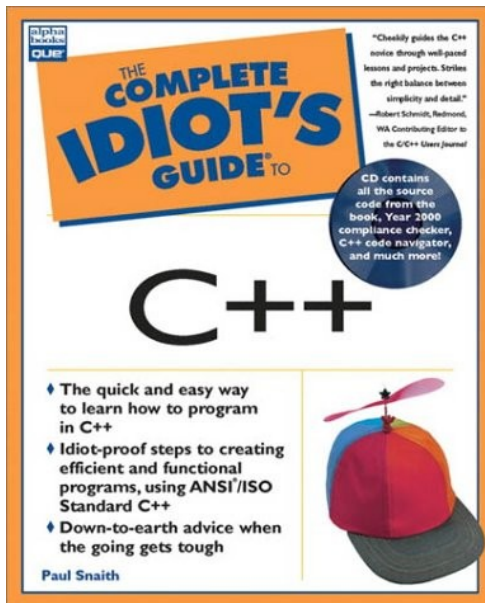
Зис бук из квайт диффикалт

энивей.



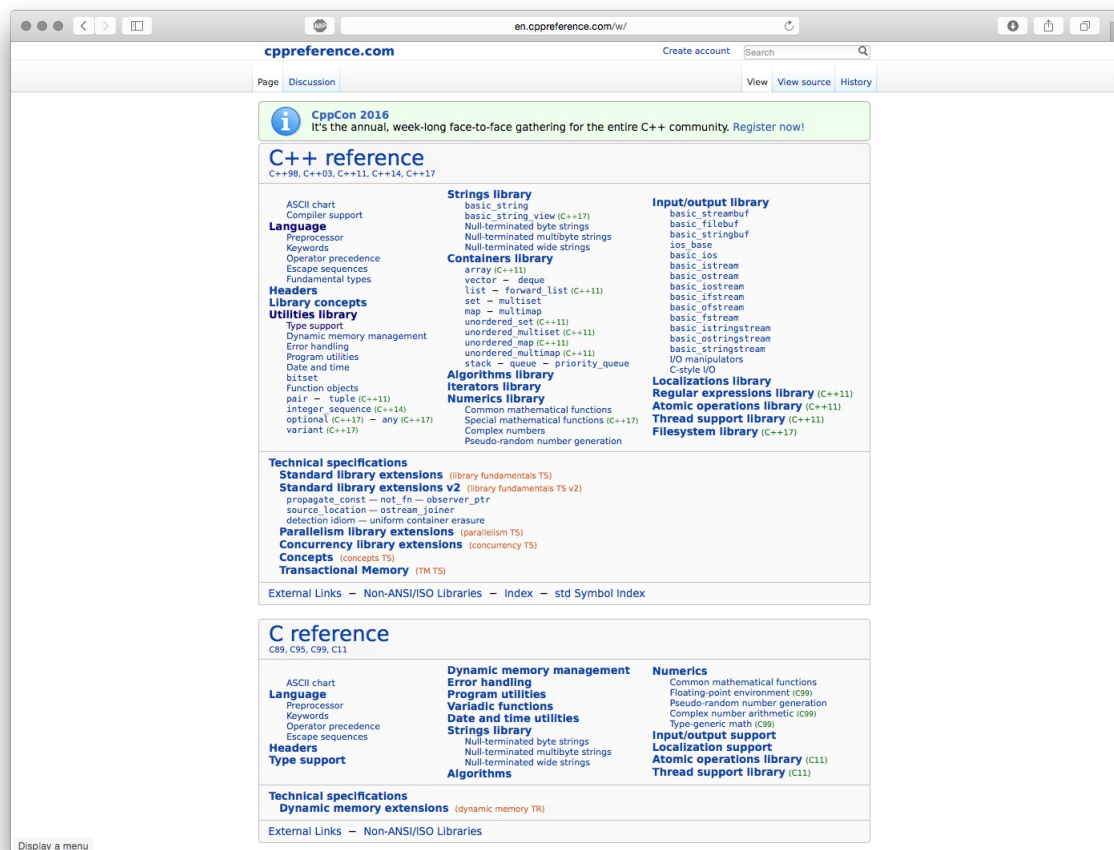
Как отличить хороший учебник от плохого

- Не стоит читать учебники по программированию:
 - из серий «освой за 24 часа...», «для чайников», «для идиотов»
 - устаревшие (за некоторыми исключениями)



Справочная информация по С (и С++)

■ <https://cppreference.com>

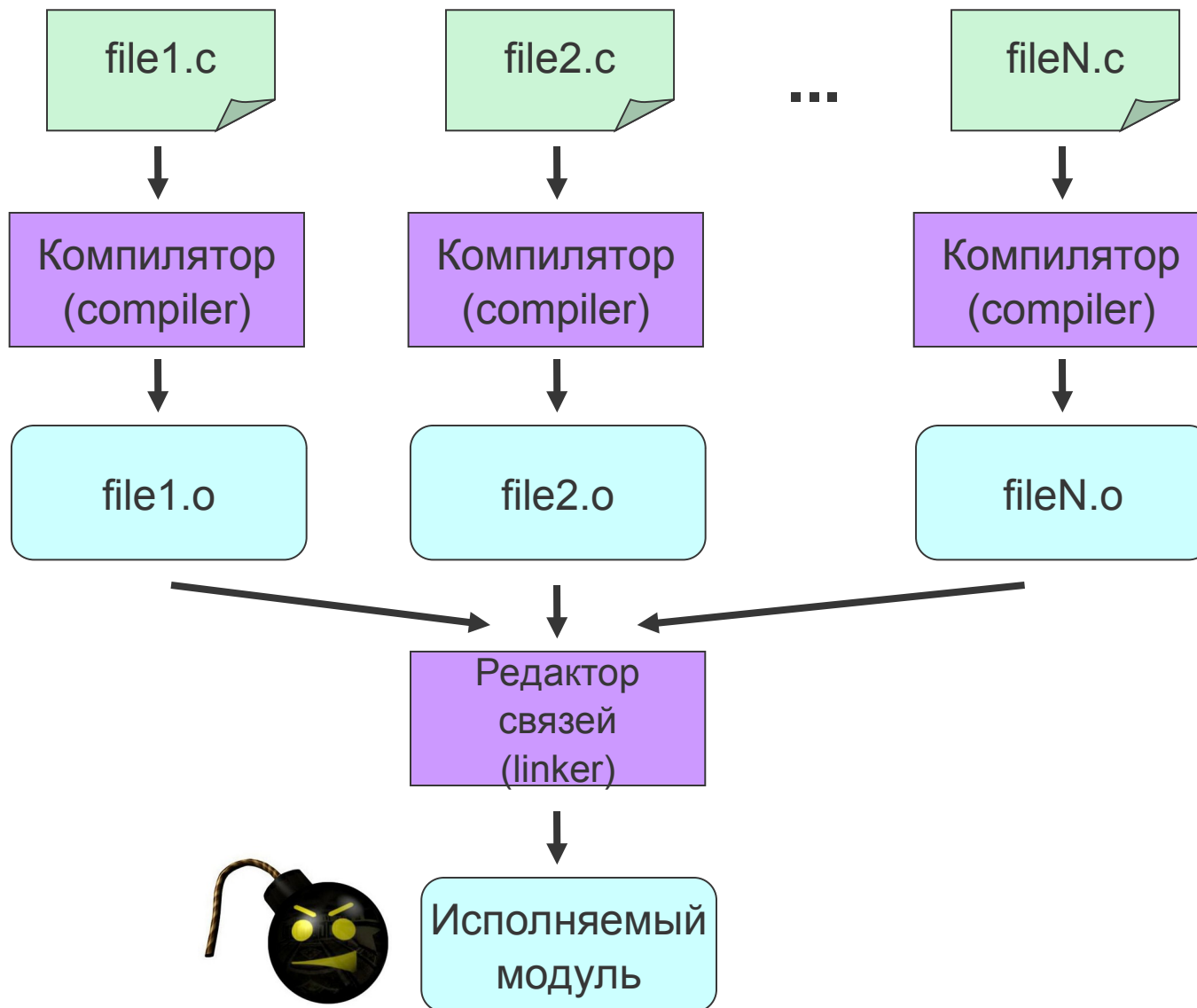


Главный источник информации о C — стандарт языка

- Используется в основном разработчиками компиляторов и теми, кого принято в шутку называть language lawyer'ами
- Трудночитаемый документ, который нелегко использовать даже в качестве справочника

N1570	Committee Draft — April 12, 2011	ISO/IEC 9899:201x
INTERNATIONAL STANDARD	©ISO/IEC	ISO/IEC 9899:201x
Programming languages — C		
<i>ABSTRACT</i>		
(Cover sheet to be provided by ISO Secretariat.)		
<p>This International Standard specifies the form and establishes the interpretation of programs expressed in the programming language C. Its purpose is to promote portability, reliability, maintainability, and efficient execution of C language programs on a variety of computing systems.</p>		
<p>Clauses are included that detail the C language itself and the contents of the C language execution library. Annexes summarize aspects of both of them, and enumerate factors that influence the portability of C programs.</p>		
<p>Although this International Standard is intended to guide knowledgeable C language programmers as well as implementors of C language translation systems, the document itself is not designed to serve as a tutorial.</p>		
<p>Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.</p>		
<p>Changes from the previous draft (N1539) are indicated by “diff marks” in the right margin: deleted text is marked with “*”, new or changed text with “ ”.</p>		

Компиляция программ на С



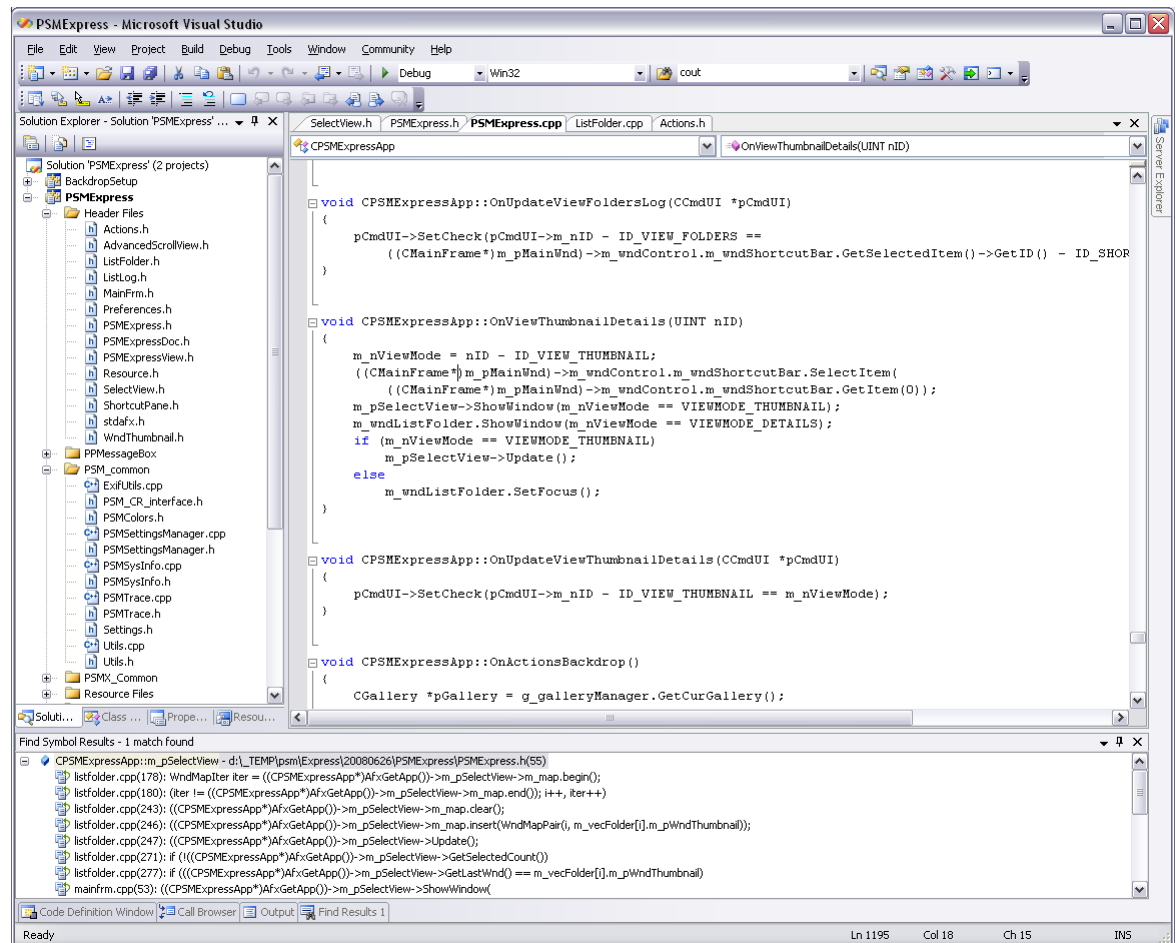
Компиляторы C/C++

- **Работа из командной строки**
 - toolchain
 - gnu (gcc, g++, gdb)
 - llvm (clang, lldb)
 - cl
 - редакторы
 - Notepad++
 - TextWrangler
 - vim, emacs, тысячи их...
- **Интегрированные среды разработки**
 - MS Visual Studio
 - Apple Xcode
 - Code::Blocks
 - QT Creator
- **Онлайн-компиляторы**
 - codepad (<http://codepad.org>)
 - ideone (<http://ideone.com>)

Интегрированные среды разработки

■ Integrated Development Environment (IDE)

- ❑ Eclipse
- ❑ QT Creator
- ❑ NetBeans
- ❑ Visual Studio Code
- ❑ Apple Xcode
- ❑ KDevelop
- ❑ Code::Blocks
- ❑ Anjuta
- ❑ CodeLite
- ❑ C++ Builder
- ❑ ... ТЫСЯЧИ ИХ



Минимальный набор

■ Редактор с подсветкой синтаксиса

- Notepad++ (Win)
- Sublime/Atom/Lite XL/...
- vim/emacs/nano/...
- ... тысячи их!

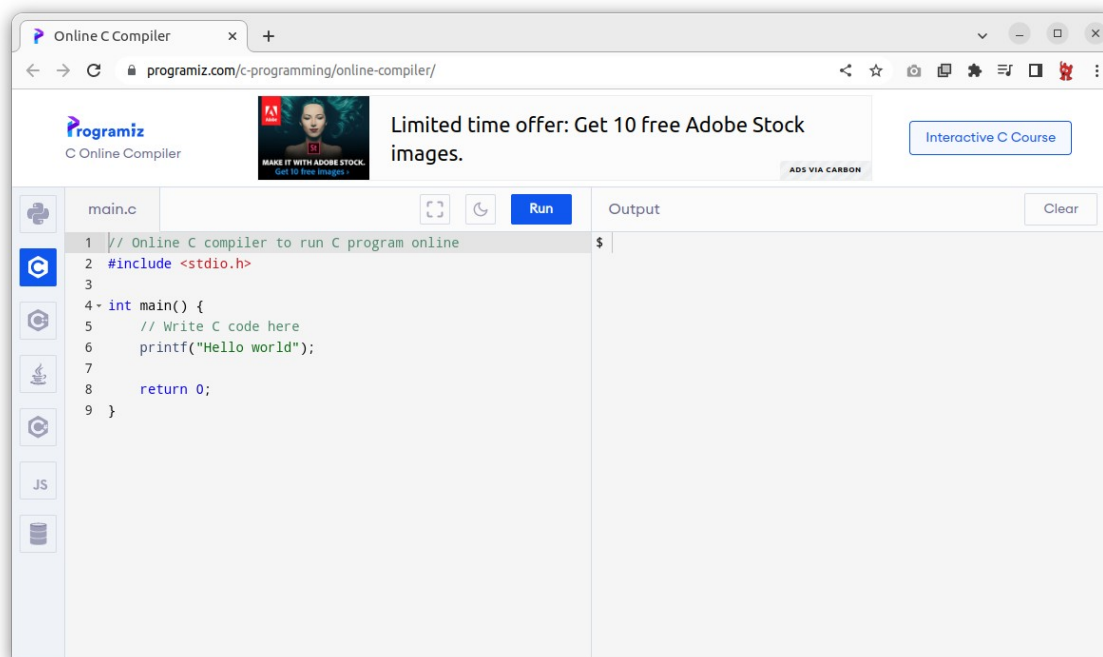
■ Компилятор (и редактор связей)

- gcc
- clang
- ...



Online компиляторы

- <http://codepad.org/>
- <https://www.programiz.com/c-programming/online-compiler/>
- https://www.tutorialspoint.com/compile_c_online.php
- <https://www.jdoodle.com/c-online-compiler/>



Операции в языке C

Ранг	Операции	Ассоциативность
1	() [] . ->	->
2	! ~ + - ++ -- & * sizeof (тип)	<-
3	* / %	->
4	+ -	->
5	>> <<	->
6	> >= < <=	->
7	== !=	->
8	&	->
9	^	->
10		->
11	&&	->
12		->
13	?:	<-
14	= *= /= %= += -= &= ^= = >>= <<=	<-
15	,	->

Операторы в языке C

```
if (...) { } else { }  
while (...) { }  
do { } while (...);  
for (...) { }  
switch (...) { }  
break;  
continue;  
goto;  
return;
```



ФУНКЦИИ И ВЫЗОВ ФУНКЦИЙ

```
// some fake crc
unsigned char crc(unsigned char arr[], size_t len) {
    unsigned char res = 0xff;
    for (int i = 0; i < len; i++) {
        res ^= arr[i];
        if (res & 1) { res >>= 1; res ^= 0xa5; }
    }
    return ~res;
}

...
unsigned char bytes[] = {0xfe, 0x0, 0x11, 0x22, 0x33};
unsigned char c = crc(bytes, sizeof(bytes));
printf("crc = %02hhx\n", c);
```

Несколько практических задач

- получение случайных* чисел в заданном диапазоне
- вывод сообщений на русском
- вывод битового представления числа
- битовые операции
 - узнать значение бита
 - изменить значение бита
 - обменивать биты местами
 - сделать циклический сдвиг
 - подсчитать количество единичных битов в числе
 - оптимизация умножений
 - обмен содержимого двух переменных

Получение случайных чисел в заданном диапазоне

```
#include <stdlib.h>
```

```
#include <time.h>
```

```
srand(time(NULL));
```

```
int    var1 = rand(),  
       var2 = rand() % 20,  
       var3 = rand() & 0xFF;
```



Получение случайных чисел в заданном диапазоне

```
#include <stdlib.h>
#include <time.h>
#include <unistd.h>                // POSIX

srand(time(NULL) + getpid()); // POSIX

int    var1 = rand(),
       var2 = rand() % 20,
       var3 = rand() & 0xFF;
```

Получение случайных чисел в заданном диапазоне

```
#ifdef _WIN32
#include <windows.h>
#define getpid GetCurrentProcessId
#else
#include <unistd.h>
#endif

srand(time(NULL) + getpid());
```

Вывод сообщений на русском

```
#include <locale.h>
```

```
int main() {
```

```
    // не требуется в системах с правильно
```

```
    // сконфигурированной локалью UTF-8
```

```
    setlocale(LC_ALL, "Russian");
```

```
    ...
```

```
    printf("Лец ми спик фром май харт!");
```

```
}
```

Логические и поразрядные операции

Сверхбыстрый экскурс в алгебру логики

Алгебра логики – алгебра над множеством из двух элементов $B = \{0, 1\}$ (или $\{\text{false}, \text{true}\}$)

Над элементами множества определены три элементарных операции:

- отрицание
- дизъюнкция
- конъюнкция

На основе элементарных операций можно построить сколь угодно сложные логические функции (логические связки)

Подробнее об алгебре логики:

http://ru.wikipedia.org/wiki/Алгебра_логики

Некоторые важные операции

- Отрицание («НЕ»)
- Конъюнкция (логическое «И»)
- Дизъюнкция (логическое «ИЛИ»)
- Исключающее ИЛИ («либо то, либо другое, но не оба вместе»)
- Эквиваленция («то же самое»)
- Импликация («если, то»)
- Стрелка Пирса («антидизъюнкция»)
- Штрих Шеффера («антиконъюнкция»)

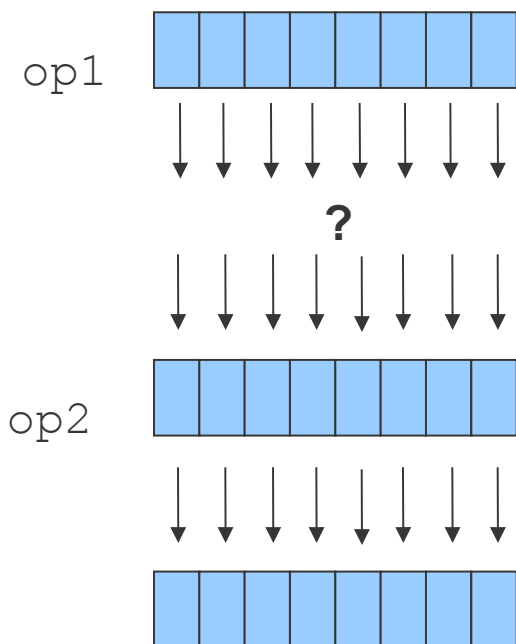
Соответствие двоичных, восьмеричных и шестнадцатеричных чисел

dec	bin	oct	hex
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

Логические и поразрядные операции в С

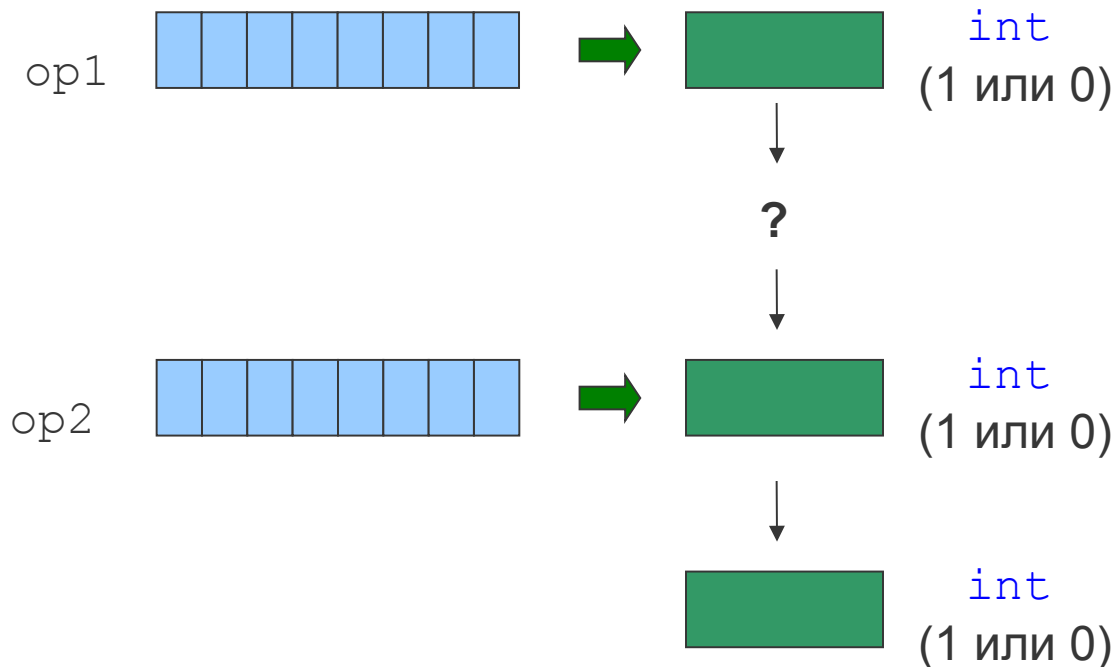
поразрядные операции

op1 ? op2



логические операции

op1 ? op2



Побитовые операции

x	y	~x (NOT)	x y (OR)	x & y (AND)	x ^ y (XOR)
0	0	1	0	0	0
0	1	1	1	0	1
1	0	0	1	0	1
1	1	0	1	1	0

Логические операции

x	y	!x (NOT)	x y (OR)	x & y (AND)
false	false	true	false	false
false	true	true	true	false
true	false	false	true	false
true	true	false	true	true

Логические операции

x	y	!x (NOT)	x y (OR)	x & y (AND)
0	0	1	0	0
0	1	1	1	0
1	0	0	1	0
1	1	0	1	1

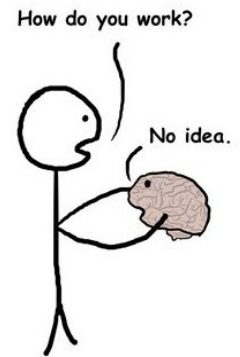
Побитовые и логические операции

```
int x = 0x5a, y = 0xa5;
```

```
int result1 = x & y;
```

```
int result2 = x && y;
```

```
printf("%d %d", result1, result2);
```



Проверка значения бита

```
unsigned int x = 42;
```

```
...
```

```
if (x & 1) { // Нулевой бит
```

```
...
```

```
}
```

```
if (x & (1 << n)) { // N-ный бит
```

```
...
```

```
}
```

Установка и сброс бита

```
unsigned int x = 42;
```

```
...
```

```
x = x | (1 << n);           // Установка
```

```
x |= 1 << n;
```

```
...
```

```
x = x & ~(1 << n);         // Сброс
```

```
x &= ~(1 << n);
```


Инвертирование бита

```
unsigned int x = 42;
```

```
...
```

```
x = x ^ (1 << n);
```

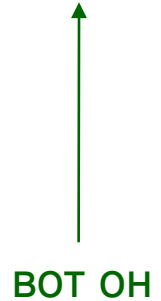
```
x ^= 1 << n;
```

Задача: сброс самого правого единичного бита

```
unsigned int x = 42;
```

```
// 0010 0110
```

```
...
```



BOT ON

Сброс самого правого единичного бита

```
unsigned int x = 42;           // 0010 0110
```

```
...
```

```
unsigned int m = 1;
```

```
if (x) {
```

```
    while (! (x & m) ) m << 1;
```

```
    x &= ~m;
```

```
}
```

```
// в коде есть ошибка?
```

Сброс самого правого единичного бита

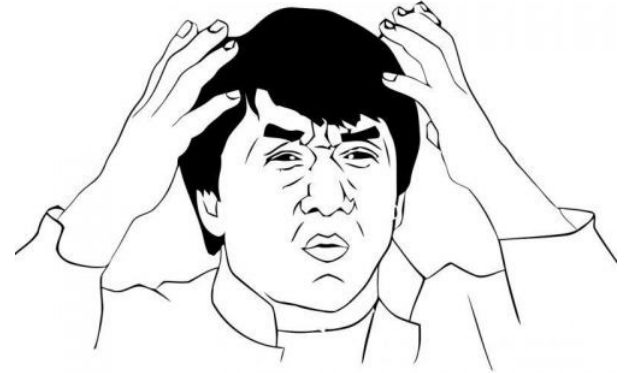
```
unsigned int x = 42;           // 0010 0110
...
unsigned int m = 1;
if (x) {
    while (!(x & m)) m <<= 1; // Теперь верно
    x &= ~m;
}
```

Сброс самого правого единичного бита

```
unsigned int x = 42;
```

```
...
```

```
x = x & (x - 1);
```



Обмен значений двух битов

```
unsigned int x = 42;
```

```
unsigned int n1 = 3, n2 = 5;
```

Обмен значений двух битов

```
unsigned int x = 42;  
unsigned int n1 = 3, n2 = 5;  
  
if ( (x & (1 << n1)) != (x & (1 << n2)) ) {  
    x ^= 1 << n1;  
    x ^= 1 << n2;  
}
```

// В коде есть ошибка?

Обмен значений двух битов

```
unsigned int x = 42;  
unsigned int n1 = 3, n2 = 5;  
  
if ((bool)(x & (1 << n1)) != (bool)(x & (1 << n2))) {  
    x ^= 1 << n1;  
    x ^= 1 << n2;  
}
```


Циклический сдвиг вправо

```
unsigned int x = 42,  
    n = 3,  
    t;  
  
for (int i = 0; i < n; ++i) {  
    t = x & 1;  
    x >>= 1;  
    x |= t << 31;  
}
```

Циклический сдвиг влево

```
unsigned int x = 42,  
    n = 3,  
    t;  
  
for (int i = 0; i < n; ++i) {  
    t = x & 0x80000000;  
    x <<= 1;  
    x |= t;  
}
```

// в коде есть ошибка?

Циклический сдвиг влево

```
unsigned int x = 42,  
    n = 3,  
    t;  
  
for (int i = 0; i < n; ++i) {  
    t = (bool) (x & 0x80000000);           // Теперь верно  
    x <<= 1;  
    x |= t;  
}
```

Циклический сдвиг без цикла

```
unsigned int x = 42,  
    n = 3,  
    right = 1;
```

Циклический сдвиг без цикла

```
unsigned int x = 42,  
    n = 3,  
    right = 1;
```

```
unsigned int mask = right ?  
    (1 << n) - 1 :  
    ~((1 << (31-n+1)) - 1);
```

```
unsigned int temp = x & mask;
```

```
x = right ? x >> n : x << n;
```

```
temp = right ? temp << (32 - n) : temp >> (32 - n);
```

```
x |= temp;
```

```
// в коде есть ошибка?
```

Циклический сдвиг без цикла

```
unsigned int x = 42,  
    n = 300,  
    right = 1;
```

```
n %= 32;                                // Так лучше
```

```
unsigned int mask = right ?  
    (1 << n) - 1 :  
    ~((1 << (31-n+1)) - 1);
```

```
unsigned int temp = x & mask;
```

```
x = right ? x >> n : x << n;
```

```
temp = right ? temp << (32 - n) : temp >> (32 - n);
```

```
x |= temp;
```

Подсчет количества единичных битов

```
unsigned int x = 42;
```

```
int count = 0;
```

Подсчет количества единичных битов

```
unsigned int x = 42;  
int count = 0;  
  
for (int i = 0, m = 1; i < 32; ++i, m <<= 1)  
    count += (bool) (x & m);  
  
printf("%d", count);
```


Подсчет количества единичных битов

```
unsigned int x = 42;
```

```
int count = 0;
```

```
// Оптимизированный (?) вариант
```

```
while (x) {
```

```
    count++;
```

```
    x = x & (x - 1);           // !
```

```
}
```

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

Подсчет количества единичных битов

```
unsigned char x = 42;           // в байте
```

```
int count = (bool)(x & 1)  +  
            (bool)(x & 2)  +  
            (bool)(x & 4)  +  
            (bool)(x & 8)  +  
            (bool)(x & 16) +  
            (bool)(x & 32) +  
            (bool)(x & 64) +  
            (bool)(x & 128);
```

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

Вывод на экран разрядов числа

- Выводить цифры, начиная со старшего разряда

```
unsigned int x = 42;
```

```
for (int i = 0; i < 32; ++i) {  
    printf("%d", (bool) (x & 0x80000000));  
    x <<= 1;  
}
```

Вывод на экран разрядов числа

- Разделять байты на экране пробелами

```
unsigned int x = 42;
```

```
for (int i = 0; i < 32; ++i) {  
    if (i && !(i % 8)) printf(" ");  
    printf("%d", (bool) (x & 0x80000000));  
    x <<= 1;  
}
```

Проблема с разрядностью

- что, если количество разрядов в числе $\neq 32$?

```
unsigned short x = 42;
```

```
for (int i = 0; i < sizeof(x)*8; ++i) {  
    if (i && !(i % 8)) printf(" ");  
    printf("%d",  
        (bool) (x & (1L << (sizeof(x)*8 - 1))));  
    x <<= 1;  
}
```

Обмен значений двух переменных

- традиционный способ

```
int a = 10, b = 20;
```

```
int temp = a;
```

```
a = b;
```

```
b = temp;
```

Обмен значений двух переменных

■ «хакерский» способ

```
int a = 10, b = 20;
```

```
a ^= b;
```

```
b ^= a;
```

```
a ^= b;
```

// или даже так:

```
a ^= b ^= a ^= b;
```



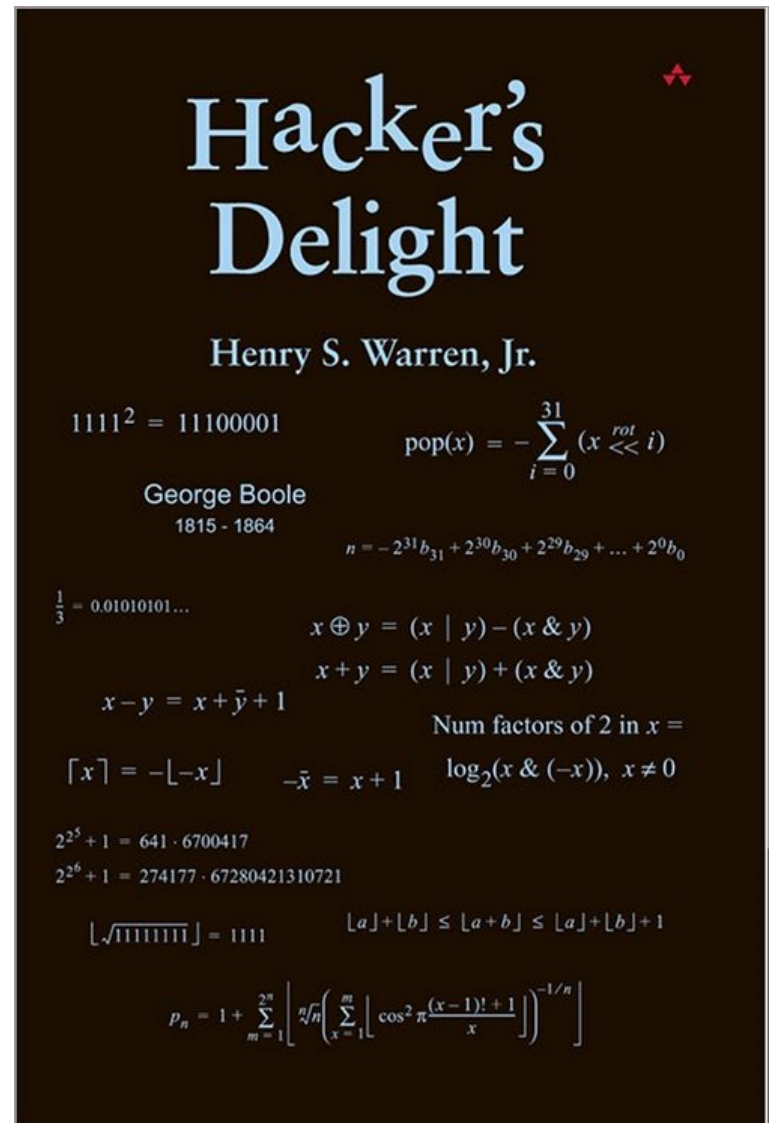
Just for fun

- Будет ли скомпилирована эта программа? Если да, что будет выведено на экран?

```
int a = ~~~~0, b = ~~~~0;  
printf("a = %d, b = %d", a, b);
```


Наслаждение для хакера

- *Henry S. Warren*
Hacker's Delight
 - <http://www.hackersdelight.org/>
- *Генри Уоррен*
Алгоритмические трюки
для программистов



Ссылки

- Bit Twiddling Hacks
 - <https://catonmat.net/low-level-bit-hacks>
 - <http://graphics.stanford.edu/~seander/bithacks.html>

Резюме

- В С, в отличие от некоторых других ЯВУ, поразрядные операции «дешевы», так как непосредственно переводятся в соответствующие инструкции процессора
- Для того, чтобы выполнить поразрядные операции с числом, не нужно использовать массивы



Задание к следующей лекции

- Макарова, Волков. Информатика. Учебник для вузов.
 - гл. 14, 19
- Керниган, Ритчи. Язык программирования С.
 - гл. 1 – 5
- Савельев, ПТЦА
 - гл. 10

