



лекция 3

ПРОСТЫЕ ТИПЫ ДАННЫХ ЯЗЫКА C

План лекции

- Простые типы данных
 - Ограничения на простые типы данных
- Машинное представление простых типов данных
- Общая структура программы на Си
- Объявление переменных простых типов

Простые типы данных

- *Тип данных* – это пара, состоящая из множества значений и набора операций над ними
- Языки программирования позволяют строить одни типы данных из других типов данных
- *Простые* типы данных – это типы данных, которые нельзя построить из других типов данных
- *Составные* типы данных – это типы данных, которые строятся из других типов данных

Простые типы данных Си

- Символы, 8-битовые целые
- Целые
- Числа с плавающей точкой
- Перечислимые типы

Простые типы данных -- СИМВОЛЫ

- C89
 - спецификатор-символьного-типа ::= [signed|unsigned] char
 - Символы и 8-битовые целые со знаком (signed) или без знака (unsigned)
- CHAR_MIN, CHAR_MAX, UCHAR_MAX и др. в limits.h
- Стандарт не определяет, есть ли знак у значений типа char

Простые типы данных -- целые

- C89
 - спецификатор-целого-типа ::= [signed|unsigned] [short|long] int
- C99, C11 (поддержка есть в gcc 4.6)
 - спецификатор-целого-типа ::= [signed|unsigned] [short|long **long**] int
- C89/C99/C11 не определяют, есть ли знак у int
 - Все известные компиляторы считают int = signed int
- Нестандартные целые типы
 - __int16, __int32, __int64, __int128
 - Наличие и смысл зависят от компилятора

Простые типы данных -- целые

Варианты имени	Диапазон значений в limits.h
[signed] short [int]	SHRT_MIN ... SHRT_MAX
unsigned short [int]	0 ... USHRT_MAX
int signed [int]	INT_MIN ... INT_MAX
unsigned [int]	0 ... UINT_MAX
[signed] long [int]	LONG_MIN ... LONG_MAX
unsigned long [int]	0 ... ULONG_MAX
[signed] long long [int]	LLONG_MIN ... LLONG_MAX
unsigned long long [int]	0 ... ULLONG_MAX

sizeof(char) == sizeof(unsigned char) <=

<= sizeof(short) == sizeof(unsigned short) <=

<= sizeof(int) == sizeof(unsigned) <=

<= sizeof(long) == sizeof(unsigned long) <=

<= sizeof(long long) == sizeof(unsigned long long)

Простые типы данных – числа с плавающей точкой

- C89/C99/C11
 - спецификатор-типа-с-плавающей ::=
float | [long] double
- `sizeof(float) <= sizeof(double) <= sizeof(long double)`
- `FLT_MIN`, `FLT_MAX`, `DBL_MIN`, `DBL_MAX`, `LDBL_MIN`, `LDBL_MAX` и др. в файле `float.h`

Простые типы данных – перечислимые типы

- C89/C99/C11
 - enum-спецификатор ::=
 - | 'enum' [имя] '{' список-перечислителей '}'
 - | 'enum' [имя] '{' список-перечислителей ',' '}'
 - | 'enum' имя
 - список-перечислителей ::= перечислитель
| список-перечислителей ',' перечислитель
 - перечислитель ::= перечислимая-константа
| перечислимая-константа '='
константное-выражение
 - перечислимая-константа ::= имя
 - *константное-выражение на след. лекции*
- Тип, диапазон значений и размер в памяти такие же, как у int

Простые типы данных – перечислимые типы

- Примеры

- `enum my_boolean_t { my_false = 0, my_true = 1 }`

- `enum my_boolean_t { my_false, my_true }`

- `my_false = 0`

- `my_true = my_false + 1 = 1`

- `enum my_boolean_t { my_false = 0, my_true = 0 }`

- `my_false = my_true = 0`

- `enum my_day_t { mon, tue, wed, thu, fri, sat, sun }`



Машинное представление данных простых типов

- Символы, 8-битовые целые
 - Целые
 - Числа с плавающей точкой
- 

Машинное представление значений типа char, signed char, unsigned char, 1 байт памяти,

- signed char целые числа от -128 до 127
- unsigned char целые числа от 0 до 255
- Программы на Си используют значения типов char, signed char, unsigned char для печати текстовых сообщений на экране, бумаге и т.п.
- Соответствие значений и символов определяется кодировкой ОС

Машинное представление значений типа char, signed char, unsigned char

Кодировка CP866 (MS DOS)

	00	10	20	30	40	50	60	70
0		▸		0	@	P	'	p
1	☒	◀	!	1	A	Q	a	q
2	☒	‡	"	2	B	R	b	r
3	♥	!!	#	3	C	S	c	s
4	♦	¶	\$	4	D	T	d	t
5	♣	§	%	5	E	U	e	u
6	♠	-	&	6	F	V	f	v
7	•	±	'	7	G	W	g	w
8	☐	↑	(8	H	X	h	x
9	o	↓)	9	I	Y	i	y
A	☒	→	*	:	J	Z	j	z
B	♂	←	+	;	K	[k	{
C	♀	└	,	<	L	\	l	
D	℞	↕	-	=	M]	m	}
E	℞	▲	.	>	N	^	n	~
F	※	▼	/	?	O	_	o	△

	80	90	A0	B0	C0	D0	E0	F0
0	А	Р	а	▯	└	┘	р	≡
1	Б	С	б	▯	└	┘	с	±
2	В	Т	в	▯	└	┘	т	>
3	Г	У	г		└	┘	у	<
4	Д	Ф	д	└	-	┘	ф	┌
5	Е	Х	е	└	+	┘	х	└
6	Ж	Ц	ж	└	└	┘	ц	÷
7	З	Ч	з	└	└	┘	ч	≈
8	И	Ш	и	└	└	┘	ш	°
9	Й	Щ	й	└	└	┘	щ	·
A	К	Ь	к	└	└	┘	ь	·
B	Л	Ы	л	└	└	┘	ы	√
C	М	Ъ	м	└	└	┘	ъ	π
D	Н	Э	н	└	=	┘	э	2
E	О	Ю	о	└	└	┘	ю	●
F	П	Я	п	└	└	┘	я	

Машинное представление значений типа char, signed char, unsigned char (КОИ-8)

Если в тексте в КОИ-8 убрать восьмой бит каждого символа, то получается текст, подобный транслиту. Например, «Русский Текст» --> «rUSSKIJ tEKST».

- Win 1251

- Mac OS

Code	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
80	:																
90	:																
A0	:			Г:г													
B0	:																
C0	:	ю	а	б	ц	д	е	ф	з	х	и	й	к	л	м	н	о
D0	:	п	я	р	с	т	ч	ш	щ	ъ	ы	ь	э	ю	я		
E0	:	Ю	А	Б	Ц	Д	Е	Ф	З	Х	И	Й	К	Л	М	Н	О
F0	:	П	Я	Р	С	Т	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		

Code	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
80	:																
90	:																
A0	:								Ё								
B0	:								ё								
C0	:	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
D0	:	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
E0	:	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F0	:	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я

Code	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
80	:	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
90	:	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
A0	:																
B0	:																
C0	:																
D0	:													Ё	ё	я	
E0	:	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F0	:	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	

Машинное представление беззнаковых (unsigned) целых

- *Двоичная запись* числа $Ч$ -- набор $b_n \dots b_1 b_0$ такой, что $Ч = b_0 \cdot 2^0 + b_1 \cdot 2^1 + \dots + b_n \cdot 2^n$
- М.П. unsigned числа x – это *двоичная запись* числа $x \bmod 2^{8 \cdot \text{sizeof}(T)}$

Машинное представление целых со знаком (signed)

- М.П. signed числа x
 - двоичная запись $x \bmod 2^{8 \cdot \text{sizeof}(T)}$, если $x \geq 0$
 - *дополнительный код* $|x|$ -- двоичная запись $2^{8 \cdot \text{sizeof}(T)} - |x| \bmod 2^{8 \cdot \text{sizeof}(T)}$, если $x < 0$
- Свойство дополнительного кода
 - Вычисление на компьютере $\text{М.П.}(x) + \text{М.П.}(-x)$ дает $\text{М.П.}(0)$

Машинное представление целых со знаком (signed)

- Построение дополнительного кода $|x|$
- $b[n]$ – двоичная запись $|x|$
- $d[n]$ – дополнительный код $|x|$
- Алгоритм

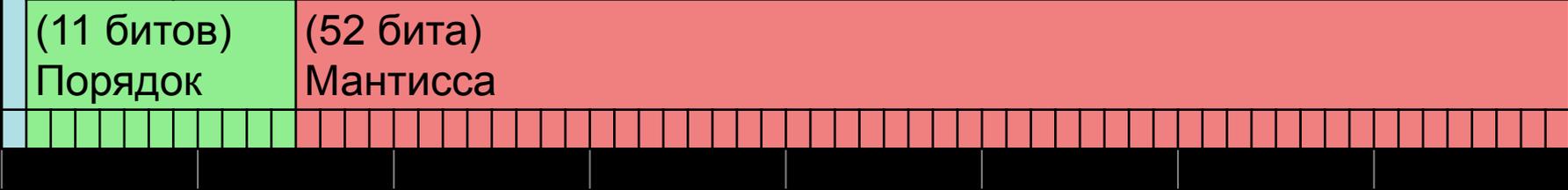
```
for (i = 0; i < n; i = i+1) d[i] = 1-b[i];
for (i = 0; i < n && d[i] == 1; i = i+1) d[i] = 0;
if (i < n) d[i] = 1;
```

Машинное представление чисел с плавающей точкой

- Числа вида $S \cdot M \cdot 2^P$
- S – знак +1 или -1, 1 бит
- M – *мантисса*, $x/2^{mb}$ от 0 до 1
 - mb – число битов в мантиссе
 - Intel, AMD, ARM -- 23 для float, 52 для double
 - x – целое от 0 до $2^{mb}-1$
- P – порядок
 - Intel, AMD, ARM – 8 битов для float, 11 битов для double
- float занимает $1+8+23 = 32$ бита
- double занимает $1+11+52 = 64$ бита
- long double обычно совпадает с double или эмулируется

Машинное представление значений типа double – стандарт

IEEE 754



Порядок	Мантисса 0	Мантисса != 0	Формула
0x000	0 и -0	Денормализов. числа	$(-1)^{\text{знак}} \cdot 2^{\text{порядок}-1022} \cdot (0.\text{мантисса})_{(2)}$
0x001 ... 0x7fe	Нормализованные числа		$(-1)^{\text{знак}} \cdot 2^{\text{порядок}-1023} \cdot (1.\text{мантисса})_{(2)}$
0x7ff	$+\infty$ или $-\infty$	NaN	

$$3\text{ff}0\ 0000\ 0000\ 0000_{(16)} = 1$$

$$0000\ 0000\ 0000\ 0000_{(16)} = 0$$

$$7\text{ff}0\ 0000\ 0000\ 0000_{(16)} = \infty$$

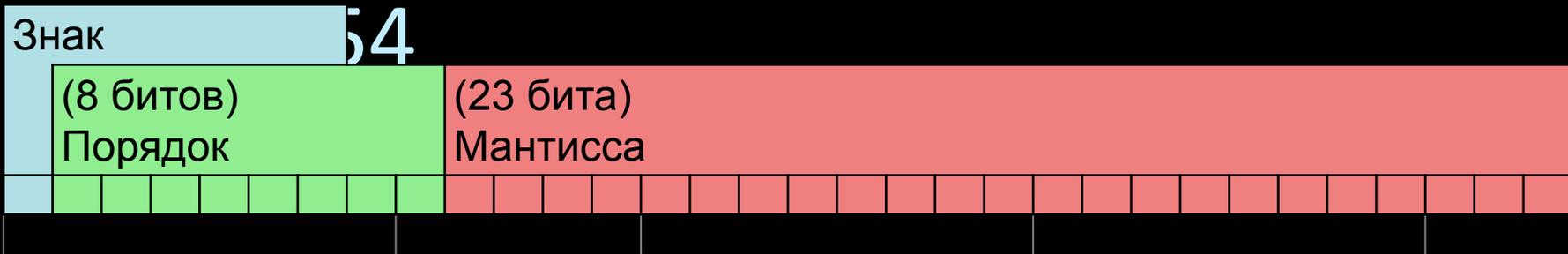
$$3\text{ff}0\ 0000\ 0000\ 0001_{(16)} \approx 1.0000000000000002$$

$$8000\ 0000\ 0000\ 0000_{(16)} = -0$$

$$\text{ff}f0\ 0000\ 0000\ 0000_{(16)} = -\infty$$

$$3\text{fd}5\ 5555\ 5555\ 5555_{(16)} \approx 1/3$$

Машинное представление значений типа float – стандарт



Порядок	Мантисса 0	Мантисса != 0	Формула
0x00	0 и -0	Денормализов. числа	$(-1)^{\text{знак}} \cdot 2^{\text{порядок}-126} \cdot (0.\text{мантисса})_{(2)}$
0x01 ... 0xfe	Нормализованные числа		$(-1)^{\text{знак}} \cdot 2^{\text{порядок}-127} \cdot (1.\text{мантисса})_{(2)}$
0xff	$+\infty$ или $-\infty$	NaN	

Машинное представление данных простых типов -- разное

- Адрес значения переменной простого типа B *выровнен* (кратен) $\text{sizeof}(B)$

Общая структура программы на Си

Для РБНФ $\langle X \rangle$ обозначим $\langle X \rangle^*$ РБНФ $\langle \text{список } X \rangle$, заданную правилом

$$\langle \text{список } X \rangle ::= \langle X \rangle \mid \langle \text{список } X \rangle \langle X \rangle$$

Общая структура программы на Си

<единица-трансляции> ::=
 <внешнее-объявление>*

<внешнее-объявление> ::=
 <определение-функции> | <объявление>

<определение-функции> ::=
 [<спецификаторы-объявления>] <объявитель>
 [<список-объявлений>] <составная-инструкция>

<объявление> ::=
 <простое-объявление> | <составное-объявление>

Общая структура программы на Си

<инструкция> ::=

- | <помеченная-инструкция>
- | <инструкция-выражение>
- | <составная-инструкция>
- | <инструкция-выбора>
- | <циклическая-инструкция>
- | <инструкция-перехода>

<инструкция-выражение> ::= [<выражение>] ';' ;

<составная-инструкция> ::=

'{' [<объявление>*] [<инструкция>*] '}'

Объявление и инициализация переменных простых типов

<простое-объявление> ::=

<спецификаторы-объявления>

[<простой-объявитель-инициализатор>*]

Объявления переменных встречаются либо вне самого внешнего блока { }, либо сразу же после {

Объявление и инициализация переменных простых типов

<простой-объявитель-инициализатор> ::=
 <простой-объявитель>
| <простой-объявитель> '=' <инициализатор>

<простой-объявитель> ::= <идентификатор>

<инициализатор> ::= <выражение-присваивания>

Объявление и инициализация переменных простых типов

<спецификаторы-объявления> ::=

(

| <спецификатор-класса-памяти>

| <спецификатор-простого-типа>

| <квалификатор-типа>

)*

Объявление и инициализация переменных простых типов

<спецификатор-класса-памяти> ::=

| 'auto'
| 'register'
| 'static'
| 'extern'
| 'typedef'

- auto
 - На стеке (по умолчанию)
- register
 - В регистре
- static
 - В статической памяти единицы компиляции
- extern
 - В статической памяти программы
- typedef
 - Вне памяти, объявляемый идентификатор далее обозначает тип

Объявление и инициализация переменных простых типов

<спецификатор-простого-типа> ::=
'void' | 'char' | 'short' | 'int' | 'long' | 'float'
| 'double' | 'signed' | 'unsigned'
| <спецификатор-enum> -- было
| <typedef-имя>

<typedef-имя> ::= <идентификатор>

<квалификатор-типа> ::= 'const' | 'volatile'

- const
 - Неизменяемое значение
- volatile
 - Значение может асинхронно изменяться – например, в многопоточной программе

Примеры объявлений переменных простых типов

- `int x;`
- `auto int x;` // то же, что выше
- `const int x;` // как задать начальное // значение?!
- `const double x = 1.234567;`
- `float x = 0, y = x+1;`
- `static int x = 5;`
- `extern unsigned long long global_uuid;`

Примеры объявлений переменных простых типов

- `typedef int my_int; // my_int – синоним int`
`my_int x = 0, y = x+1;`

Заключение

- Простые типы данных
- Ограничения на простые типы данных
- Машинное представление простых типов данных
 - Представление целых и вещественных чисел
- Объявление и инициализация переменных простых типов