

Программирование на языке C++

Лекция 8

Строки в стиле Си
C-style strings

Символьный тип

Тип char

Тип `char` предназначен для хранения символов

Из-за ряда сложностей символы не хранятся в памяти в явном виде, они хранятся в виде целых чисел.

Число которое хранит `char` — это **код** символа в некотором наборе символов

Размер `char` — 1 байт (256 значений)

Одним из самых распространённых наборов символов является ASCII

Таблица ASCII

ASCII - American Standard Code for Information Interchange
Американский Стандартный Код для Обмена Информацией

Коды с 0 по 31 – непечатные (служебные) символы

Коды с 32 по 127 – печатные символы

Стандартный набор символов ASCII использует только 7 бит для каждого символа. Добавление 8-го разряда позволяет увеличить количество кодов таблицы ASCII до 255.

Коды от 128 до 255 представляют собой расширение таблицы ASCII. Эти коды используются для кодирования символов национальных алфавитов, а также символов псевдографики, которые можно использовать, например, для оформления в тексте различных рамок и текстовых таблиц

Часть ASCII таблицы

Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ
0	NUL	16	DLE	32	(space)	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	\$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	
13	CR	29	GS	45	-	61	=	77	M	93]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL (delete)

Инициализация символьной переменной

// инициализация кодом 97

```
char ch1 = 97;
```

// инициализация символом (код 97)

```
char ch2 = 'a';
```

// инициализация символом новой строки

```
char ch3 = '\n';
```

Вывод символов на экран

```
#include <iostream>
```

```
int main(){
```

```
    char ch = 97;
```

```
    char ch2 = 'b';
```

```
    // На экран выведется: ab
```

```
    std::cout << ch << ch2;
```

```
    return 0;
```

```
}
```

Преобразование символа в число и наоборот

```
#include <iostream>
```

```
int main(){
```

```
    // На экран выведется 97
```

```
    std::cout << 97;
```

```
    // На экран выведется символ a
```

```
    std::cout << (char) 97;
```

```
    // На экран выведется 97
```

```
    std::cout << (int) 'a';
```

```
    return 0;
```

```
}
```


Операторы преобразования типов

Преобразование в стиле Си

```
(тип_в_который_преобразуем) то_что_преобразуем;  
(char) 97;    или    char(97);  
(int) 'a';    или    int('a');
```

Преобразование в стиле C++

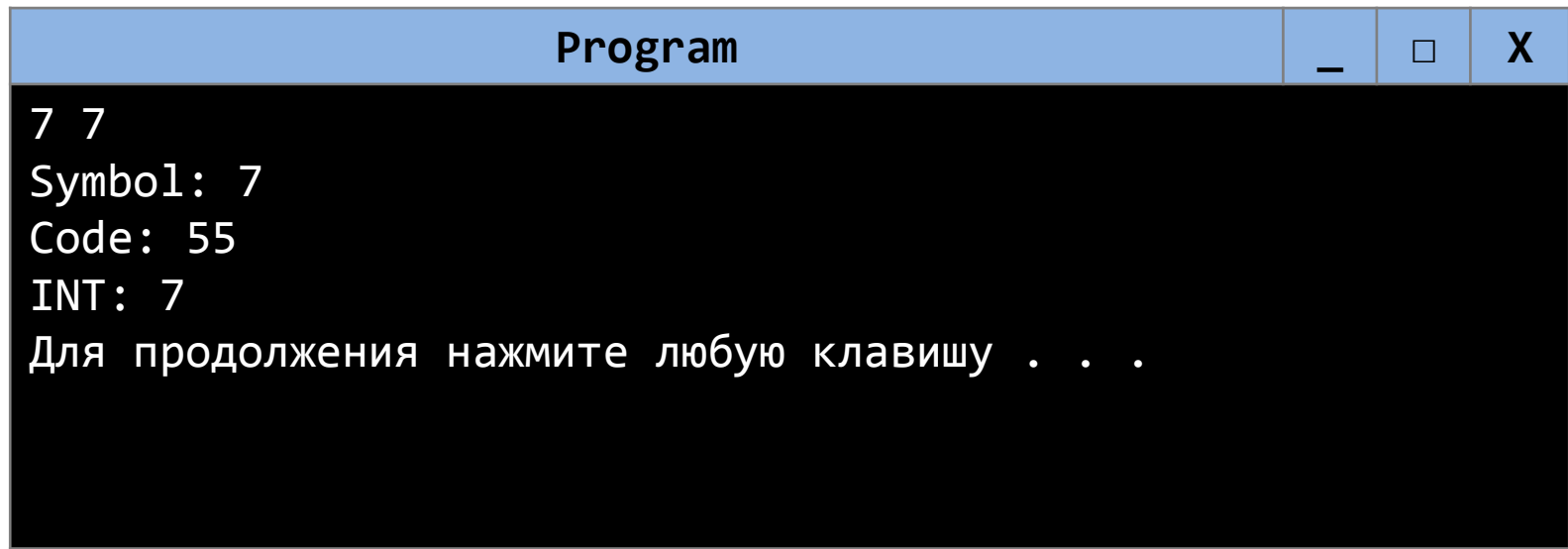
```
static_cast<тип_в>(то_что)  
static_cast<char>(97);  
static_cast<int>('a');
```

Чтение символов

```
#include <iostream>
```

```
int main(){  
    char ch;  
    int i;  
    // Читаем  
    std::cin >> ch >> i;  
    // Смотрим что прочитали  
    std::cout << "Symbol: " << ch << '\n';  
    std::cout << "Code: " << (int)ch << '\n';  
    std::cout << "INT: " << i << '\n';  
    return 0;  
}
```

Чтение символов



```
Program
7 7
Symbol: 7
Code: 55
INT: 7
Для продолжения нажмите любую клавишу . . .
```

Эскейп-последовательности

Имя	Символ	Значение
Alert	\a	Звуковой сигнал
Backspace	\b	Перемещает курсор на символ назад
Newline	\n	Перемещает курсор на след. строку
Carriage return	\r	Перемещает курсор в начало строки
Horizontal tab	\t	Горизонтальная табуляция
Vertical tab	\v	Вертикальная табуляция
Single quote	\'	Одинарная кавычка
Double quote	\"	Двойная кавычка
Backslash	\\	Бэкслеш
Question mark	\?	Вопросительный знак
Octal number	\(number)	Восьмеричная запись символа
Hex number	\x(number)	Шестнадцатеричная запись символа

Применение эскейп-последовательностей

```
#include <iostream>
```

```
int main(){
```

```
    std::cout << '\\'; \\ \
```

```
    std::cout << '\''; \\ '
```

```
    std::cout << '\"'; \\ "
```

```
    return 0;
```

```
}
```

char как целое число

```
#include <iostream>
```

```
int main() {
```

```
    char ch1 = 'a'+1; // к коду 'a' прибавится 1  
    std::cout << ch1 << '\n'; // b
```

```
    std::cout << '2'+'2' << '\n'; // 100
```

```
    // Сравниваются коды символов
```

```
    if('$' < ')') std::cout << "true";
```

```
    return 0;
```

```
}
```

Строки

Что такое Си строка

Строка в стиле Си – символьный массив
последний элемент которого нуль-терминатор.

Если нуль-терминатор отсутствует, то
символьный массив не является строкой

Нуль-терминатор или нулевой символ

Запись в виде символа: `'\0'`

Запись в виде кода: `0`

Объявление и инициализация строки

```
// Для строки важно '\0'
```

```
char str[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

```
// Это не строка
```

```
char str2[5] = {'H', 'e', 'l', 'l', 'o'};
```

```
// Размер myString 7 элементов
```

```
char myString[] = "string";
```

```
// {'A', 'l', 'e', 'x', '\0', '\0', '\0', '\0', '\0', '\0'};
```

```
char name[10] = "Alex";
```

Вывод строки

```
#include <iostream>
```

```
int main()
```

```
{
```

```
    char name[20] = "Alex";
```

```
    std::cout << "My name is: " << name << '\n';
```

```
    return 0;
```

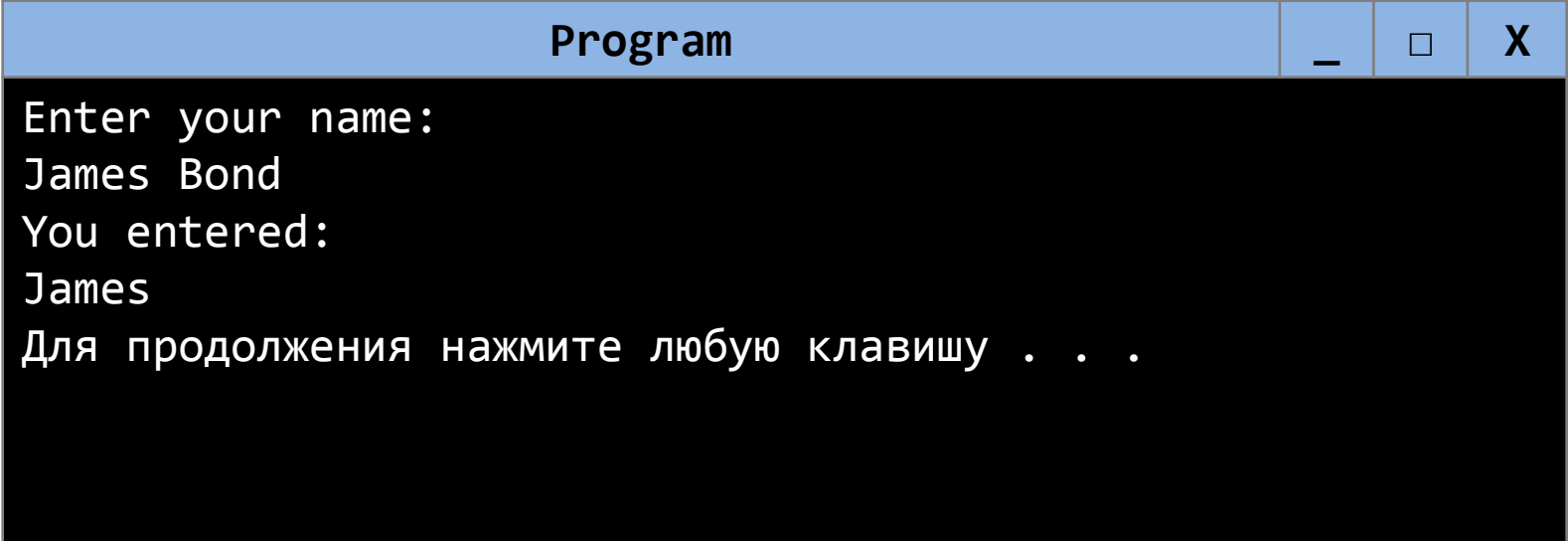
```
}
```

Ввод строки I

```
#include <iostream>
int main()
{
    char name[255]; // буфер
    std::cout << "Enter your name: ";
    std::cin >> name;
    std::cout << "You entered: " << name << '\n';

    return 0;
}
```

Ввод строки I



The image shows a screenshot of a console application window. The window has a title bar with the text "Program" and standard Windows window controls (minimize, maximize, close). The main area of the window is black with white text. The text displayed is as follows:

```
Enter your name:  
James Bond  
You entered:  
James  
Для продолжения нажмите любую клавишу . . .
```

Ввод строки | Через `cin`

`cin` читает до пробельных символов (пробел, табуляция, перевод строки, ...) и оставляет их в потоке.

Нуль-терминатор автоматически добавляется в строку после считанных символов.

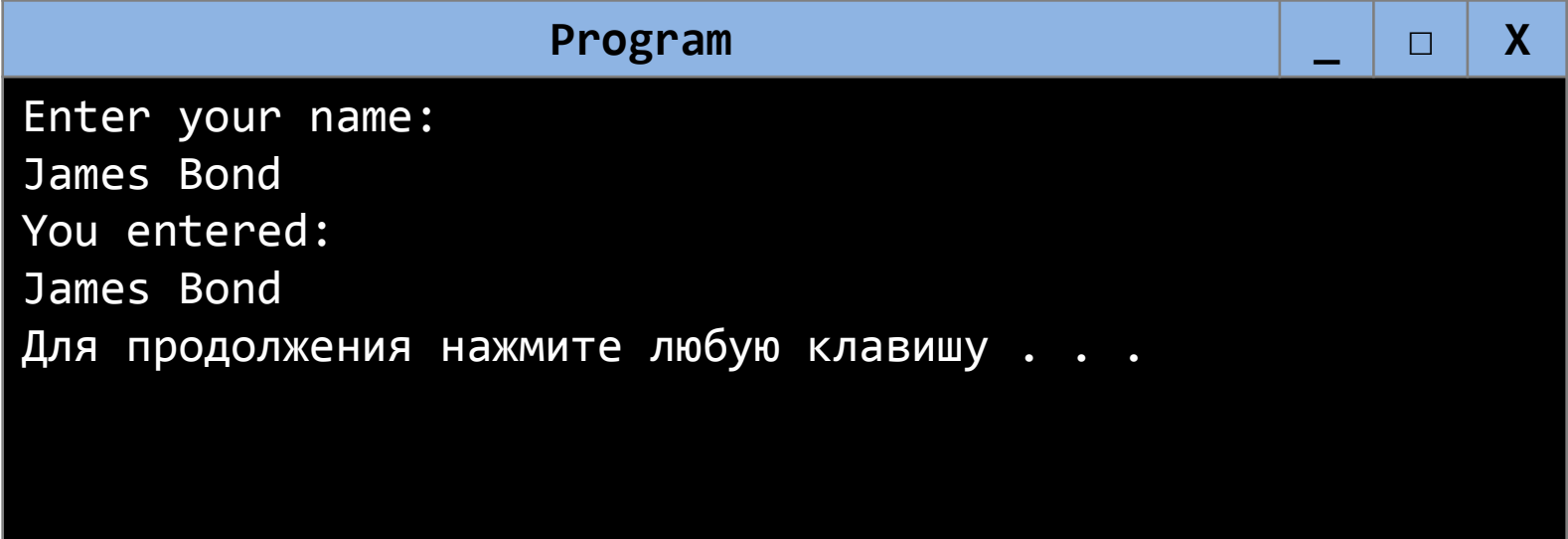
При новом чтении все пробельные символы игнорируются и удаляются из потока, пока не встретятся не пробельные с которых и начнётся считывание

Ввод строки II

```
#include <iostream>
int main()
{
    char name[255]; // буфер
    std::cout << "Enter your name: ";
    std::cin.getline(name, 255);
    std::cout << "You entered: " << name << '\n';

    return 0;
}
```

Ввод строки II



A screenshot of a Windows-style application window titled "Program". The window has a blue title bar with standard minimize, maximize, and close buttons. The main content area is black with white text. The text shows a prompt "Enter your name:", followed by the input "James Bond", and then "You entered:" followed by "James Bond". At the bottom, there is a message in Russian: "Для продолжения нажмите любую клавишу . . .".

```
Enter your name:  
James Bond  
You entered:  
James Bond  
Для продолжения нажмите любую клавишу . . .
```

Ввод строки II Через `cin.getline`

```
cin.getline(char* s, streamsize n, char delim);
```

Читает в строковую переменную `s`, символы из потока, но не более `n-1` штук, т.к. последний зарезервирован под нуль-терминатор.

Если задан `delim`, то строка будет считываться пока не встретится данный символ, если не задан, то до символа перевода строки. Данный символ извлекается из потока, не попадает в строку.

Если в процессе чтения символ разделитель так и не встретился и было считано `n-1` символ, то будет установлен флаг ошибки `failbit` и дальнейшее чтение будет заблокировано, чтобы сбросить этот флаг нужно воспользоваться командой: `cin.clear();`

Некоторые полезные функции библиотеки "cstring"

<code>strlen</code>	Определить длину строки
<code>strcpy</code>	Скопировать строку
<code>strncpy</code>	Скопировать n символов строки
<code>strcat</code>	Объединение строк
<code>strncat</code>	Добавление n символов к строке
<code>strcmp</code>	Сравнение двух строк
<code>strncmp</code>	Сравнение n первых символов двух строк
<code>strstr</code>	Функция ищет первое вхождение подстроки str2 в строке str
<code>strtok</code>	Поиск лексем в строке, используя разделители