

Основы программирования

ФИСТ 1 курс

Власенко

Олег

Федосович

Лекция 13

Символы и строки в Си.

Стандартная библиотека: `ctype.h` и `string.h`.

Работа с текстовыми файлами

Тип char

char – это «очень короткий» целый тип

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

```
void main() {
```

```
    char ch = 32;
```

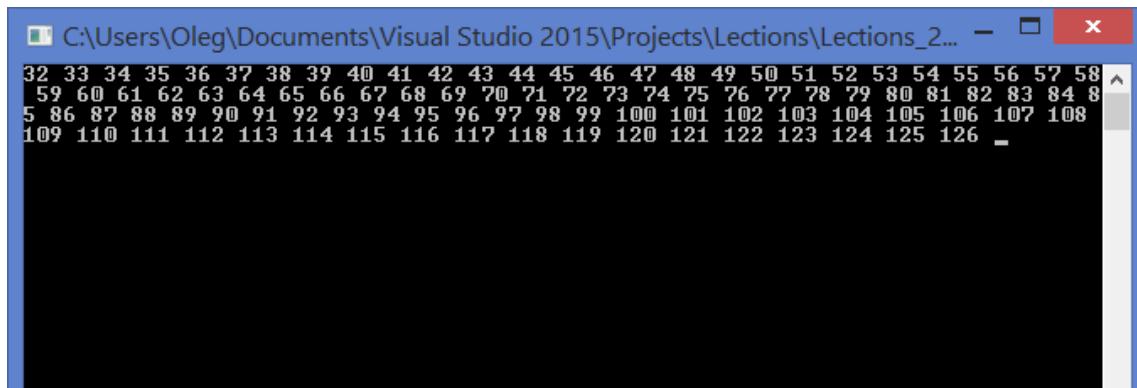
```
    while (ch < 127) {
```

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

```
        ch++;
```

```
    }
```

```
}
```



```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lectons\Lectons_2...  
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58  
59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 8  
5 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108  
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 _
```

```
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58  
59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 8  
5 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108  
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
```

Тип char (2)

char – это символьный тип

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

```
void main() {
```

```
    char ch = 32;
```

```
    while (ch < 127) {
```

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

```
        ch++;
```

```
    }
```

```
}
```

Тип char (3)

unsigned char = [0 .. 255]

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

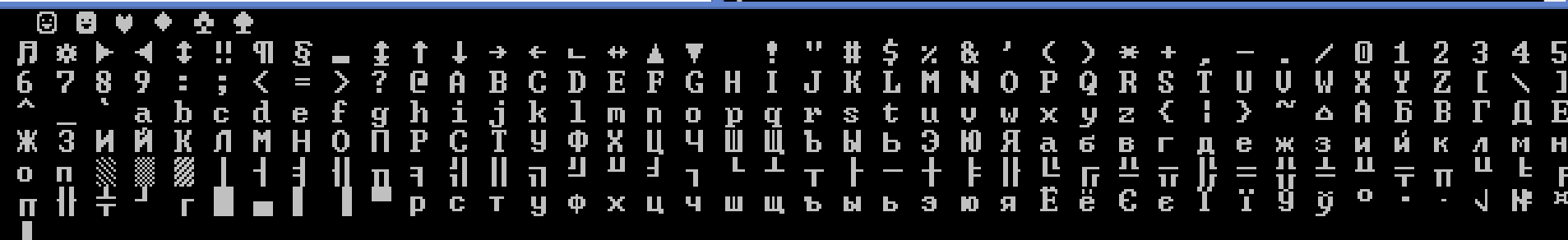
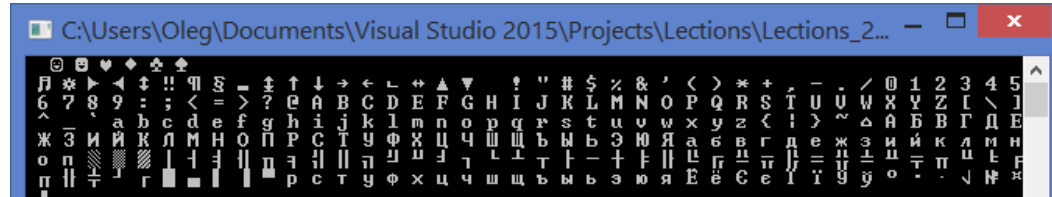
```
void main() {
```

```
unsigned char ch = 0;
```

```
while (ch < 255) {  
    printf("%c ", ch);  
    ch++;  
}
```

}

}



Тип char (4)

signed char = [-128 .. +127]

#include <stdio.h>

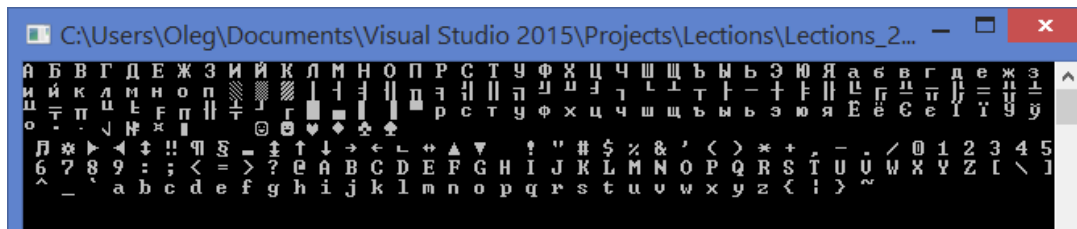
```
void main() {
```

```
    signed char ch = -128;
```

```
    while (ch < 127) {  
        printf("%c ", ch);  
        ch++;
```

```
    }
```

```
}
```



Тип char (5)

Загадка:

Тип char == signed char

ИЛИ

Тип char == unsigned char

?

Тип char (6)

<http://stackoverflow.com/questions/2054939/is-char-signed-or-unsigned-by-default>

The standard does not specify if plain char is signed or unsigned...

ASCII

<https://ru.wikipedia.org/wiki/ASCII>

ASCII ([англ.](#) *American standard code for information interchange*) — название таблицы (кодировки, набора), в которой некоторым распространённым печатным и непечатным символам сопоставлены числовые коды. Таблица была разработана и стандартизована в [США](#) в 1963 году.

ASCII Code Chart																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

Таблица ASCII



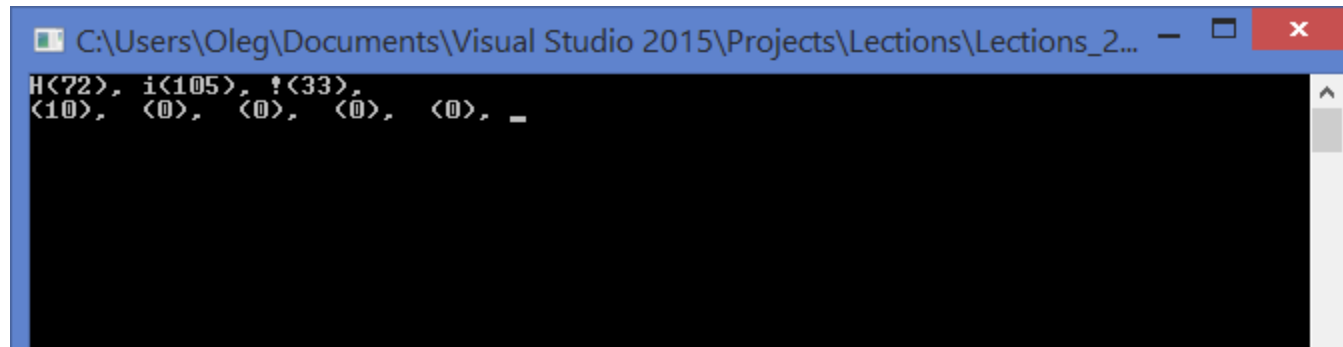
ASCIIZ

<http://stackoverflow.com/questions/7783044/whats-the-difference-between-asciiz-vs-ascii>

In computing, a C string is a character sequence terminated with a null character ('\0', called NUL in ASCII). It is usually stored as one-dimensional character array.[dubious – discuss] The name refers to the C programming language which uses this string representation. Alternative names are ASCIIZ (note that C strings do not imply the use of ASCII) and **null-terminated string**

null-terminated string

```
void main() {  
    char s1[8] = "Hi!\n";  
  
    int i;  
    for (i = 0; i < 8; i++) {  
        printf("%c(%d), ", s1[i], s1[i]);  
    }  
}
```

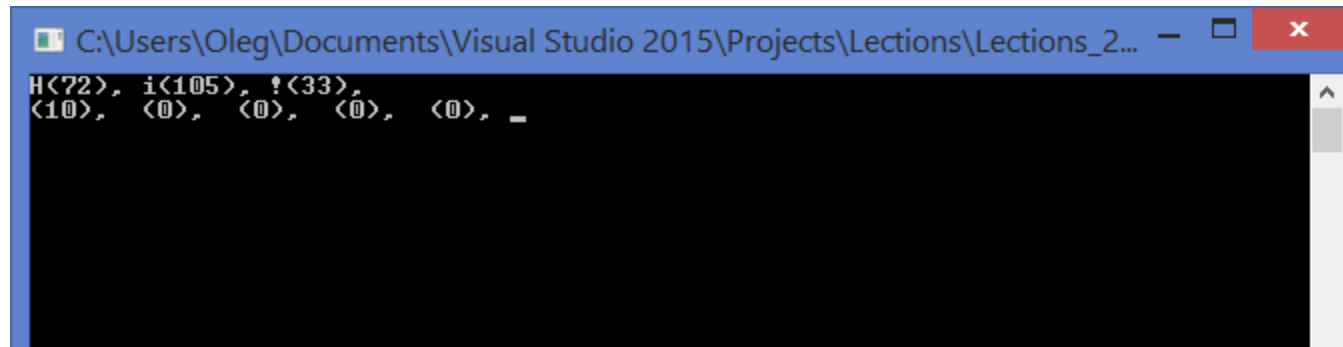


```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lectons\Lectons_2...  
H<72>, i<105>, !<33>,  
<10>, <0>, <0>, <0>, <0>, _
```

```
H<72>, i<105>, !<33>,  
<10>, <0>, <0>, <0>, <0>, _
```

Инициализация строки как массива символов

```
void main() {  
    char s1[8] = { 'H', 'i', '!', '\n', '\0' };  
  
    int i;  
    for (i = 0; i < 8; i++) {  
        printf("%c(%d), ", s1[i], s1[i]);  
    }  
}
```

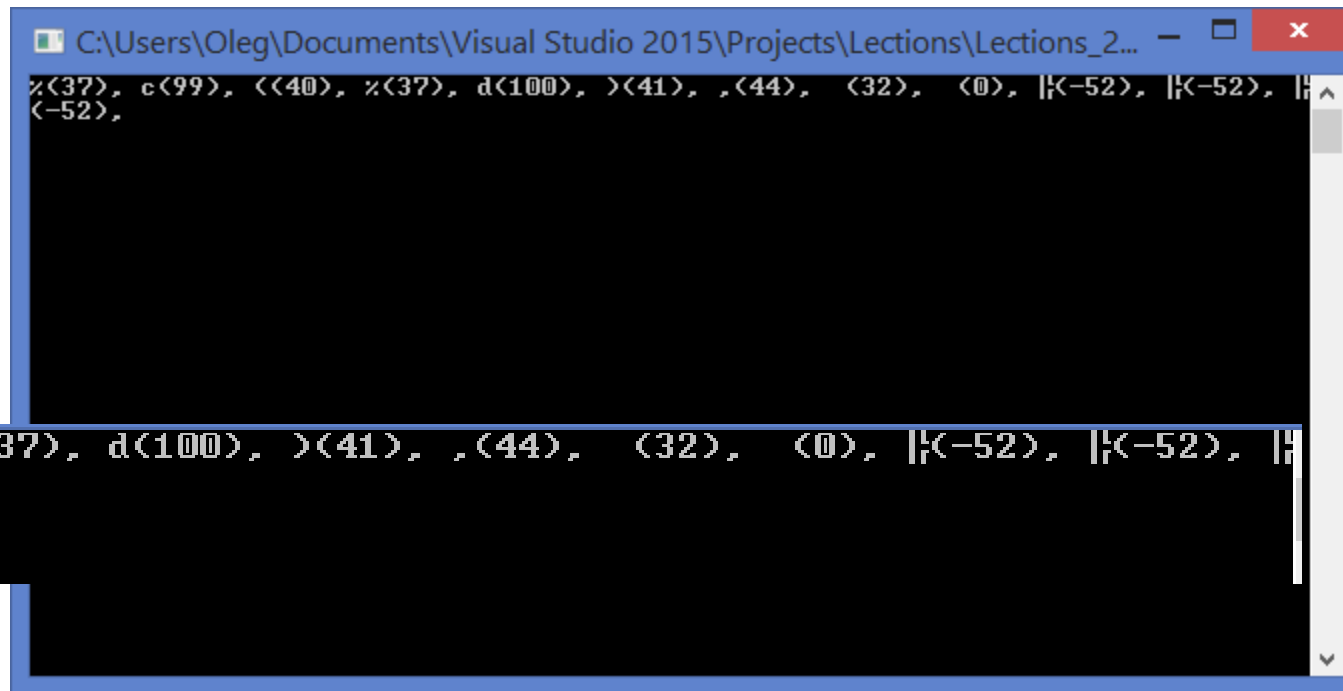


```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lectons\Lectons_2...  
H(72), i(105), !(33),  
<10>, <0>, <0>, <0>, <0>, _
```

```
H(72), i(105), !(33),  
<10>, <0>, <0>, <0>, <0>, _
```

Инициализация строки как строки

```
void main() {  
    char s2[] = "%c(%d), ";  
  
    int i;  
    for (i = 0; i < 12; i++) {  
        printf("%c(%d), ", s2[i], s2[i]);  
    }  
}
```



```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lecti...  
%(37), c(99), ((40), %(37), d(100), >(41), ,(44), (32), (0), |(-52), |(-52), |(-52),
```

Простейшие алгоритмы обработки строк (как массива символов с '\0' в конце)

Все цифры заменить на символ «#»

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

```
void main() {
```

```
    char s3[] = "I have 32 USD and 5 EUR!";
```

```
    printf("s3 = %s\n", s3);
```

```
    int i = 0;
```

```
    while (s3[i] != '\0') {
```

```
        if (s3[i] >= '0' && s3[i] <= '9') {
```

```
            s3[i] = '#';
```

```
        }
```

```
        i++;
```

```
    }
```

```
    printf("s3 = %s\n", s3);
```

```
}
```

```
s3 = I have 32 USD and 5 EUR!  
s3 = I have ## USD and # EUR!
```

```
-
```

Используем функции из ctype.h

Все цифры заменить на символ «#»

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

```
#include <ctype.h>
```

```
void main() {
```

```
    char s3[] = "I have 32 USD and 5 EUR!";
```

```
    printf("s3 = %s\n", s3);
```

```
    int i = 0;
```

```
    while (s3[i] != '\0') {
```

```
        if (isdigit(s3[i])) {
```

```
            s3[i] = '#';
```

```
        }
```

```
        i++;
```

```
    }
```

```
    printf("s3 = %s\n", s3);
```

```
}
```




```
s3 = I have 32 USD and 5 EUR!  
s3 = I have ## USD and # EUR!
```

Используем функции из ctype.h

Все ?????? заменить на символ «#»

```
void main() {  
    char s3[] = "I have 32 USD and 5 EUR!";  
    printf("s3 = %s\n", s3);  
    int i = 0;  
    while (s3[i] != '\0') {  
        if (isalpha(s3[i])) {  
            s3[i] = '#';  
        }  
        i++;  
    }  
    printf("s3 = %s\n", s3);  
}
```



The terminal output shows the string transformation. The first line is "s3 = I have 32 USD and 5 EUR!". The second line is "s3 = # ##### 32 ### ## 5 ###?".

Используем функции из ctype.h

Все ?????? заменить на символ «#»

```
void main() {  
    char s3[] = "I have 32 USD and 5 EUR!";  
    printf("s3 = %s\n", s3);  
    int i = 0;  
    while (s3[i] != '\0') {  
        if (isspace(s3[i])) {  
            s3[i] = '#';  
        }  
        i++;  
    }  
    printf("s3 = %s\n", s3);  
}
```

```
s3 = I have 32 USD and 5 EUR!  
s3 = I#have#32#USD#and#5#EUR!
```


Используем функции из ctype.h

Все ?????? заменить на символ «#»

```
void main() {  
    char s3[] = "I have 32 USD and 5 EUR!";  
    printf("s3 = %s\n", s3);  
    int i = 0;  
    while (s3[i] != '\0') {  
        if (isupper(s3[i])) {  
            s3[i] = '#';  
        }  
        i++;  
    }  
    printf("s3 = %s\n", s3);  
}
```

```
s3 = I have 32 USD and 5 EUR!  
s3 = # have 32 ### and 5 ###!
```

Используем функции из ctype.h

Все ?????? заменить на символ «#»

```
void main() {  
    char s3[] = "I have 32 USD and 5 EUR!";  
    printf("s3 = %s\n", s3);  
    int i = 0;  
    while (s3[i] != '\0') {  
        if (islower(s3[i])) {  
            s3[i] = '#';  
        }  
        i++;  
    }  
    printf("s3 = %s\n", s3);  
}
```




```
s3 = I have 32 USD and 5 EUR!  
s3 = I #### 32 USD ### 5 EUR!
```

Используем функции из ctype.h

Все ?????? заменить на ???????

```
void main() {  
    char s3[] = "I have 32 USD and 5 EUR!";  
  
    printf("s3 = %s\n", s3);  
    int i = 0;  
  
    while (s3[i] != '\0') {  
        s3[i] = toupper(s3[i]);  
        i++;  
    }  
  
    printf("s3 = %s\n", s3);  
}
```



```
s3 = I have 32 USD and 5 EUR!  
s3 = I HAVE 32 USD AND 5 EUR!
```

Используем функции из ctype.h

Все ?????? заменить на ???????

```
void main() {  
    char s3[] = "I have 32 USD and 5 EUR!";  
  
    printf("s3 = %s\n", s3);  
    int i = 0;  
  
    while (s3[i] != '\0') {  
        s3[i] = tolower(s3[i]);  
        i++;  
    }  
  
    printf("s3 = %s\n", s3);  
}
```



```
s3 = I have 32 USD and 5 EUR!  
s3 = i have 32 usd and 5 eur!
```

Стандартные функции обработки строк

strlen(s) - Возвращает длину строки без завершающей литеры '\0'.

strcmp(s1, s2) – посимвольное сравнение строк (НЕЛЬЗЯ сравнивать строки так «s1 == s2» или «s1 < s2»!!!)

strcpy (dest, src) – копирует строку src в dest, включая завершающий '\0'

strcat (dest, src) – добавляет копию src в конец dest

И еще около 20 функций из string.h

strlen()

```
#include <string.h>
```

```
void main() {
```

```
    char s[10] = "Hi!";
```

```
    printf("len = %d\n", strlen(s));
```

```
    s[3] = ' '; s[4] = '\0';
```

```
    printf("len = %d\n", strlen(s));
```

```
    s[4] = 'W'; s[5] = 'o'; s[6] = 'r'; s[7] = 'l';
```

```
    s[8] = 'd'; s[9] = '\0';
```

```
    printf("len = %d\n", strlen(s));
```

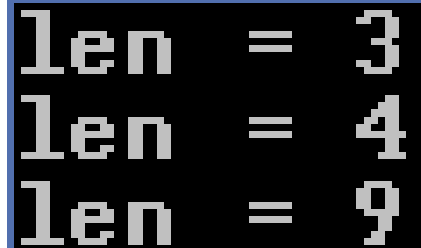
```
}
```

strlen()

```
#include <string.h>
void main() {
    char s[10] = "Hi!";
    printf("len = %d\n", strlen(s));

    s[3] = ' '; s[4] = '\0';
    printf("len = %d\n", strlen(s));

    s[4] = 'W'; s[5] = 'o'; s[6] = 'r'; s[7] = 'l';
    s[8] = 'd'; s[9] = '\0';
    printf("len = %d\n", strlen(s));
}
```



A terminal window with a blue title bar and a black background. It displays three lines of output, each showing the variable 'len' followed by an equals sign and a number. The numbers are 3, 4, and 9, corresponding to the three printf statements in the code block.

```
len = 3
len = 4
len = 9
```

Сравнение строк – НЕ ДЕЛАЙТЕ ТАК НИКОГДА!!!

```
void main() {  
    char s1[] = "Button";  
    char s2[] = "We";  
    char s3[] = "Apple !!";  
    char * min = s1;    char * max = s1;
```

```
max = Button  
min = Apple !!
```

```
    if (s2 > max) max = s2;  
    if (s3 > max) max = s3;  
    printf("max = %s\n", max);
```

```
    if (s2 < min) min = s2;  
    if (s3 < min) min = s3;  
    printf("min = %s\n", min);
```

```
}
```


Сравнение строк через strcmp

```
int strcmp(const char *str1, const char *str2);
```

Функция strcmp() сравнивает в лексикографическом порядке две строки и возвращает целое значение, зависящее следующим образом от результата сравнения.

<i>Значение</i>	<i>Результат сравнения строк</i>
Меньше нуля	str1 меньше str2
Нуль	str1 равен str2
Больше нуля	str1 больше str2

Сравнение строк через strcmp

```
void main() {  
    char s1[] = "Button";  
    char s2[] = "We";  
    char s3[] = "Apple !!";  
    char * min = s1;    char * max = s1;  
  
    if (strcmp(s2, max) > 0) max = s2;  
    if (strcmp(s3, max) > 0) max = s3;  
    printf("max = %s\n", max);  
  
    if (strcmp(s2, min) < 0) min = s2;  
    if (strcmp(s3, min) < 0) min = s3;  
    printf("min = %s\n", min);  
}
```

```
max = We  
min = Apple !!
```

Копирование строк

```
void main() {  
    char src[] = "Button";  
    char dest[10];  
  
    printf("src = %s, dest = %s\n", src, dest);  
  
    strcpy(dest, src);  
    printf("src = %s, dest = %s\n", src, dest);  
  
}
```

```
src = Button, dest = [AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA]Button  
src = Button, dest = Button
```

Конкатенация строк

```
void main() {  
    char src[] = "Button";  
    char dest[10] = "<>";  
    printf("src = %s, dest = %s\n", src, dest);  
  
    strcat(dest, src);  
    printf("src = %s, dest = %s\n", src, dest);  
  
    strcat(dest, "!");  
    printf("src = %s, dest = %s\n", src, dest);  
}
```

```
src = Button, dest = <>  
src = Button, dest = <>Button  
src = Button, dest = <>Button!
```

Еще раз - int strlen(char s[])

```
int strlen(char s[]) {  
    int len;  
    ...  
    return len;  
}
```

Возвращает длину строки без завершающей литеры '\0'.

Пример:

strlen("!!") == 2

strlen("Hi!\n") == 4

Собственная реализация strlen

```
int strlen_my(char s[]) {  
    int len;  
    ...  
    return len;  
}
```

Нужно написать код функции `strlen_my(s)`, работающей аналогично `strlen(s)`

Пример использования:

```
strlen_my("!!") == 2
```

```
strlen_my("Hi!\n") == 4
```

Собственная реализация strlen

```
int strlen_my(char *s)
{
    int len;

    for (len=0; s[len] != '\0'; len++);

    return len;
}
```

Еще раз - `int strcmp(char s1[], char s2[])`

```
int strcmp(const char *str1, const char *str2);
```

Функция `strcmp()` сравнивает в лексикографическом порядке две строки и возвращает целое значение, зависящее следующим образом от результата сравнения.

<i>Значение</i>	<i>Результат сравнения строк</i>
Меньше нуля	<code>str1</code> меньше <code>str2</code>
Ноль	<code>str1</code> равен <code>str2</code>
Больше нуля	<code>str1</code> больше <code>str2</code>

Пример использования:

```
strcmp("Abba", "Beta") < 0
```


Собственная реализация strcmp

```
/*  
s1 < s2  : <0  
s1 == s2 : 0  
s1 > s2  : >0  
*/  
int strcmp_my(char * s1, char * s2)  
{  
    return ...;  
}
```

Нужно написать код функции `strcmp_my(s1, s2)`,
работающей аналогично `strcmp(s1, s2)`

Пример использования:

`strcmp_my("Abba", "Beta") < 0`

Собственная реализация strcmp

```
/*  
s1 < s2  : <0  
s1 == s2 : 0  
s1 > s2  : >0  
*/  
int strcmp_my(char * s1, char * s2)  
{  
    int i = 0;  
    while (s1[i] != 0 && s2[i] != 0 && s1[i] == s2[i])  
        i++;  
  
    return s1[i] - s2[i];  
}
```

Домашнее задание

1. Написать собственную версию
`strcpy_my(dest, src)`
2. Написать собственную версию
`strcat(dest, src)`

Текстовый файл

Текстовый файл содержит последовательность символов (в основном печатных знаков, принадлежащих тому или иному набору символов). Эти символы обычно сгруппированы в строки (англ. *lines, rows*). В современных системах строки разделяются разделителями строк

https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB

Перевод строки

Перевод строки, или **разрыв строки** — продолжение печати текста с новой строки, то есть с левого края на строку ниже, или уже на следующей странице.

Разделителем строк, обозначающим место перевода строки, в текстовых данных служит один или пара управляющих символов, а в размеченном тексте также — определённый тег (в HTML — тег
, от англ. *break* — «разрыв»).

Перевод строки – в разных ОС

- (“\n”) LF ([ASCII 0x0A](#)) используется в [Multics](#), [UNIX](#), [UNIX-подобных операционных системах](#) ([GNU/Linux](#), [AIX](#), [Xenix](#), [Mac OS X](#), [FreeBSD](#) и др.), [BeOS](#), [Amiga](#), [UNIX](#), [RISC OS](#) и других;
- (“\r”) CR (ASCII 0x0D) используется в 8-битовых машинах [Commodore](#), машинах [TRS-80](#), [Apple II](#), системах [Mac OS](#) до [версии 9](#) и [OS-9](#);
- (“\r\n”) CR+LF (ASCII 0x0D 0x0A) используется в [DEC RT-11](#) и большинстве других ранних не-UNIX- и не-IBM-систем, а также в [CP/M](#), [MP/M](#) (*англ.*), [MS-DOS](#), [OS/2](#), [Microsoft Windows](#), [Symbian OS](#), протоколах [Интернет](#).

Чтение из текстового файла - feof

Проверяет поток на достижение конца файла.

```
int feof( FILE *stream );
```

Параметры

stream Указатель на структуру **FILE**.

Возвращаемое значение

Функция feof возвращает ненулевое значение, если операция чтения попыталась читать данные после конца файла; в противном случае она возвращает 0.

<https://msdn.microsoft.com/ru-ru/library/xsskct6e.aspx>

Чтение из текстового файла - fgets

Считывание строки из потока.

```
char *fgets(  
    char *str,  
    int n,  
    FILE *stream  
);
```

Параметры

Str - Место хранения данных.

n - Наибольшее число символов для чтения.

Stream - Указатель на структуру FILE.

Возвращаемое значение

возвращает str. Для указания ошибки или условия конца файла функция возвращает NULL.

<https://msdn.microsoft.com/ru-ru/library/c37dh6kf.aspx>

Чтение из текстового файла построчно

```
FILE * fin;
char s[MAX_LEN];
fin = fopen(filename, "rt");
// в цикле для всех строк
while (!feof(fin))
{
    // загрузить строку
    if (fgets(s, MAX_LEN - 1, fin) != NULL) {
        if (s[strlen(s) - 1] == '\n')
            s[strlen(s) - 1] = '\0';
        addWord(s);
    }
}
fclose(fin);
```

Хранилище для словаря

```
#define MAX_WORDS 10
```

```
#define MAX_LEN 25
```

```
struct Dictionary {  
    char words[MAX_WORDS][MAX_LEN];  
    int cnt_words;  
};
```

```
typedef struct Dictionary DICTIONARY;
```

```
DICTIONARY dict;
```

Хранилище для словаря (2)

```
void addWord(char * word)
{
    strncpy(dict.words[dict.cnt_words], word, MAX_LEN - 1);
    ++dict.cnt_words;
}
```

```
int contains(char * word)
{
    int i;
    for (i = 0; i < dict.cnt_words; ++i)
    {
        if (strcmp(word, dict.words[i]) == 0)
            return 1;
    }
    return 0;
}
```

Хранилище для словаря (3)

```
void print()
{
    int i;
    printf("DICTIONARY contains %d words:\n", dict.cnt_words);
    for (i = 0; i < dict.cnt_words; i++)
    {
        printf("\n%s\n", dict.words[i]);
    }
    printf("\n");
}
```

Хранилище для словаря (4)

```
int loadDictionary(char * filename) {  
    FILE * fin;  
    char s[MAX_LEN];  
    dict.cnt_words = 0;  
    fin = fopen(filename, "rt");  
    if (fin == NULL)  
    {  
        return 0;  
    }  
}
```

Хранилище для словаря (5)

```
// в цикле для всех строк
while (!feof(fin))
{
    // загрузить строку
    if (fgets(s, MAX_LEN - 1, fin) != NULL) {
        if (s[strlen(s) - 1] == '\n')
            s[strlen(s) - 1] = '\0';
        addWord(s);
    }
}
// закрыть файл
fclose(fin);
return 1;
}
```

Чтение посимвольно - `getc`

Считывает символ из потока.

```
int getc( FILE *stream );
```

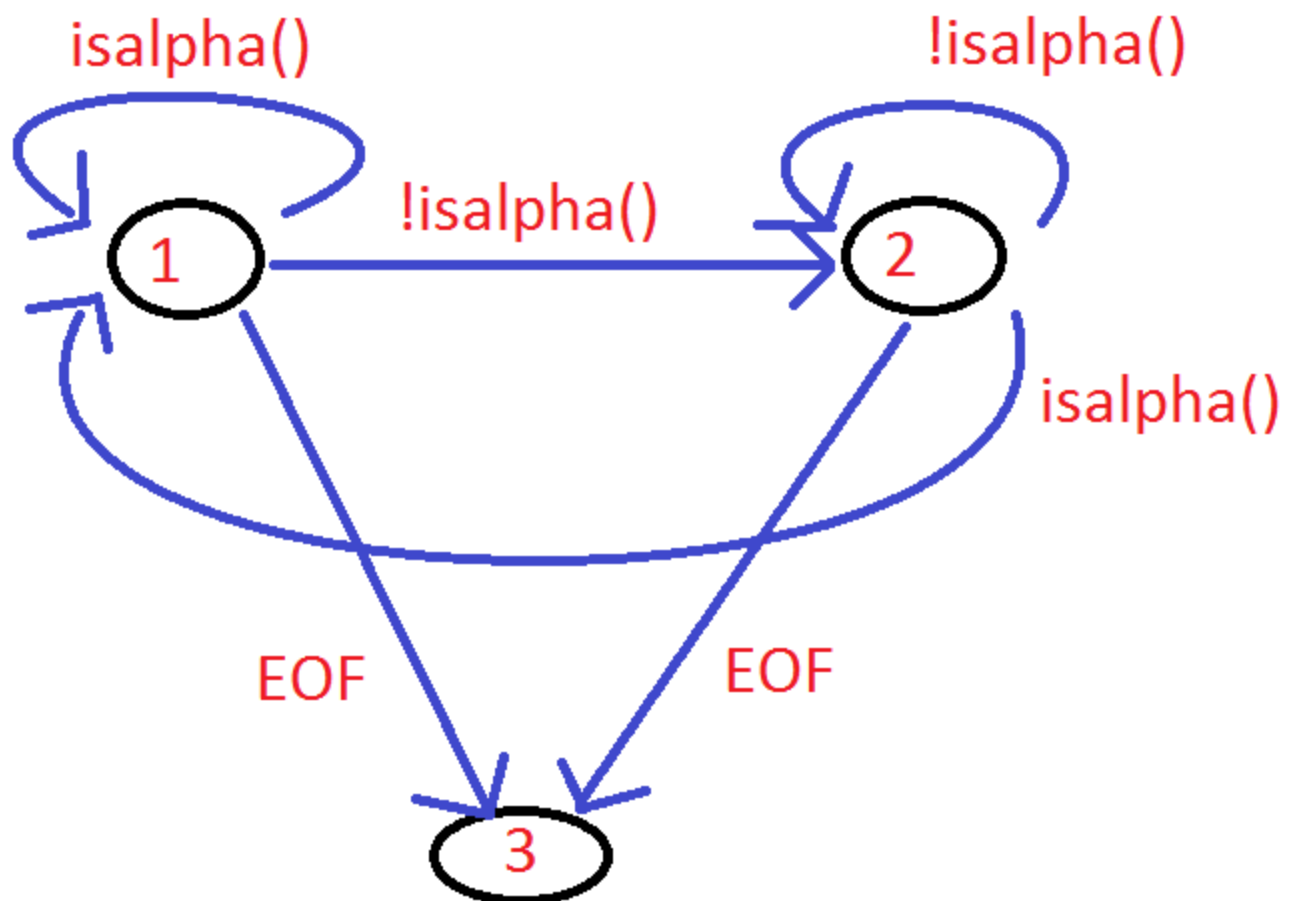
Параметры

`stream` Входной поток.

Возвращаемое значение

Возвращает считанный символ. Чтобы указать на ошибку чтения или конец файла, `getc` возвращает EOF

Граф состояний



Чтение посимвольно – для выделения слов

```
void main()
{
    char file_in[] = "c:\\Temp\\texts\\t2.txt";
    if (!loadDictionary(file_in)) {
        printf("file %s is not found!\n", file_in);
        return;
    }
    print();

    FILE *fin_t1 = fopen("c:\\Temp\\texts\\alice1.txt", "rt");
    FILE *fout = fopen("c:\\Temp\\texts\\alice1_out.txt", "wt");
    char ch;

    int is_letter = 0;
    char word[81];
    int word_len = 0;
```

Чтение посимвольно – для выделения слов (2)

```
while ((ch = getc(fin_t1)) != EOF)
{
    if (isalpha((unsigned char)ch)) {
        if (!is_letter) {
            word_len = 0;
        }
        is_letter = 1;
        word[word_len++] = ch;
    }
    else { // if (!isalpha(ch)) {
```

Чтение посимвольно – для выделения слов (3)

```
    else { // if (!isalpha(ch)) {
        if (is_letter) {
            word[word_len] = '\0';

            if (contains(word))
                fprintf(fout, "<%s>", word);
            else
                fprintf(fout, "%s", word);
        }
        is_letter = 0;
        fprintf(fout, "%c", ch);
    }
}
fclose(fin_t1);
fclose(fout);
}
```

Итог работы программы



Lister - [c:\Temp\texts\alice1.txt]

File Edit Options Help

Alice's Adventures in Wonderland
Lewis Carroll

Chapter I
DOWN THE RABBIT-HOLE

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, and what is the use of a book, thought Alice, without pictures or conversation? So she was considering in her own mind (as well as she could, for the hot day made her feel very sleepy and stupid), whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her. There was nothing so VERY remarkable in that; nor did Alice think it so VERY much out of the way to hear the Rabbit say to itself, 'Oh dear! Oh dear! I shall be late!' (when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the Rabbit actually TOOK A WATCH OUT OF ITS WAISTCOAT-POCKET, and looked at it, and then hurried on, Alice started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge.

In another moment down went Alice after it, never once considering how in the world she was to get out again. The rabbit-hole went straight on like a tunnel for some way, and then dipped suddenly down, so suddenly that Alice had not a moment to think about stopping herself before she found herself falling down a very deep well. Either the well was very deep, or she fell very slowly, for she had plenty of time as she went down to look about her and to wonder what was going to happen next. First, she tried to look down and make out what she was coming to, but it was too dark to see anything; then she looked at the sides of the well, and noticed that they were filled with cupboards and book-shelves; here and there she saw maps and pictures hung upon pegs. She took down a jar from one of the shelves as she passed; it was labelled 'ORANGE MARMALADE', but to her great disappointment it was empty: she did not like to drop the jar for fear of killing somebody, so managed to put it into one of the cupboards as she fell past it. 'Well!' thought Alice to herself, 'after such a fall as this, I shall think nothing of tumbling down stairs! How brave they'll all think me at home! Why, I



Lister - [c:\Temp\texts\alice1_out.txt]

File Edit Options Help

<Alice>'s Adventures in Wonderland
Lewis Carroll

Chapter I
DOWN THE RABBIT-HOLE

<Alice> was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, and what is the use of a book, thought <Alice> without pictures or conversation? So she was considering in her own mind (as well as she could, for the hot day made her feel very sleepy and stupid), whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her. There was nothing so VERY remarkable in that; nor did <Alice> think it so VERY much out of the way to hear the Rabbit say to itself, 'Oh dear! Oh dear! I shall be late!' (when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the Rabbit actually TOOK A WATCH OUT OF ITS WAISTCOAT-POCKET, and looked at it, and then hurried on, <Alice> started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge.

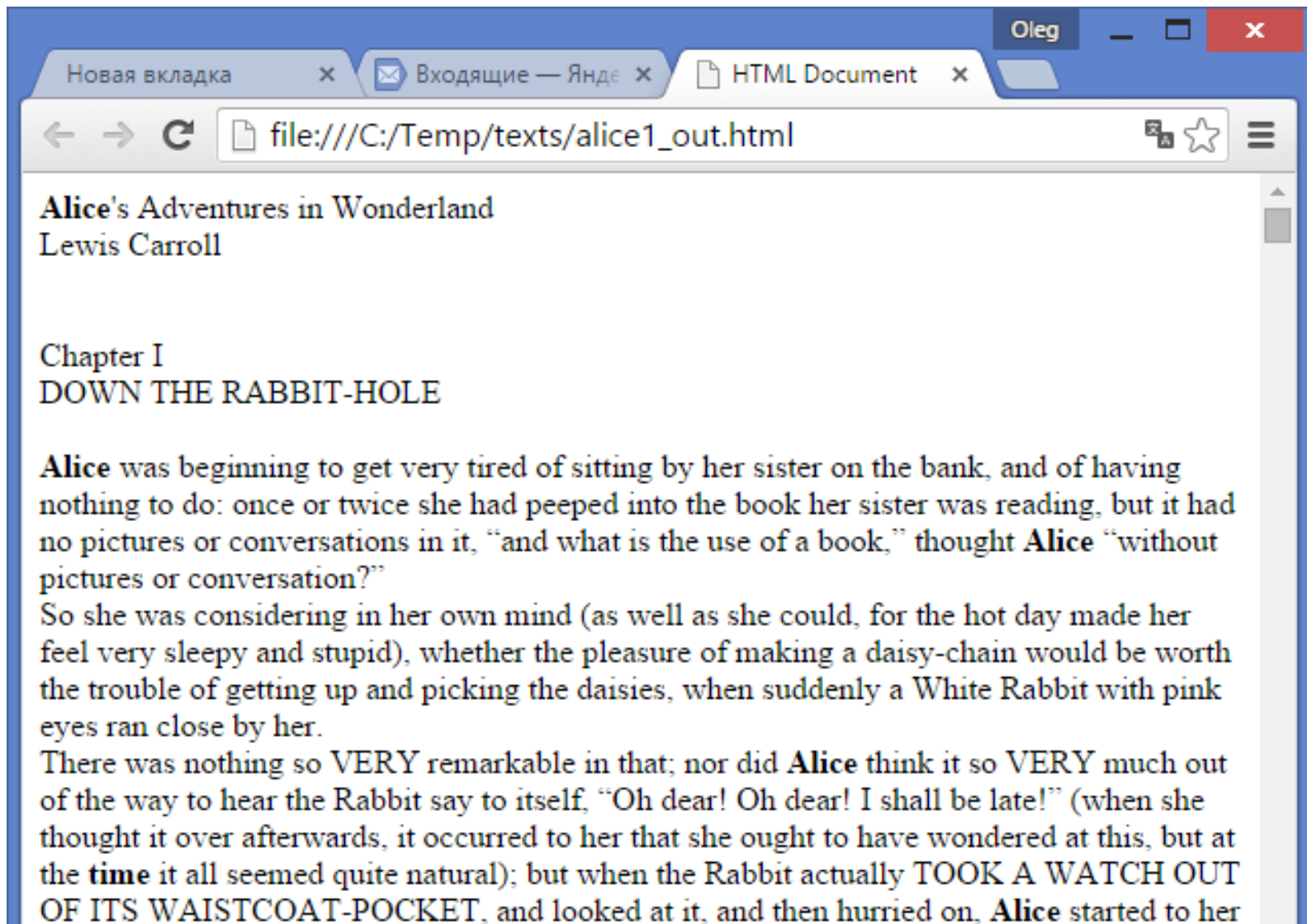
In another moment down went <Alice> after it, never once considering how in the world she was to get out again. The rabbit-hole went straight on like a tunnel for some way, and then dipped suddenly down, so suddenly that <Alice> had not a moment to think about stopping herself before she found herself falling down a very deep well. Either the well was very deep, or she fell very slowly, for she had plenty of time as she went down to look about her and to wonder what was going to happen next. First, she tried to look down and make out what she was coming to, but it was too dark to see anything; then she looked at the sides of the well, and noticed that they were filled with cupboards and book-shelves; here and there she saw maps and pictures hung upon pegs. She took down a jar from one of the shelves as she passed; it was labelled 'ORANGE MARMALADE', but to her great disappointment it was empty: she did not like to drop the jar for fear of killing somebody, so managed to put it into one of the cupboards as she fell past it. 'Well!' thought <Alice> to herself, 'after such a fall as this, I shall think nothing of tumbling down stairs! How brave they'll all think me at home! Why, I

HTML

HTML (от [англ.](#) *HyperText Markup Language* — «язык [гипертекстовой](#) разметки») — стандартный [язык разметки](#) документов во [Всемирной паутине](#). Большинство [веб-страниц](#) содержат описание разметки на языке HTML (или [XHTML](#)). Язык HTML интерпретируется [браузерами](#); полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>HTML Document</title>
  </head>
  <body>
    <p>
      <b>
        Этот текст будет полужирным,
        <i>а этот - ещё и курсивным</i>
      </b>
    </p>
  </body>
</html>
```

Генерация HTML



Генерация HTML (1)

```
void main()
{
    char file_in[] = "c:\\Temp\\texts\\t2.txt";
    if (!loadDictionary(file_in))
    {
        printf("file %s is not found!\n", file_in);
        return;
    }
    print();
    FILE *fin_t1 = fopen("c:\\Temp\\texts\\alice1.txt", "rt");
    FILE *fout = fopen("c:\\Temp\\texts\\alice1_out.html", "wt");
    fprintf(fout, "<!DOCTYPE html>");
    fprintf(fout, "<html>");
    fprintf(fout, "<head>");
    fprintf(fout, "<meta http - equiv = \"Content-Type\" content =
\"text/html; charset=utf-8\" />");
    fprintf(fout, "<title>HTML Document</title>");
    fprintf(fout, "</head>");
    fprintf(fout, "<body>");
```

Генерация HTML (2)

```
char ch;
int is_letter = 0;
char word[81];
int word_len = 0;
// в цикле
while ((ch = getc(fin_t1)) != EOF)
{
    if (isalpha((unsigned char)ch)) {
        if (!is_letter) {
            word_len = 0;
        }
        is_letter = 1;
        word[word_len++] = ch;
    }
    else { // if (!isalpha(ch)) {
```


Генерация HTML (3)

```
else { // if (!isalpha(ch)) {
    if (is_letter) {
        word[word_len] = '\0';
        if (contains(word))
            fprintf(fout, "<b>%s</b>", word);
        else
            fprintf(fout, "%s", word);
    }
    is_letter = 0;
    fprintf(fout, "%c", ch);
    if (ch == '\n')
        fprintf(fout, "<br>");
}
}
fclose(fin_t1);
fprintf(fout, "</body>");
fprintf(fout, "</html>");
fclose(fout);
}
```

Домашнее задание

1. Собрать код для превращения текста в HTML с учетом слов из словаря
2. Файл словарь содержит 100+ слов – переделать программу так, чтобы она могла обработать такой файл
3. Генерация связанных HTML страниц
`alice2_out.html`
разбить входной файл на несколько частей

Источники информации

- Google в помощь!
- MSDN