

# **ОСНОВЫ ПРОГРАММИРОВАНИЯ**

**ФИСТ 1 курс**

Власенко

Олег

Федосович

**Лекция 12**

**Символы и строки в Си.**

**Работа с текстовыми файлами**

# Тип char

**char – это «очень короткий» целый тип**

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

```
void main() {
```

```
    char ch = 32;
```

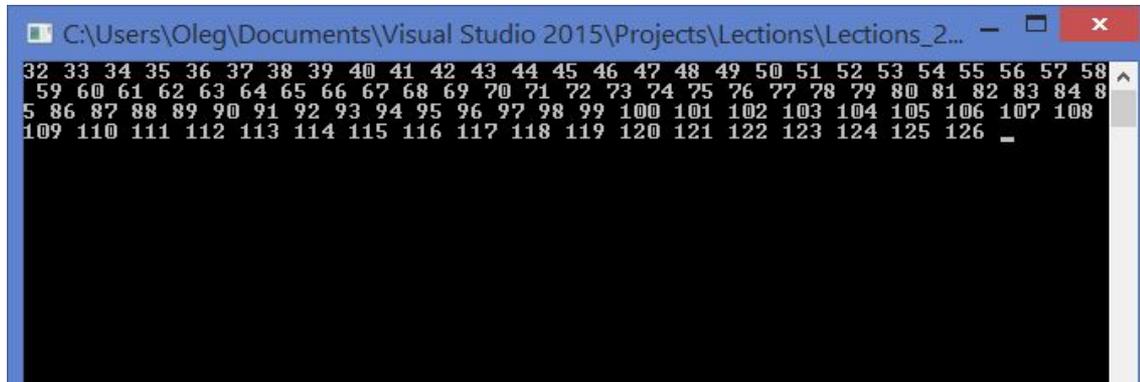
```
    while (ch < 127) {
```

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

```
        ch++;
```

```
    }
```

```
}
```



```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lecti... - [x]
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;
```

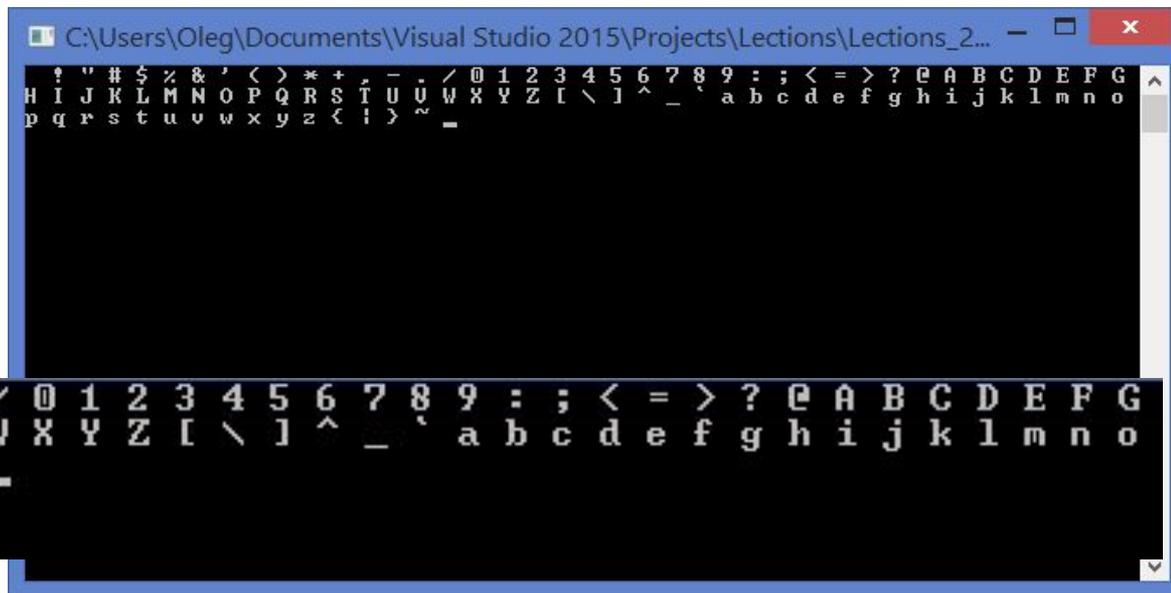
```
    while (ch < 127) {
```

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

```
        ch++;
```

```
    }
```

```
}
```



```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lections\Lections_2... - x
! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G
H I J K L M N O P Q R S T U U W X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o
p q r s t u v w x y z < ! > ~ _
```

```
! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G
H I J K L M N O P Q R S T U U W X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o
p q r s t u v w x y z < ! > ~ _
```

# Тип char (3)

```
unsigned char = [0 .. 255]
```

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

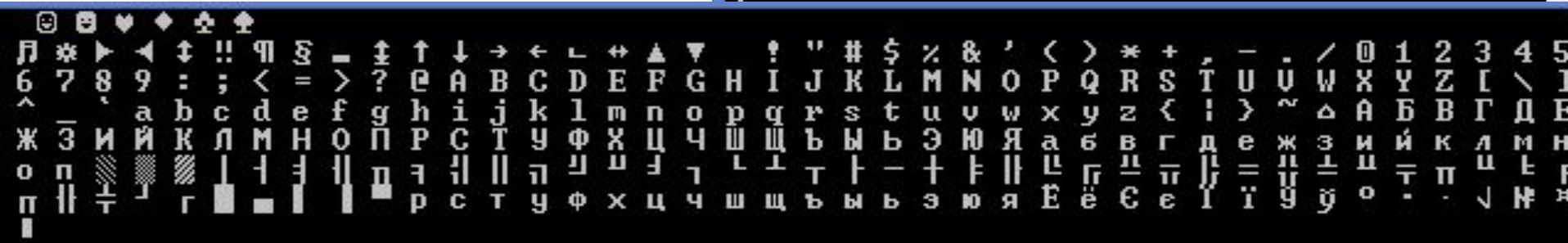
```
void main() {
```

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

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

```
    }
```

```
}
```





## Тип char (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.

	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 

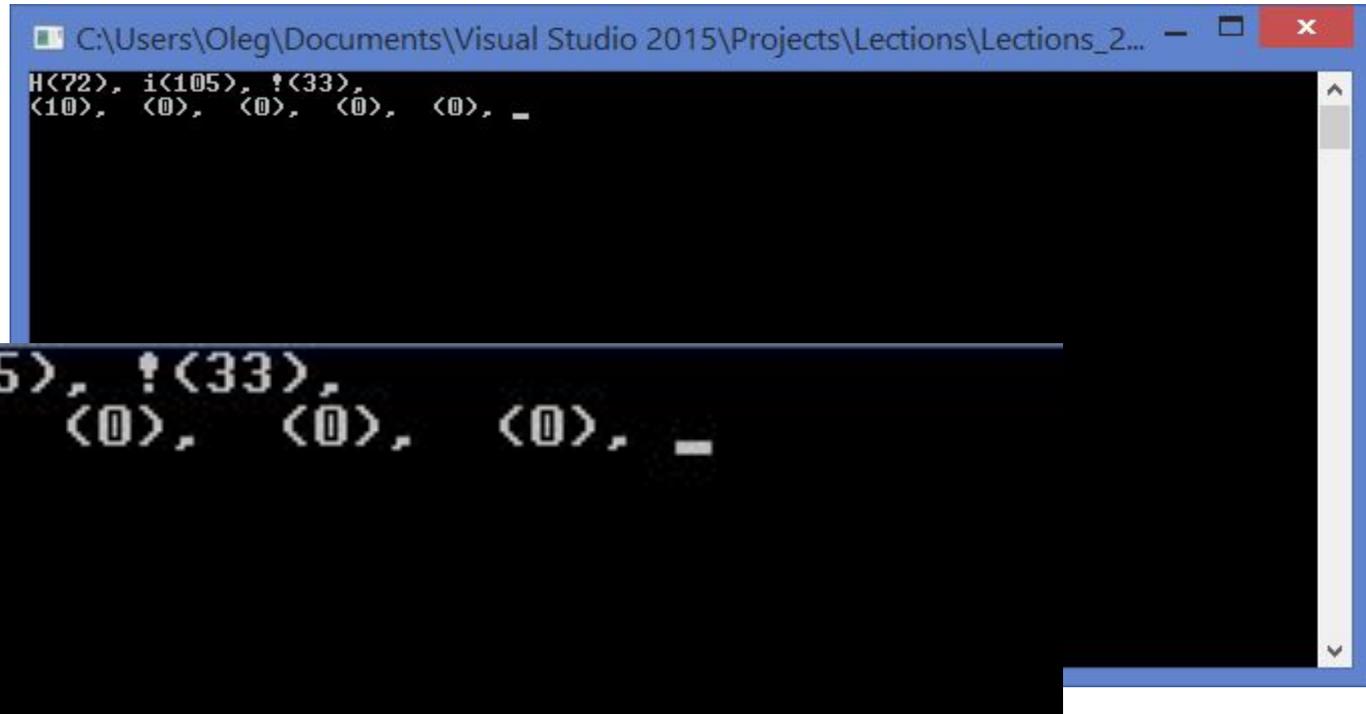
# ASCIIZ

<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

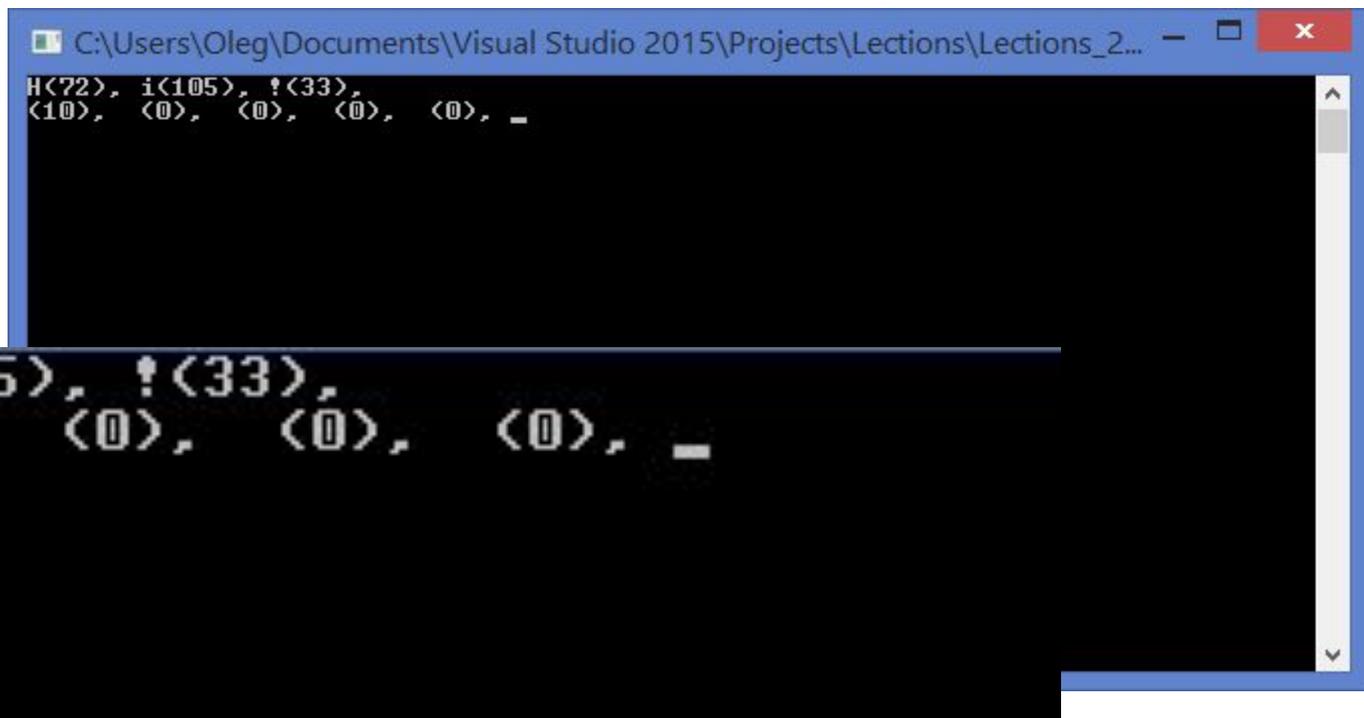
```
void main() {  
    char s1[8] = "Hi!\n";  
  
    int i;  
    for (i = 0; i < 8; i++) {  
        printf("%c(%d), ", s1[i], s1[i]);  
    }  
}
```



```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lecti... - [x]  
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\Lectiions\Lectiions_2...  
H<72>, i<105>, !<33>, <10>, <0>, <0>, <0>, <0>, _
```

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



# 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);
```

```
}
```

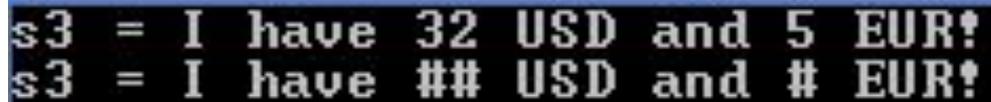
```
s3 = I have 32 USD and 5 EUR!  
s3 = I have ## USD and # EUR!
```

```
—
```

# 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);
}
```

```
s3 = I have 32 USD and 5 EUR!
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++;
    }

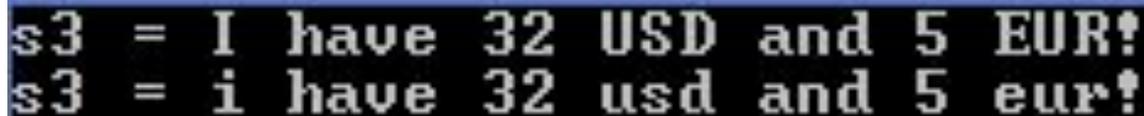
    printf("s3 = %s\n", s3);
}
```

```
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));
```

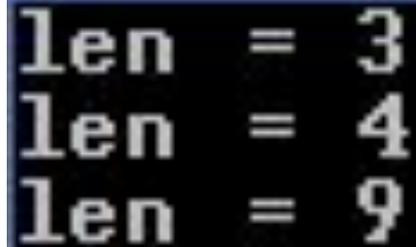
```
}
```

# 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));

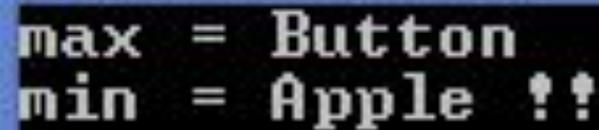
    s[4] = 'W'; s[5] = 'o'; s[6] = 'r'; s[7] = 'l';
    s[8] = 'd'; s[9] = '\0';
    printf("len = %d\n", strlen(s));
}
```



A terminal window with a black background and white text. It displays three lines of output: "len = 3", "len = 4", and "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;  
  
    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);  
}
```



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

# Сравнение строк через strcmp

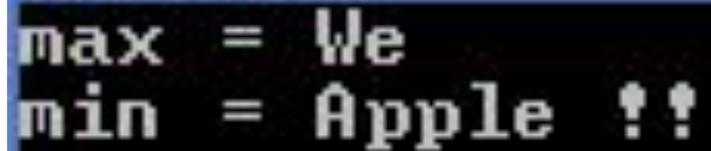
```
int strcmp(const char *str1, const char *str2);  
int strcmp(char str1[], char str2[]);
```

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 = 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 = 0;  
  
    while (s[len] != '\0')  
        len++;  
  
    return len;  
}
```

# `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>
Нуль	<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])  

```

# Трассировка 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](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 `<br>`, 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/xssk6e.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

```
char filename[] = "d:\\temp\\text_in.txt";
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';
        convert(s);
        printf("%s\n", s);
    }
}
fclose(fin);
```

# Задача шифрования текста

На входе текстовый файл in.txt (содержащий слова, разделители и знаки препинания).

Необходимо сохранив все разделители и знаки препинания на своих местах, зашифровать текст используя шифр Цезаря.

[https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80\\_%D0%A6%D0%B5%D0%B7%D0%B0%D1%80%D1%8F](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%A6%D0%B5%D0%B7%D0%B0%D1%80%D1%8F)

<http://altaev-aa.narod.ru/security/Cesar.html>

ROT13 - <https://ru.wikipedia.org/wiki/ROT13>

# Шифрование одной строки

```
#define MAX_LEN 80
```

```
#define KEY +3
```

```
#define KEY2 -3
```

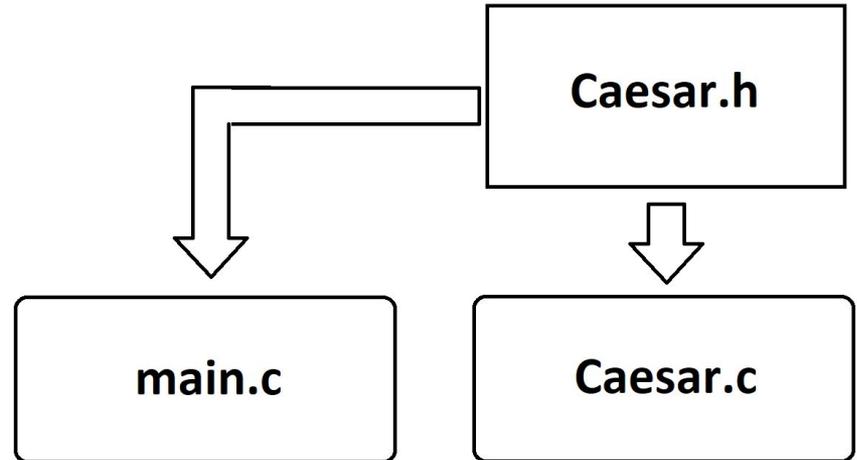
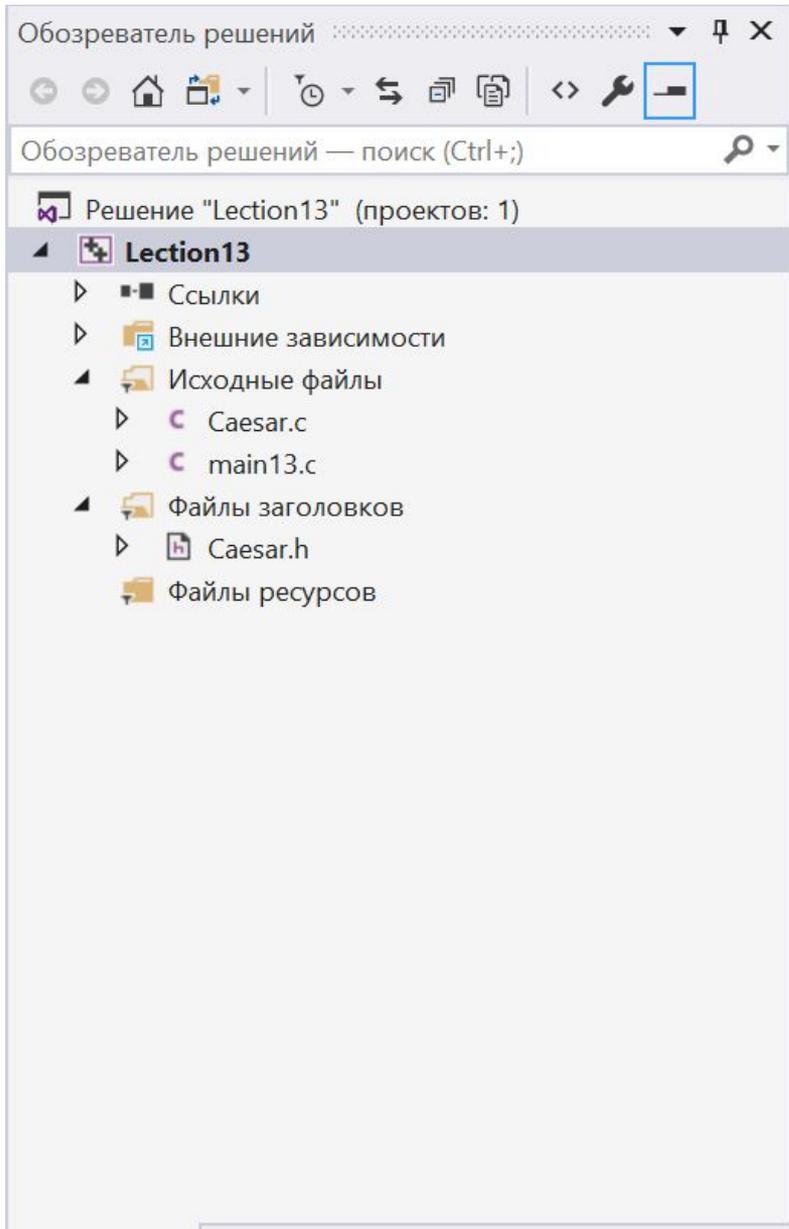
```
void convert(char * str) {  
    int i;  
    for (i = 0; str[i] != '\0'; i++) {  
        str[i] = encode(str[i], KEY);  
    }  
}
```

# Шифрование одного символа

```
int encode(int ch, int key) {  
    //char smallLetters[] ="abcdefghijklmnopqrstuvwxyz";  
    //char bigLetters[] ="ABCDEFGHIJKLmnopqrstuvwxyz";  
  
    int newCh = ch;  
  
    if (ch >= 'A' && ch <= 'Z') {  
        newCh = ch + key;  
        if (newCh > 'Z')  
            newCh = 'A' + (newCh - 'Z');  
    }  
  
    if (ch >= 'a' && ch <= 'z') {  
        newCh = ch + key;  
        if (newCh > 'z')  
            newCh = 'a' + (newCh - 'z');  
    }  
  
    return newCh;  
}
```

# **Многомодульные проекты**

# Структура многомодульного проекта



# Main.c

```
1 #define _CRT_SECURE_NO_WARNINGS
2
3 #include <stdio.h>
4 #include <string.h>
5 #include "Caesar.h"
6
7 #define MAX_LEN 80
8
9 void main() {
10
11     char filenameIn[] = "d:\\temp\\text_in.txt";
12     char filenameOut[] = "d:\\temp\\text_out.txt";
13     char filenameOut2[] = "d:\\temp\\text_out2.txt";
14
15     FILE * fin;
16     char s[MAX_LEN];
17     FILE * fout;
18
19     fin = fopen(filenameOut, "rt");
20     fout = fopen(filenameOut2, "wt");
21
22
23     // в цикле для всех строк
24     while (!feof(fin))
25     {
26         // загрузить строку
27         if (fgets(s, MAX_LEN - 1, fin) != NULL) {
28             if (s[strlen(s) - 1] == '\n')
29                 s[strlen(s) - 1] = '\0';
30             convert(s);
31             printf("%s\n", s);
32             fprintf(fout, "%s\n", s);
33         }
34     }
35     fclose(fin);
36     fclose(fout);
37
38
39     {
40         int x;
41         scanf("%d", &x);
42     }
43 }
44
45
```

Строка 22    Столбец 1    Зн    ВСТ    ↑ Добавить в систему управления версиями

# Main.c

```
#define _CRT_SECURE_NO_WARNINGS

#include <stdio.h>
#include <string.h>
#include "Caesar.h"

#define MAX_LEN 80

void main() {
    ...
    fin = fopen(filenameOut, "rt");
    fout = fopen(filenameOut2, "wt");
    // 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';
            convert(s);
            printf("%s\n", s);
        }
    }
}
```

# Caesar.h

The image shows a screenshot of the Microsoft Visual Studio IDE. The title bar reads "Lecture13 - Microsoft Visual Studio" and includes a search box for "Быстрый запуск (Ctrl+Q)". The menu bar contains "Файл", "Правка", "Вид", "Проект", "Сборка", "Отладка", and "Команда". The status bar at the bottom shows "Строка 9", "ВСТ", and "Добавить в систему управления версиями".

The interface is divided into several panes:

- Обозреватель решений (Solution Explorer):** Located on the left, it shows a project named "Lecture13" with a tree view containing "Ссылки", "Внешние зависимости", "Исходные файлы" (with sub-items "Caesar.c" and "main13.c"), "Файлы заголовков" (with "Caesar.h" selected), and "Файлы ресурсов".
- Обозреватель серверов (Server Explorer):** Located on the right, it is currently empty.
- Панель элементов (Toolbox):** Located on the right, it is currently empty.
- Свойства (Properties Window):** Located on the right, it is currently empty.

The main editor window displays the content of "Caesar.h":

```
1 #pragma once
2
3 #define KEY +3
4 #define KEY2 -3
5
6
7 void convert(char * str) {
8
9     //int encode(int ch, int key);
10
```

# Caesar.h

```
#pragma once
```

```
#define KEY +3
```

```
#define KEY2 -3
```

```
void convert(char * str);
```

```
//int encode(int ch, int key);
```

# Caesar.c

The screenshot shows the Microsoft Visual Studio IDE with the following components:

- Menu Bar:** Файл, Правка, Вид, Проект, Сборка, Отладка, Команда, Средства, Тест, Анализ, Окно, Олег Власенко
- Toolbar:** Includes icons for navigation, search, and a dropdown menu set to "Debug" with "x86" architecture. A button for "Быстрый запуск (Ctrl+Q)" is also visible.
- Left Panel (Solution Explorer):** Shows the project structure for "Lection13", including files like "Caesar.c", "main13.c", and "Caesar.h".
- Editor:** Displays the source code for "Caesar.c" with line numbers 1 through 31. The code includes a header file and implements an encode function and a convert function.
- Right Panel:** Contains the "Обозреватель серверов" (Server Explorer) and "Панель элементов" (Properties Window).
- Bottom Panel:** Shows a zoom level of 90% and navigation buttons like "Обозр...", "Предс...", "Диспе...", and "Team..."

```
1 #include "Caesar.h"
2
3 int encode(int ch, int key) {
4     // char smallLetters[] = "abcdefghijklmnopqrstuvwxyz";
5     // char bigLetters[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
6
7     int newCh = ch;
8
9     if (ch >= 'A' && ch <= 'Z') {
10        newCh = ch + key;
11        if (newCh > 'Z')
12            newCh = 'A' + (newCh - 'Z');
13    }
14
15    if (ch >= 'a' && ch <= 'z') {
16        newCh = ch + key;
17        if (newCh > 'z')
18            newCh = 'a' + (newCh - 'z');
19    }
20
21    return newCh;
22 }
23
24
25
26 void convert(char * str) {
27     int i;
28     for (i = 0; str[i] != '\0'; i++) {
29         str[i] = encode(str[i], KEY2);
30     }
31 }
```

# Caesar.c

```
#include "Caesar.h"
```

```
int encode(int ch, int key) {  
    //char smallLetters[] ="abcdefghijklmnopqrstuvwxyz";  
    //char bigLetters[] ="ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
  
    int newCh = ch;  
  
    if (ch >= 'A' && ch <= 'Z') {  
        newCh = ch + key;  
        if (newCh > 'Z')  
            newCh = 'A' + (newCh - 'Z');  
    }  
  
    if (ch >= 'a' && ch <= 'z') {  
        newCh = ch + key;  
        if (newCh > 'z')  
            newCh = 'z' + (newCh - 'z');  
    }  
  
    return newCh;  
}
```

# Caesar.c

```
void convert(char * str) {  
    int i;  
    for (i = 0; str[i] != '\0'; i++) {  
        str[i] = encode(str[i], KEY2);  
    }  
}
```

# **Работа со словами (файлы)**

# Задача выделения слов

На входе текстовый файл, содержащий большой текст (книга) на английском языке.

Нужно создать выходной текстовый файл, где есть весь текст из входного файла, но слова, содержащие 5 и больше букв помечены звездочками с обеих сторон. (Так делаются **жирными** фрагменты текста в Skype)

# 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

# Код программы

```
#define _CRT_SECURE_NO_WARNINGS
```

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

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

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

```
void main() {
```

```
    printf("Start\n");
```

```
    FILE *fin = fopen("D:\\temp\\Files\\Lec12\\alice.txt", "rt");
```

```
    FILE *fout = fopen("D:\\temp\\Files\\Lec12\\alice1_out.txt", "wt");
```

```
    int is_letter = 0;
```

```
    char word[81]; // если слово будет длиннее 80 символов – всё
```

**сломается!**

```
    int word_len = 0;
```

```
    int ch;
```

# Код программы (2)

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

# Код программы (3)

```
else { // if (!isalpha(ch)) {  
  
    if (is_letter) {  
        word[word_len] = '\0';  
  
        if (word_len >= 5)  
            fprintf(fout, "*%s*", word);  
        else  
            fprintf(fout, "%s", word);  
    }  
    is_letter = 0;  
    fprintf(fout, "%c", ch);  
} // else  
} // while ((ch = getc(fin)) != EOF)
```

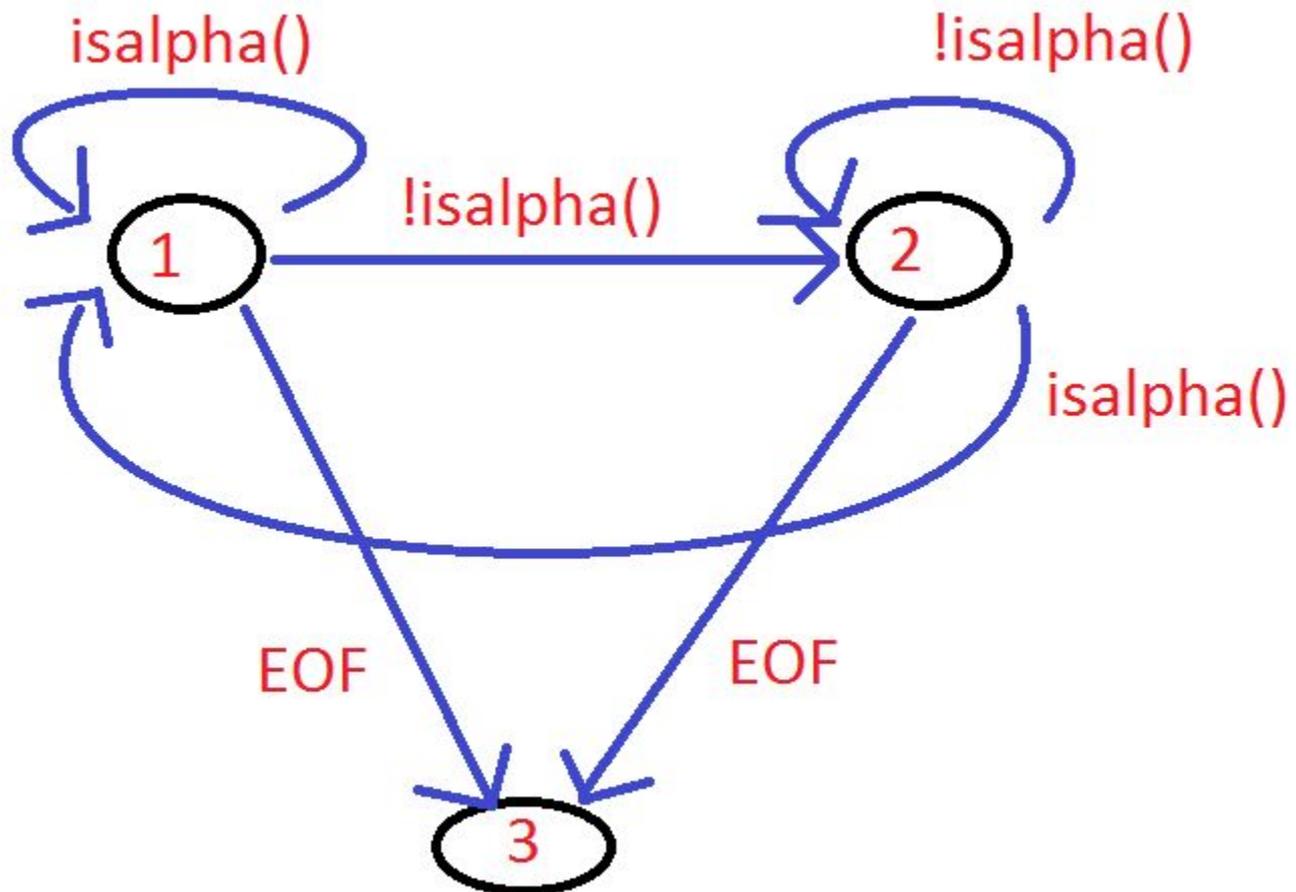
# Код программы (4)

```
fclose(fin);  
fclose(fout);
```

```
printf("Done!\n");  
{  
    int x;  
    scanf("%d", &x);  
}
```

```
}
```

# Граф состояний



# Итог работы программы

```
Lister - [D:\temp\Files\Lec12\alice.txt]
File Edit Options Help
ALICE*TAWS ADVENTURES
IN WONDERLAND

By Lewis Carroll
```

```
CHAPTER I. Down the Rabbit-Hole
Alice was beginning to get very tired of sitting by her sis
of having nothing to do: once or twice she had peeped in
was reading, but it had no pictures or conversations in it
use of a book,в Ъ™ thought Alice в Ъ without pictures or

So she was considering in her own mind (as well as she
made her feel very sleepy and stupid), whether the plea
daisy-chain would be worth the trouble of getting up and
when suddenly a White Rabbit with pink eyes ran close

There was nothing so very remarkable in that; nor did Al
out of the way to hear the Rabbit say to itself, в Ъ Oh de
late!в Ъ™ (when she thought it over afterwards, it occur
to have wondered at this, but at the time it all seemed qu
the Rabbit actually took a watch out of its waistcoat-poc
and then hurried on, Alice started to her feet, for it flashe
that she had never before seen a rabbit with either a wai
watch to take out of it, and burning with curiosity, she ra
after it, and fortunately was just in time to see it pop dow
under the hedge.

In another moment down went Alice after it, never once
world she was to get out again.
```

```
Lister - [D:\temp\Files\Lec12\alice1_out.txt]
File Edit Options Help
*ALICE*TAWS *ADVENTURES*
IN *WONDERLAND*

By *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, в Ъ and what is the use of a book,в Ъ™ *thought* *Alice* в Ъ *without*
*pictures* or *conversations*?в Ъ™

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*, в Ъ Oh dear! Oh
dear! I *shall* be late!в Ъ™ (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*.
```

# **Работа с русским языком**

# Задача 1 - создать файл в русской кодировке

```
#include <stdio.h>
#include <Windows.h>
void main() {
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    char ss[3][80] = {"задача 2",
        "Преобразовать файл in2.txt в файл out2.txt",
        "Заменяя все большие русские буквы на маленькие"};
    FILE * fout = fopen("D:\\temp\\Files\\Lab11_12\\in1.txt", "wt");
    // в цикле для всех строк
    for (int i = 0; i < 3; i++) {
        // сохранить строку в выходной файл
        fprintf(fout, "%s\n", ss[i]);
        printf(">>%s<<\n", ss[i]);
    }
    fclose(fout);
}
```

# Источники информации

- Работа с русским языком -  
<https://nicknixer.ru/programmirovanie/russkiye-simvolybukvy-pri-vvodevyvode-v-konsol-na-c/>
- Msdn
- Википедия