

# **ОСНОВЫ программирования**

## **ФИСТ 1 курс**

Власенко  
Олег  
Федосович

### **Лекция 8**

**Символы и строки в Си.**

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

**Собственная реализация стандартных функций.**

# Тип char

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

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

```
void main() {
```

```
    char ch = 32;
```

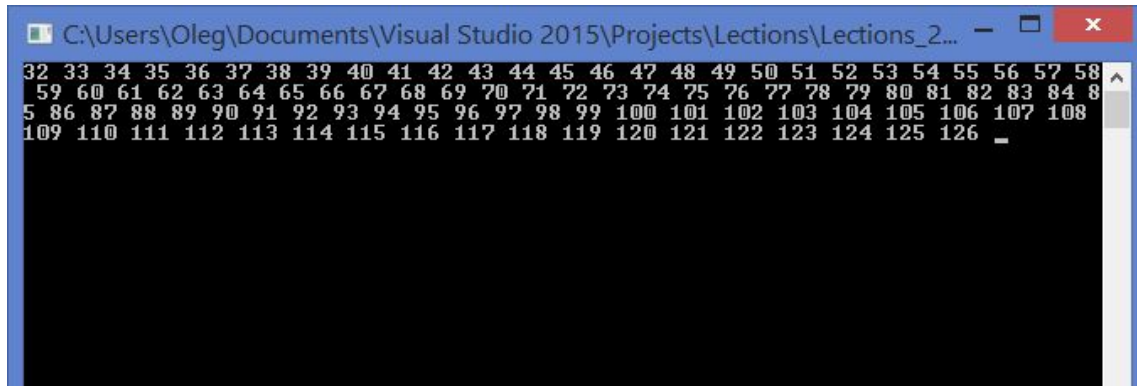
```
    while (ch < 127) {
```

```
        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++;
```

```
    }
```

```
}
```

```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lections\Lections_2...  
! \" # $ % & ' ( ) * + , - . / 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 V 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 V 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() {**

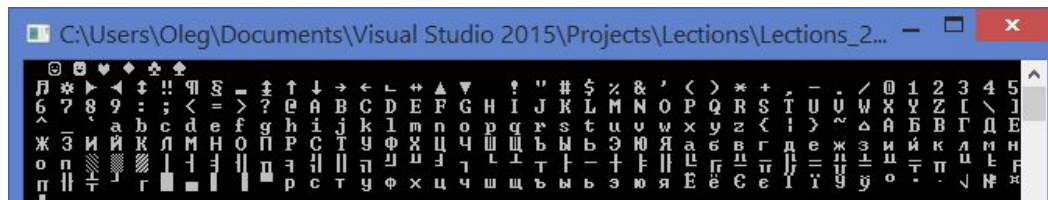
**unsigned char ch = 0;**

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

**ch++;**

**}**

**}**



# Тип char (4)

**signed char = [-128 .. +127]**

**#include <stdio.h>**

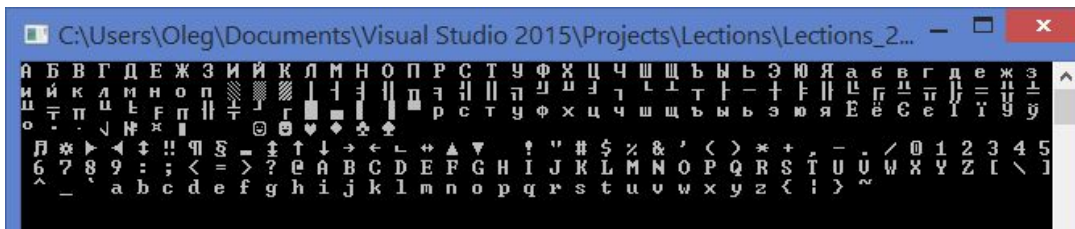
```
void main() {
```

```
    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



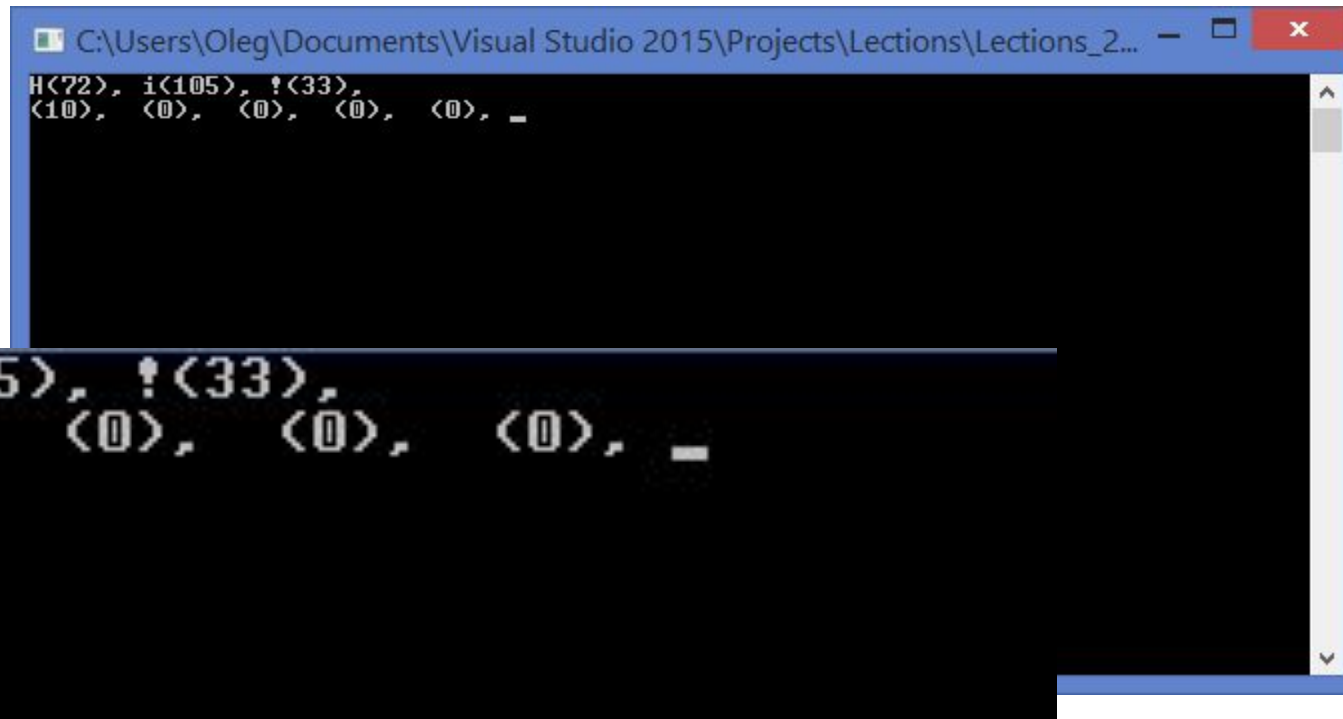
# ASCIIZ

<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]);  
    }  
}
```

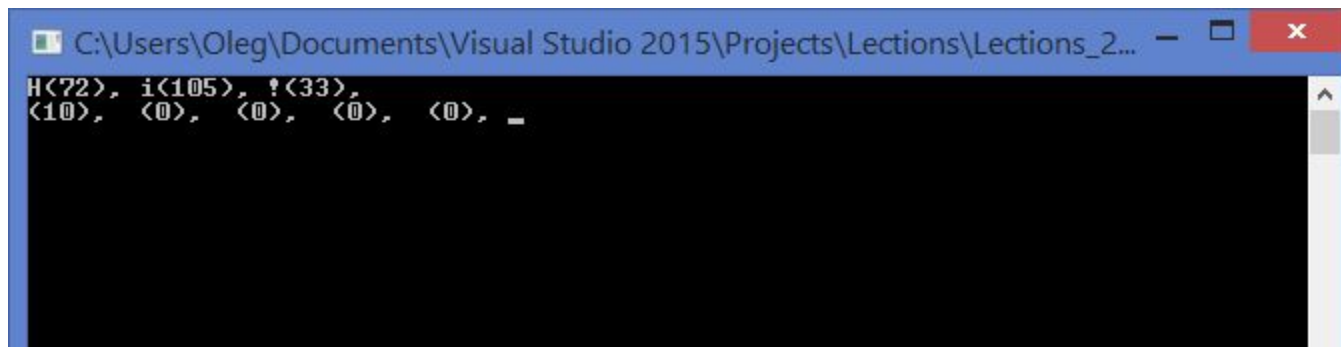


```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lectons\Lectons_2...  
H<72>, i<105>, !<33>,  
<10>, <0>, <0>, <0>, <0>, _
```

```
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]);  
    }  
}
```

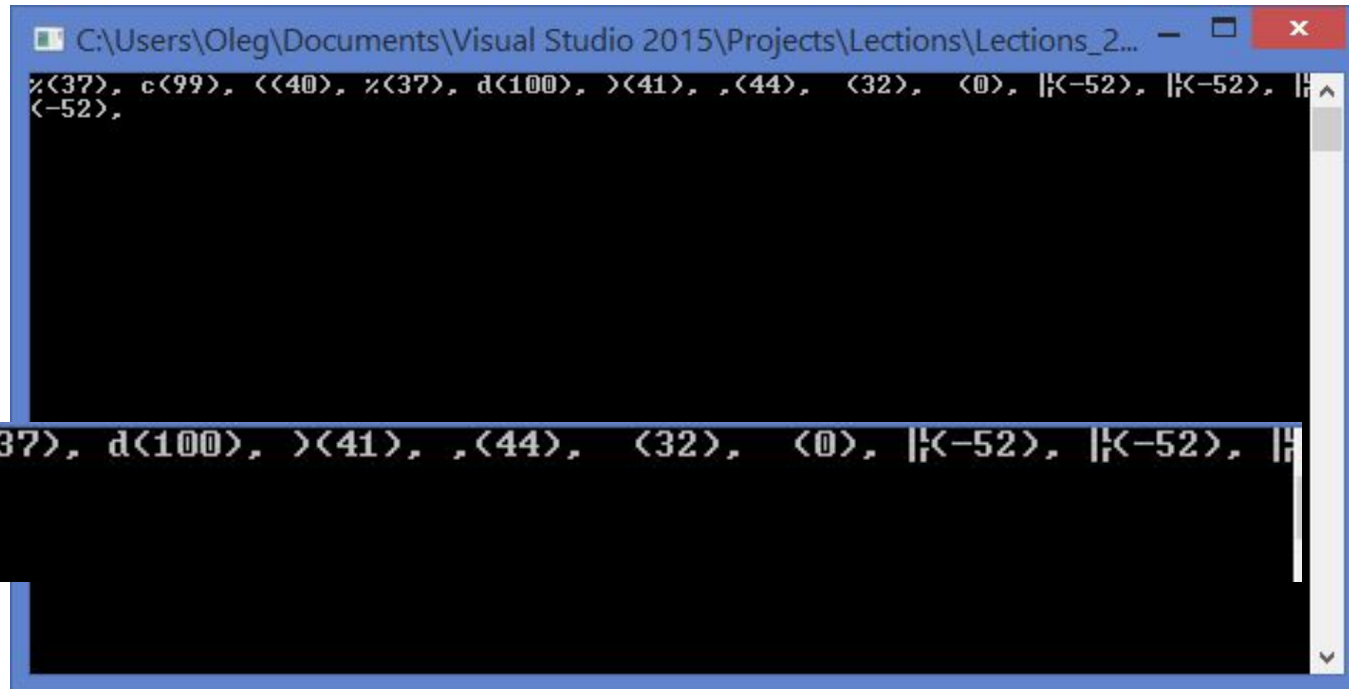


```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lectons\Lectons_2...  
H(72), i(105), !(33),  
<10>, <0>, <0>, <0>, <0>, _
```

```
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]);  
    }  
}
```



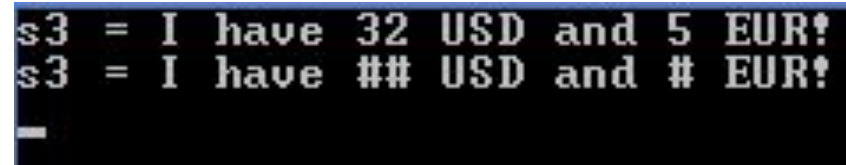
```
C:\Users\Oleg\Documents\Visual Studio 2015\Projects\Lectons\Lectons_2...  
%(37), c(99), <<(40), %(37), d(100), >(41), ,(44), <(32), <(0), |<(-52), |<(-52), |<(-52),
```

# 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] = '#';
```

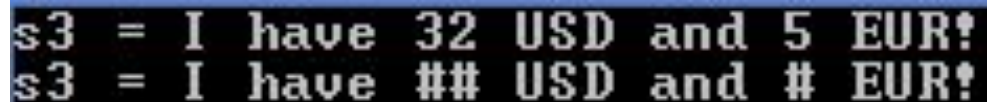
```
        }
```

```
        i++;
```

```
    }
```

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

```
}
```

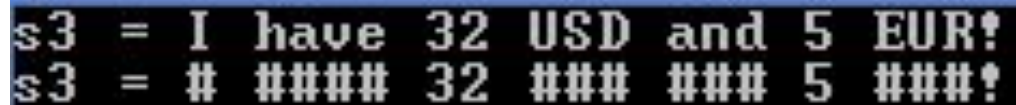


```
s3 = I have 32 USD and 5 EUR!  
s3 = I have ## USD and # EUR!  
_
```

# 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);  
}
```



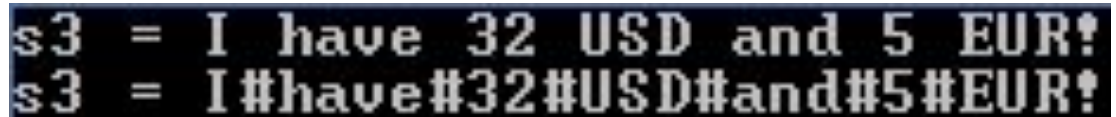
The image shows a terminal window with a black background and white text. It displays the output of the C program. The first line shows the original string: "s3 = I have 32 USD and 5 EUR!". The second line shows the string after replacing all alphabetic characters with '#' symbols: "s3 = # ##### 32 #### ## 5 ####!".

```
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);  
}
```



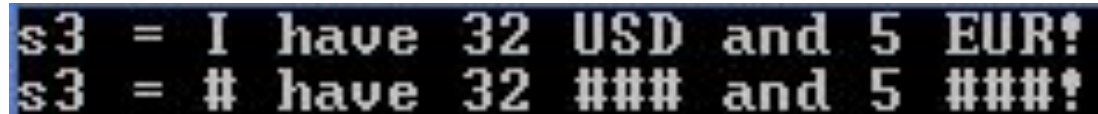
```
s3 = I have 32 USD and 5 EUR!  
s3 = I#have#32#USD#and#5#EUR!
```



# 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);  
}
```

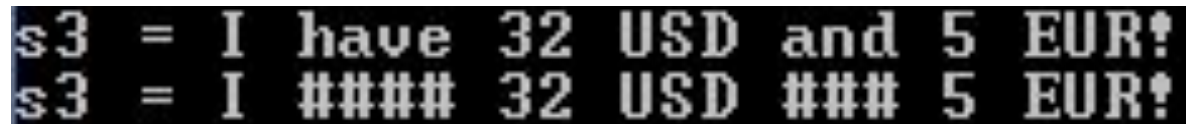


```
s3 = I have 32 USD and 5 EUR!  
s3 = # have 32 ### and 5 ###!
```

# 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);  
}
```

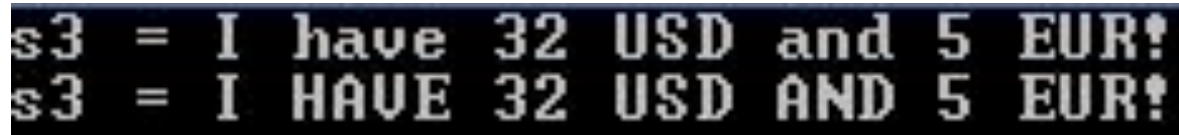


```
s3 = I have 32 USD and 5 EUR!  
s3 = I #### 32 USD ### 5 EUR!
```

# 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);  
}
```

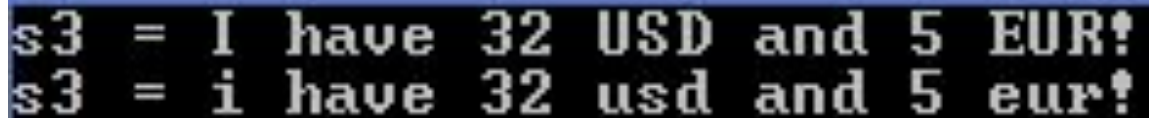


```
s3 = I have 32 USD and 5 EUR!  
s3 = I HAVE 32 USD AND 5 EUR!
```

# 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';
```

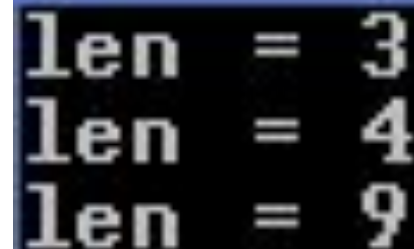
```
    printf("len = %d\n", strlen(s));
```

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

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

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

```
}
```



```
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);  
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);  
}
```

A terminal window with a black background and white text. It displays the output of the program: the first line is "max = We" and the second line is "min = Apple !!".

```
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;  
}
```

# [Домашнее] задание

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