Основы программирования

ФИСТ 1 курс

Власенко
Олег
Федосович

Лекция 12

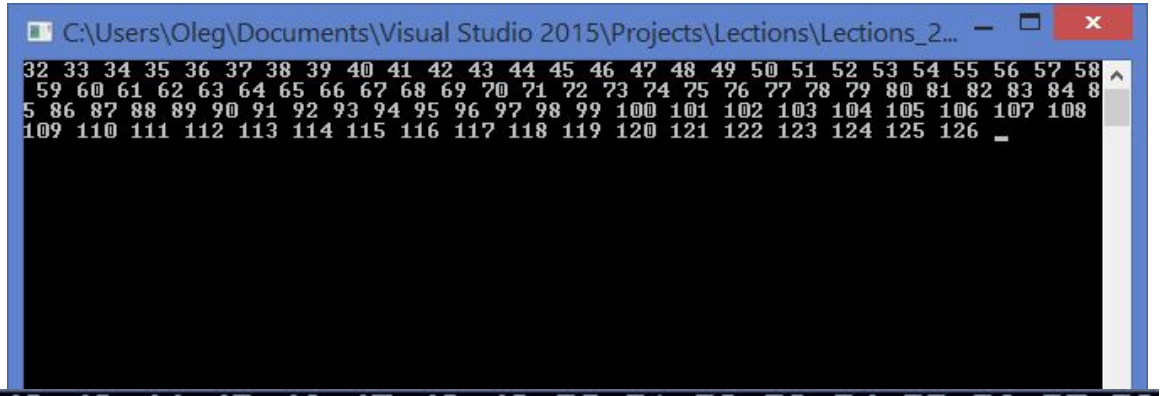
Символы и строки в Си.

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++;
    }
}
```

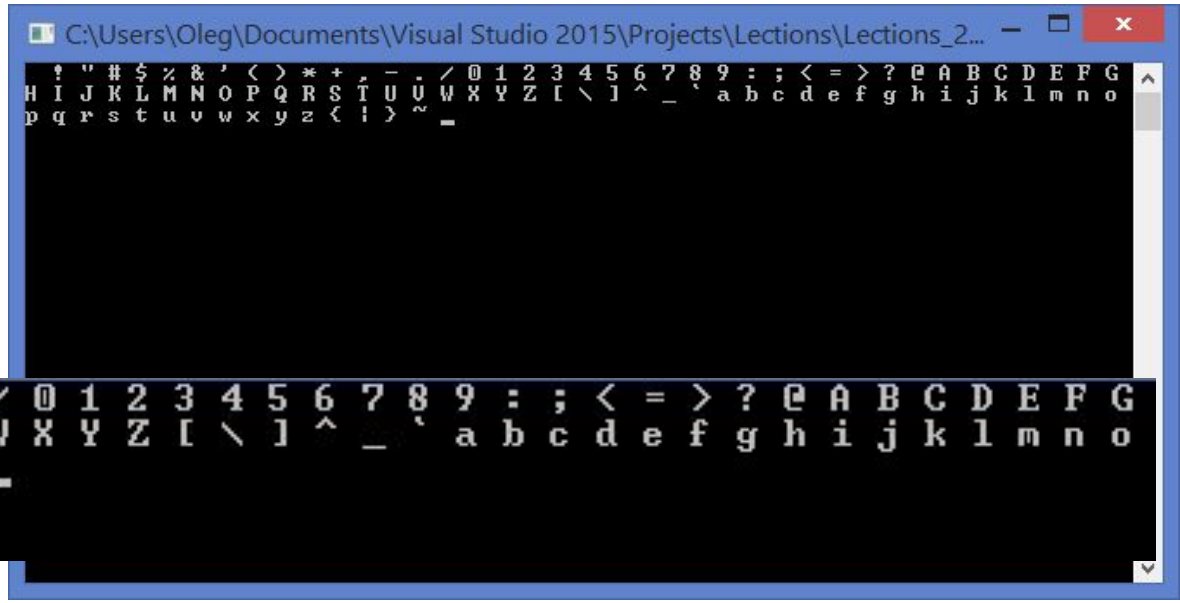


32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58
59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 8
5 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123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

Тип char (2)

char – это символьный тип

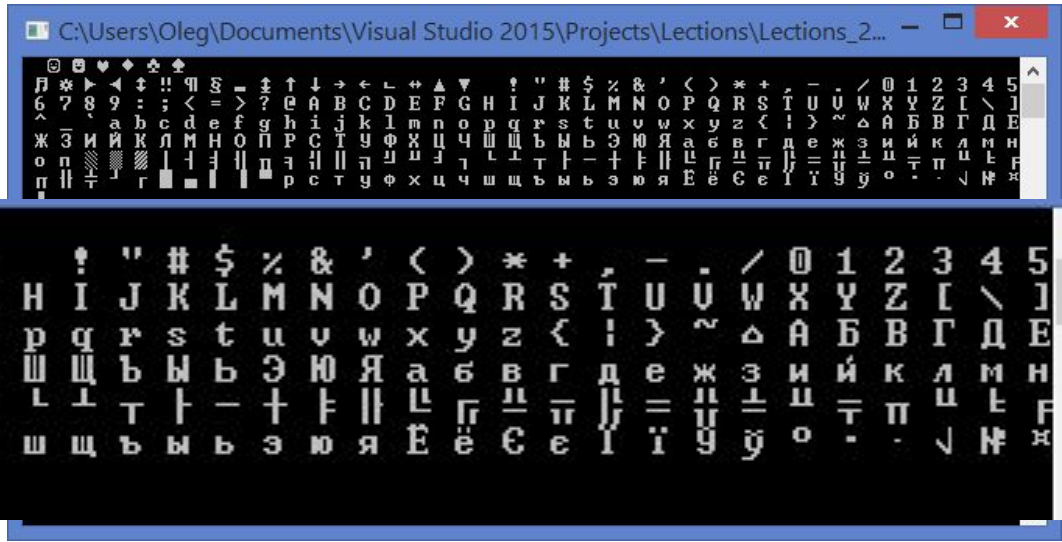
```
#include <stdio.h>

void main() {

    char ch = 32;
    while (ch < 127) {
        printf("%c ", ch);
        ch++;
    }
}
```



Тип char (3)

unsigned char = [0 .. 255]

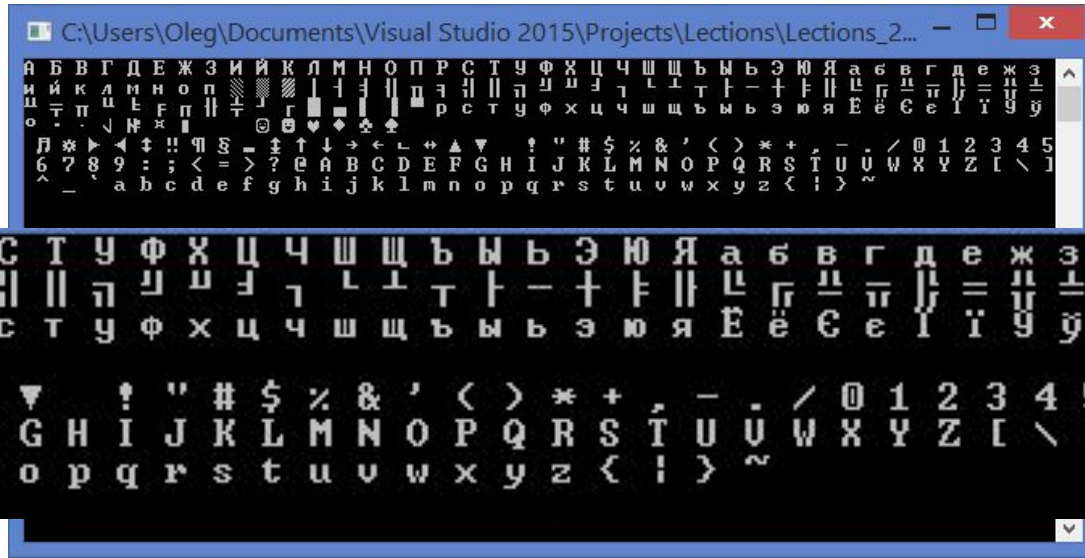
```
#include <stdio.h>

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. ***A**merican **s**tandard **c**ode for **i**nformation **i**nterchange*) — 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

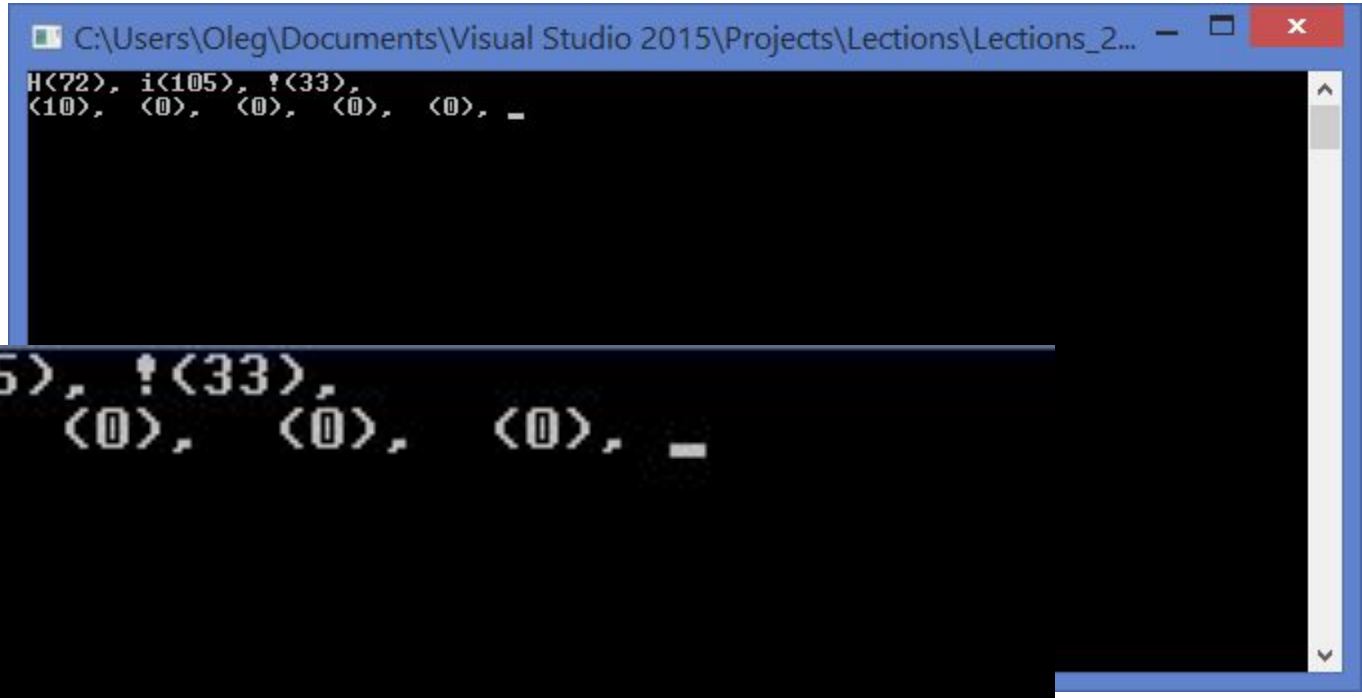
ASCIIZ

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**

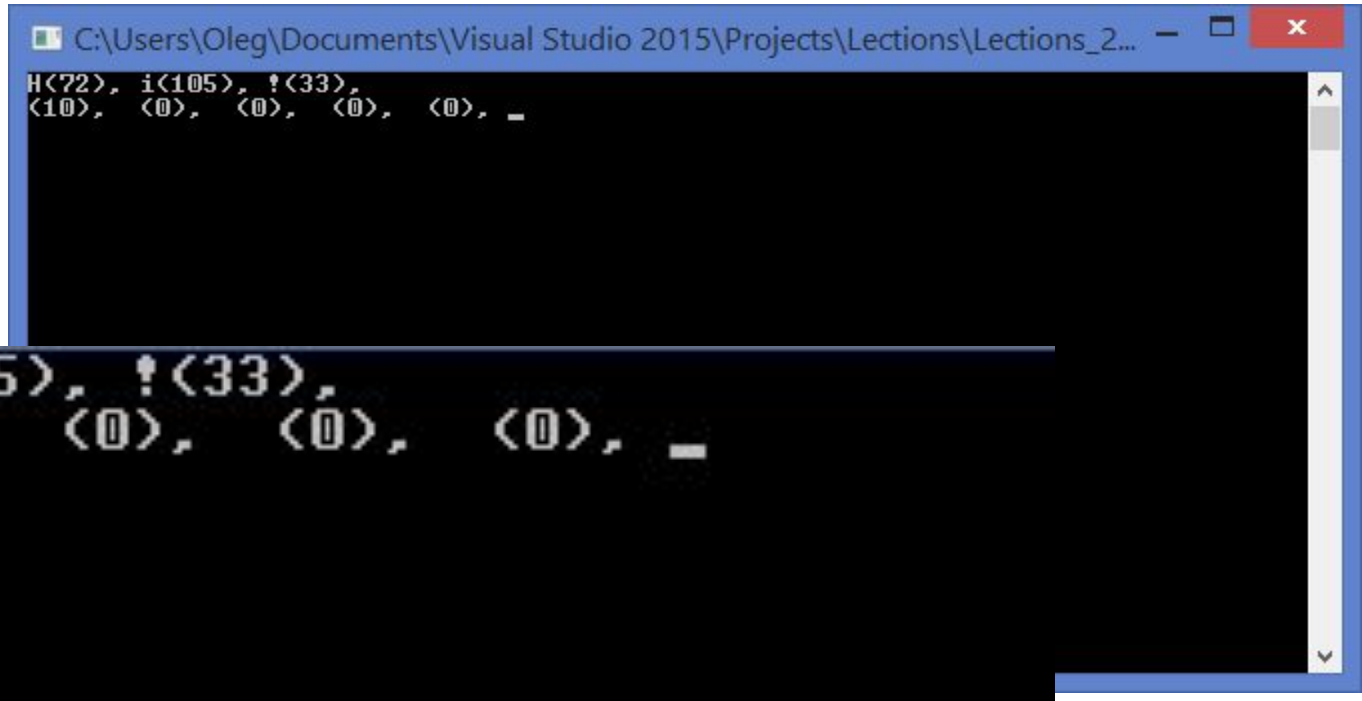
null-terminated string

```
void main() {
	char s1[8] = "Hi!\n";

	int i;
	for (i = 0; i < 8; i++) {
		printf("%c(%d), ", s1[i], s1[i]);
	}
}
```



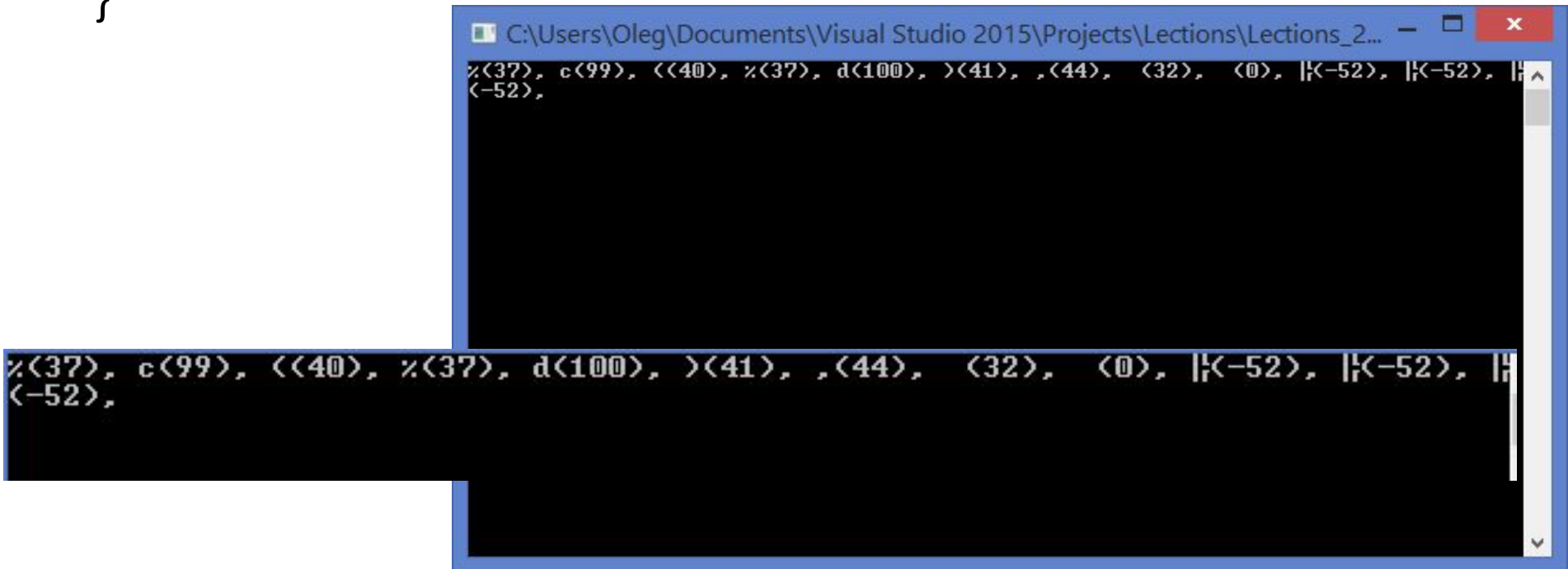
Инициализация строки как массива символов

```
void main() {
	char s1[8] = { 'H', 'i', '!', '\n', '\0' };

	int i;
	for (i = 0; i < 8; i++) {
		printf("%c(%d), ", s1[i], s1[i]);
	}
}
```



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]);
	}
}
```



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

Все цифры заменить на символ «#»

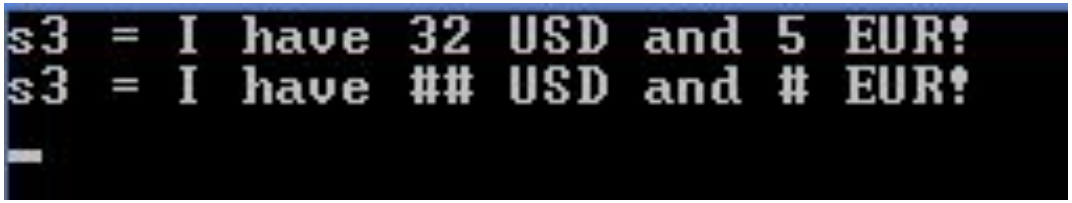
```
#include <stdio.h>
void main() {
    char s3[] = "I have 32 USD and 5 EUR!";
    printf("s3 = %s\n", s3);
    int i = 0;
    while (s3[i] != '\0') {
        if (s3[i] >= '0' && s3[i] <= '9') {
            s3[i] = '#';
        }
        i++;
    }
    printf("s3 = %s\n", s3);
}
```

```
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);
}
```



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 сроку src в dest, включая завершающий ‘\0’

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

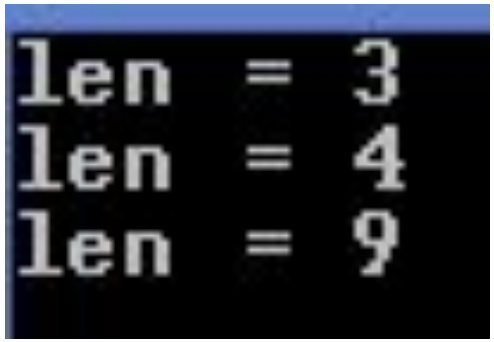
И еще около 20 функций из string.h

strlen()

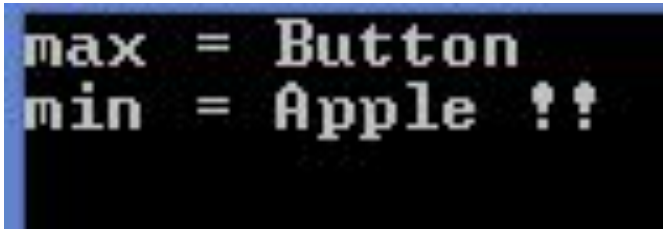
```
#include <string.h>
void main() {
    char s[10] = "Hi!";
    printf("len = %d\n", strlen(s));

    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));
}
```



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);
}
```

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.

Значение	*Результат сравнения строк*
Меньше нуля	str1 меньше str2
Нуль	str1 равен str2
Больше нуля	str1 больше str2

Сравнение строк через strcmp

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

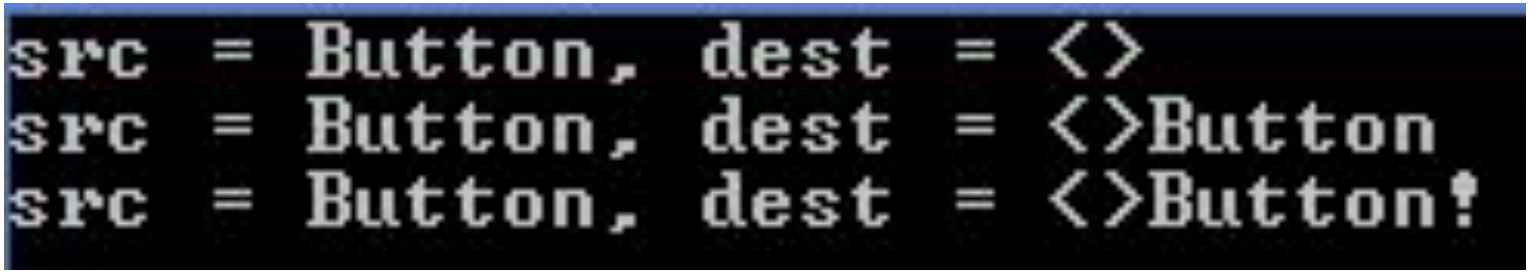
```
void main() {
    char s1[] = "Button";
    char s2[] = "We";
    char s3[] = "Apple !!";
    char * min = s1;char * max = s1;

    if (strcmp(s2, max) > 0) max = s2;
    if (strcmp(s2, 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);
}
```

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);

}
```

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);
}
```



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.

Значение	*Результат сравнения строк*
Меньше нуля	str1 меньше str2
Нуль	str1 равен str2
Больше нуля	str1 больше str2

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];
}
```

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

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/xssktc6e.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
http://altaev-aa.narod.ru/security/Cesar.html
ROT13 - https://ru.wikipedia.org/wiki/ROT13

Шифрование одной строки

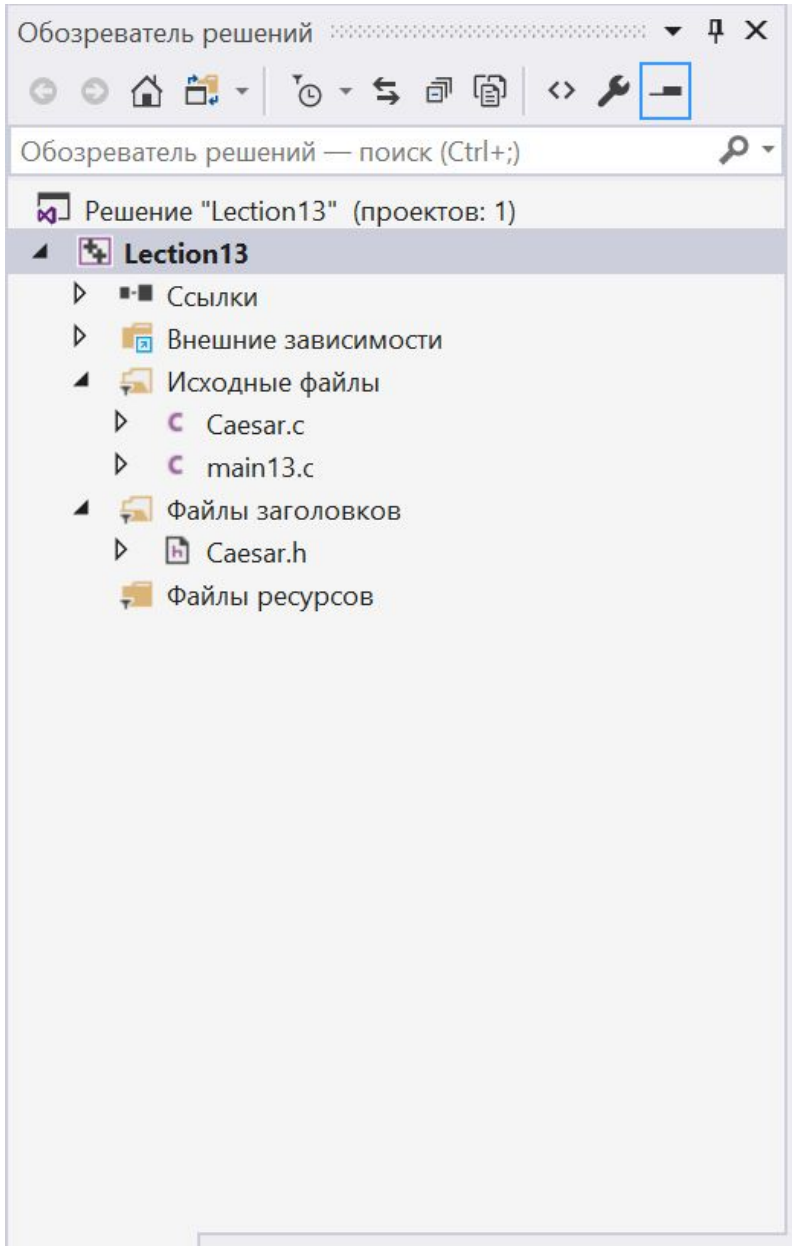
```
#define MAX_LEN 80

#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);
	}
}
```

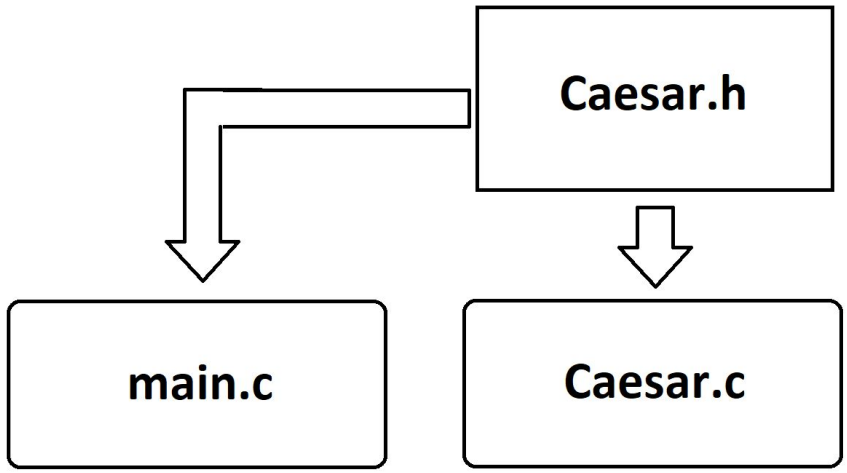
Шифрование одного символа

```
int encode(int ch, int key) {
    //char smallLetters[] ="abcdefghijklmnopqrstuvwxyz";
    //char bigLetters[] ="ABCDEGGHIJKLMNOPQRSTUVWXYZ";

    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;
}
```

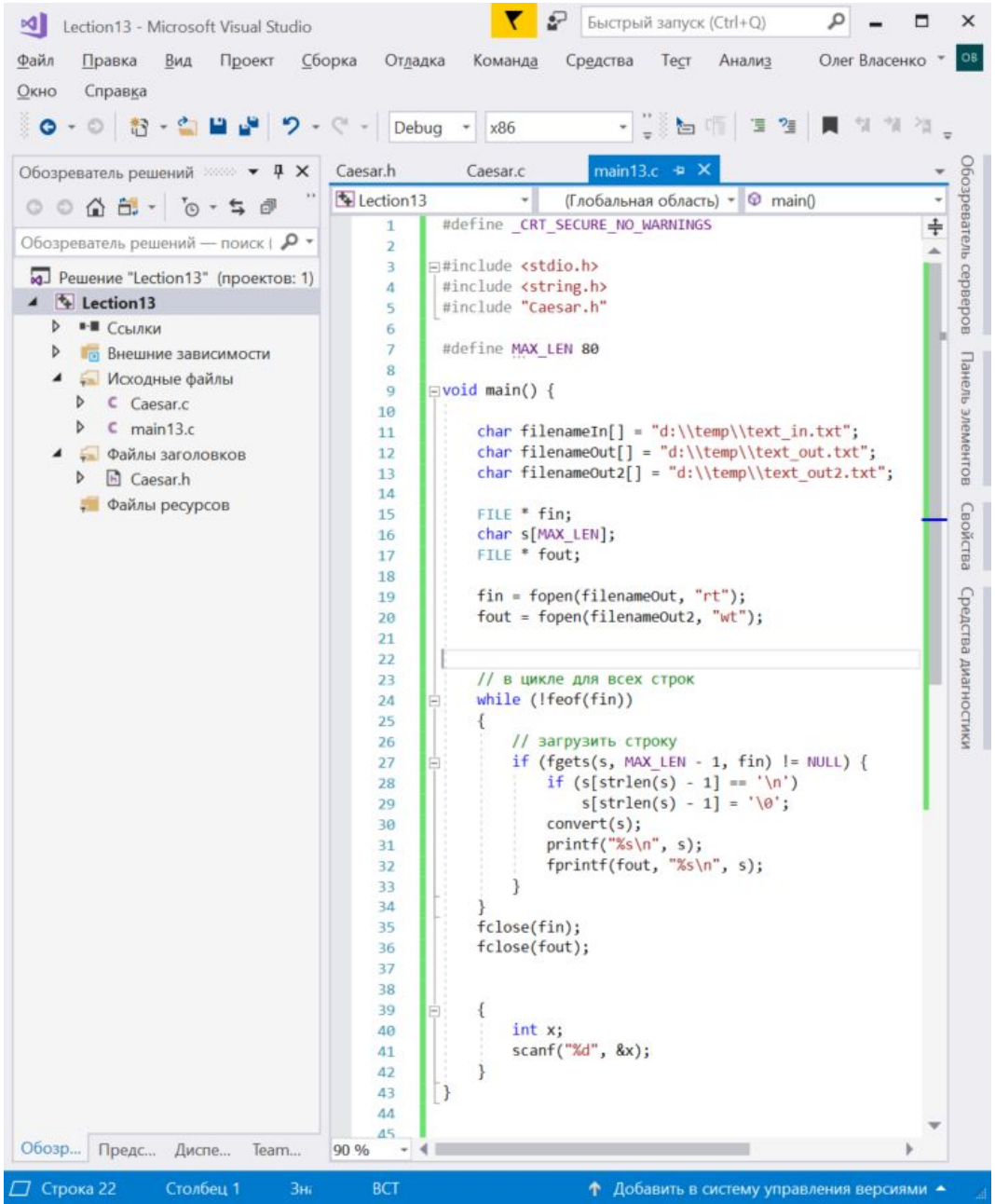
Многомодульные проекты

Структура многомодульного проекта





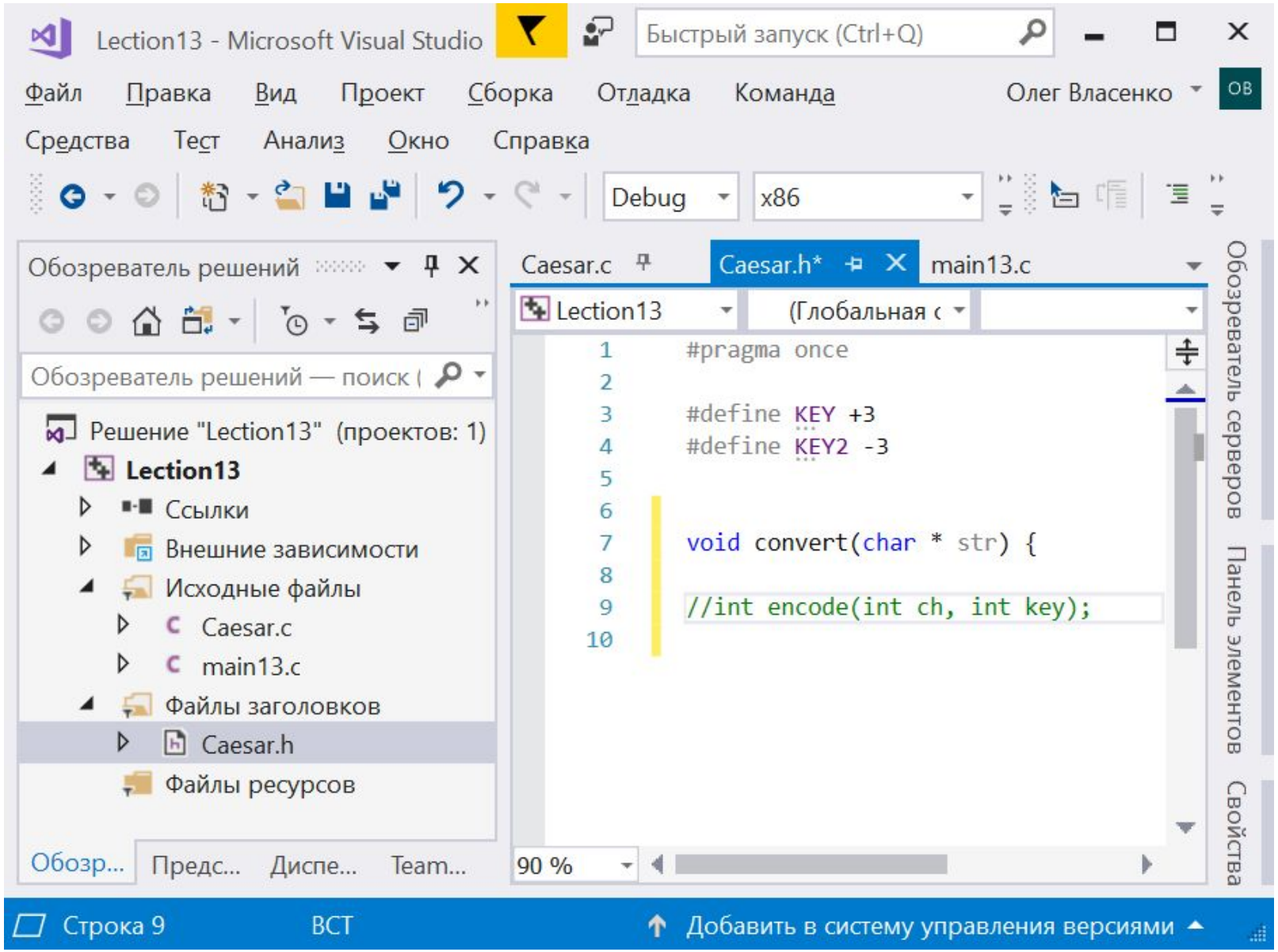
Main.c


Lection13 - Microsoft Visual Studio
Быстрый запуск (Ctrl+Q)
Файл Правка Вид Проект Сборка Отладка Команда Средства Тест Анализ Олег Власенко
Окно Справка
Debug
x86
Обозреватель решений
Обозреватель решений — поиск
Решение "Lection13" (проектов: 1)
Lection13
Ссылки
Внешние зависимости
Исходные файлы
Caesar.c
main13.c
Файлы заголовков
Caesar.h
Файлы ресурсов
Caesar.h
Caesar.c
main13.c
Lection13
(Глобальная область)
main()
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
#include "Caesar.h"
#define MAX_LEN 80
void main() {
char filenameIn[] = "d:\\temp\\text_in.txt";
char filenameOut[] = "d:\\temp\\text_out.txt";
char filenameOut2[] = "d:\\temp\\text_out2.txt";
FILE * fin;
char s[MAX_LEN];
FILE * fout;
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);
fprintf(fout, "%s\n", s);
}
}
fclose(fin);
fclose(fout);
{
int x;
scanf("%d", &x);
}
}
Обозр... Предс... Диспе... Team...
90 %
Строка 22 Столбец 1 Зн ВСТ
Добавить в систему управления версиями
Обозреватель серверов
Панель элементов
Свойства
Средства диагностики

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);
                }
        }
```

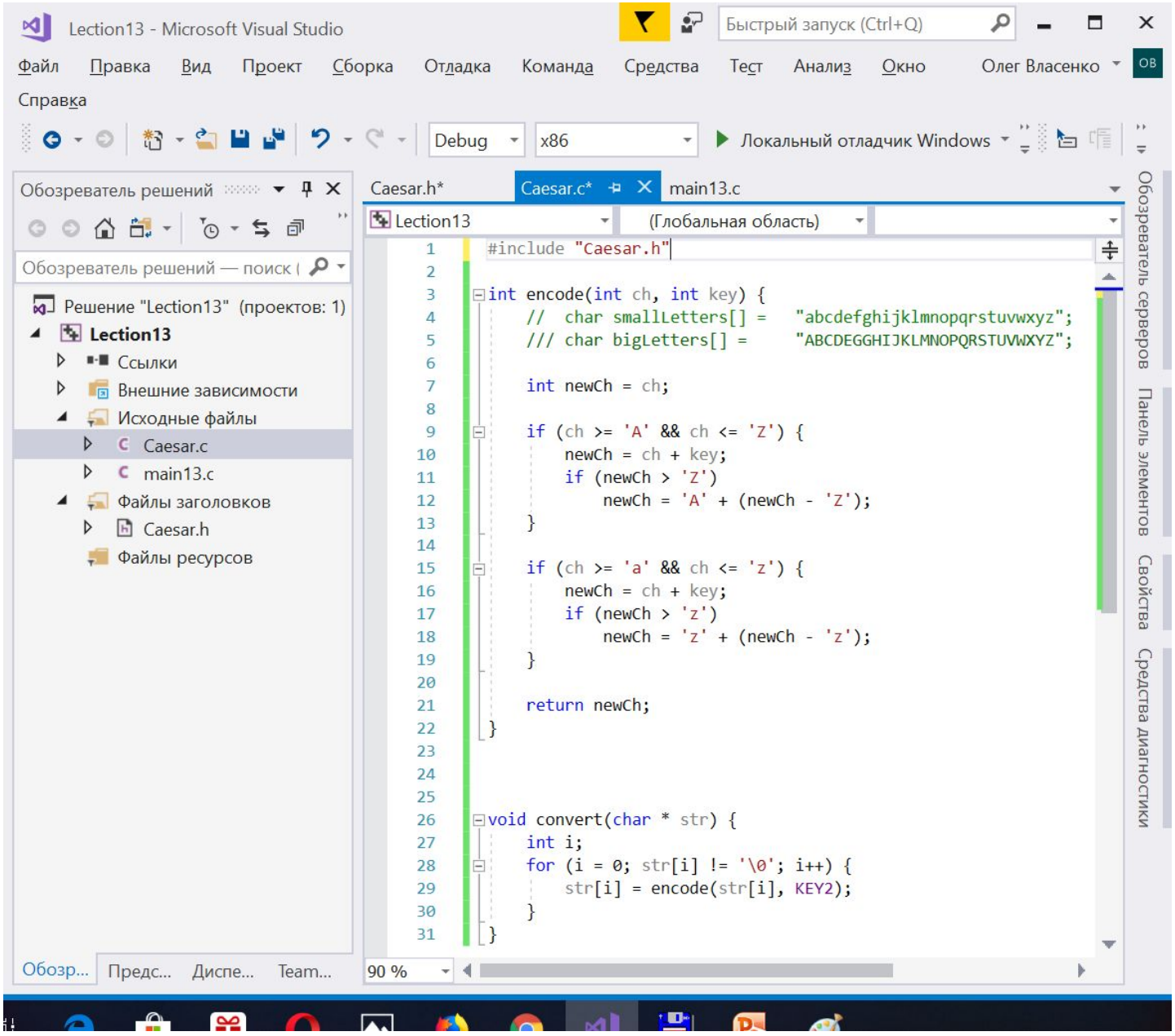
Caesar.h


Lection13 - Microsoft Visual Studio
Быстрый запуск (Ctrl+Q)
Файл Правка Вид Проект Сборка Отладка Команда Олег Власенко
Средства Тест Анализ Окно Справка
Debug
x86
Обозреватель решений
Обозреватель решений — поиск
Решение "Lection13" (проектов: 1)
Lection13
Ссылки
Внешние зависимости
Исходные файлы
Caesar.c
main13.c
Файлы заголовков
Caesar.h
Файлы ресурсов
Обозр...
Предс...
Диспе...
Team...
Caesar.c
Caesar.h*
main13.c
Lection13
(Глобальная с
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
90 %
Обозреватель серверов
Панель элементов
Свойства
Строка 9
ВСТ
Добавить в систему управления версиями

Caesar.h

```
#pragma once

#define KEY +3
#define KEY2 -3

void convert(char * str);

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

Caesar.c


Lection13 - Microsoft Visual Studio
Быстрый запуск (Ctrl+Q)
Файл Правка Вид Проект Сборка Отладка Команда Средства Тест Анализ Окно Олег Власенко
Справка
Debug
x86
Локальный отладчик Windows
Обозреватель решений
Обозреватель решений — поиск
Решение "Lection13" (проектов: 1)
Lection13
Ссылки
Внешние зависимости
Исходные файлы
Caesar.c
main13.c
Файлы заголовков
Caesar.h
Файлы ресурсов
Обозр... Предс... Диспе... Team...
Caesar.h*
Caesar.c*
main13.c
Lection13
(Глобальная область)
#include "Caesar.h"
int encode(int ch, int key) {
// char smallLetters[] = "abcdefghijklmnopqrstuvwxyz";
/// char bigLetters[] = "ABCDEGGHIJKLMNOPQRSTUVWXYZ";
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;
}
void convert(char * str) {
int i;
for (i = 0; str[i] != '\0'; i++) {
str[i] = encode(str[i], KEY2);
}
}
90 %
Обозреватель серверов
Панель элементов
Свойства
Средства диагностики

Caesar.c

```
#include "Caesar.h"

int encode(int ch, int key) {
	//char smallLetters[] ="abcdefghijklmnopqrstuvwxyz";
	//char bigLetters[] ="ABCDEGGHIJKLMNOPQRSTUVWXYZ";

	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)) {
```

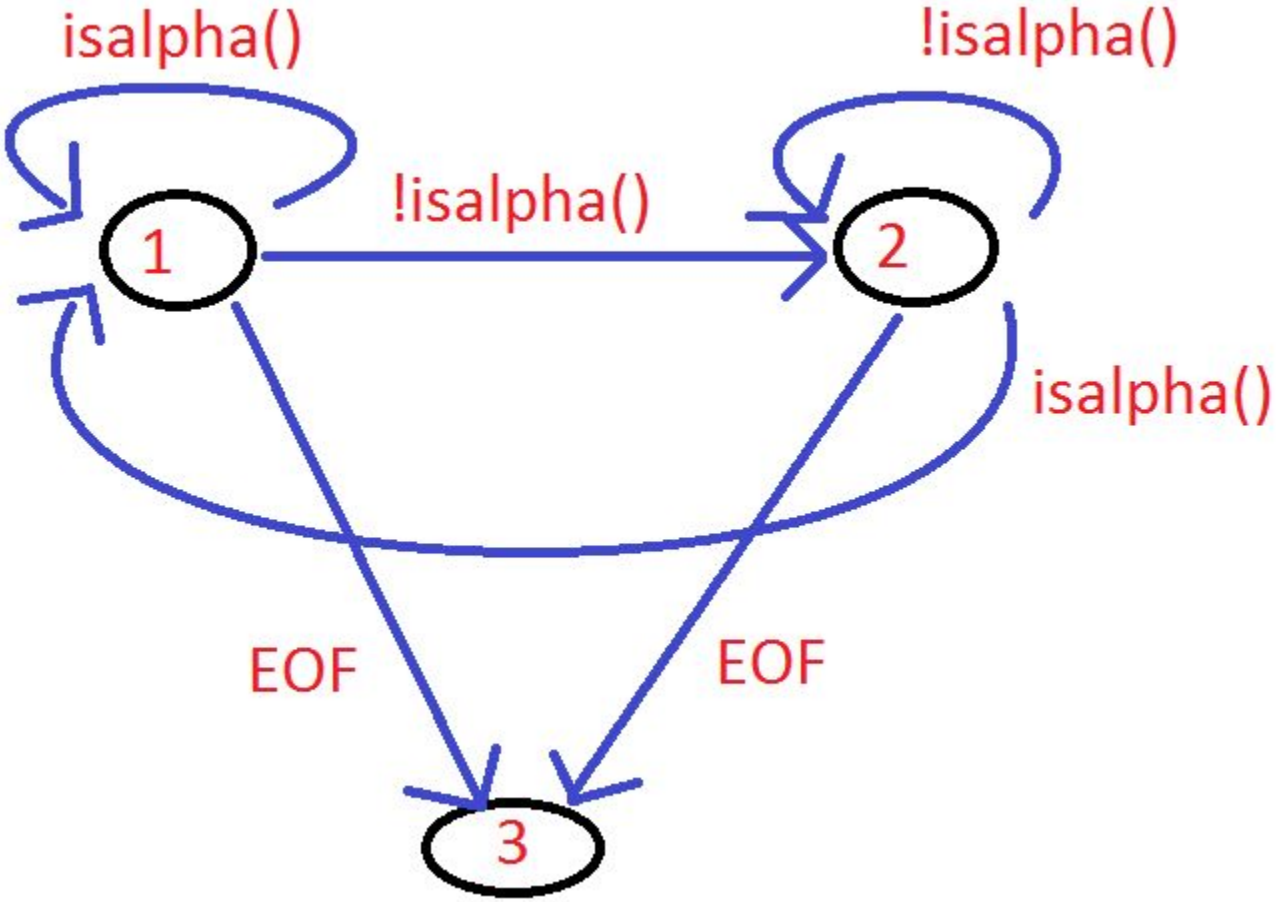
Код программы (3)

```
    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)
```

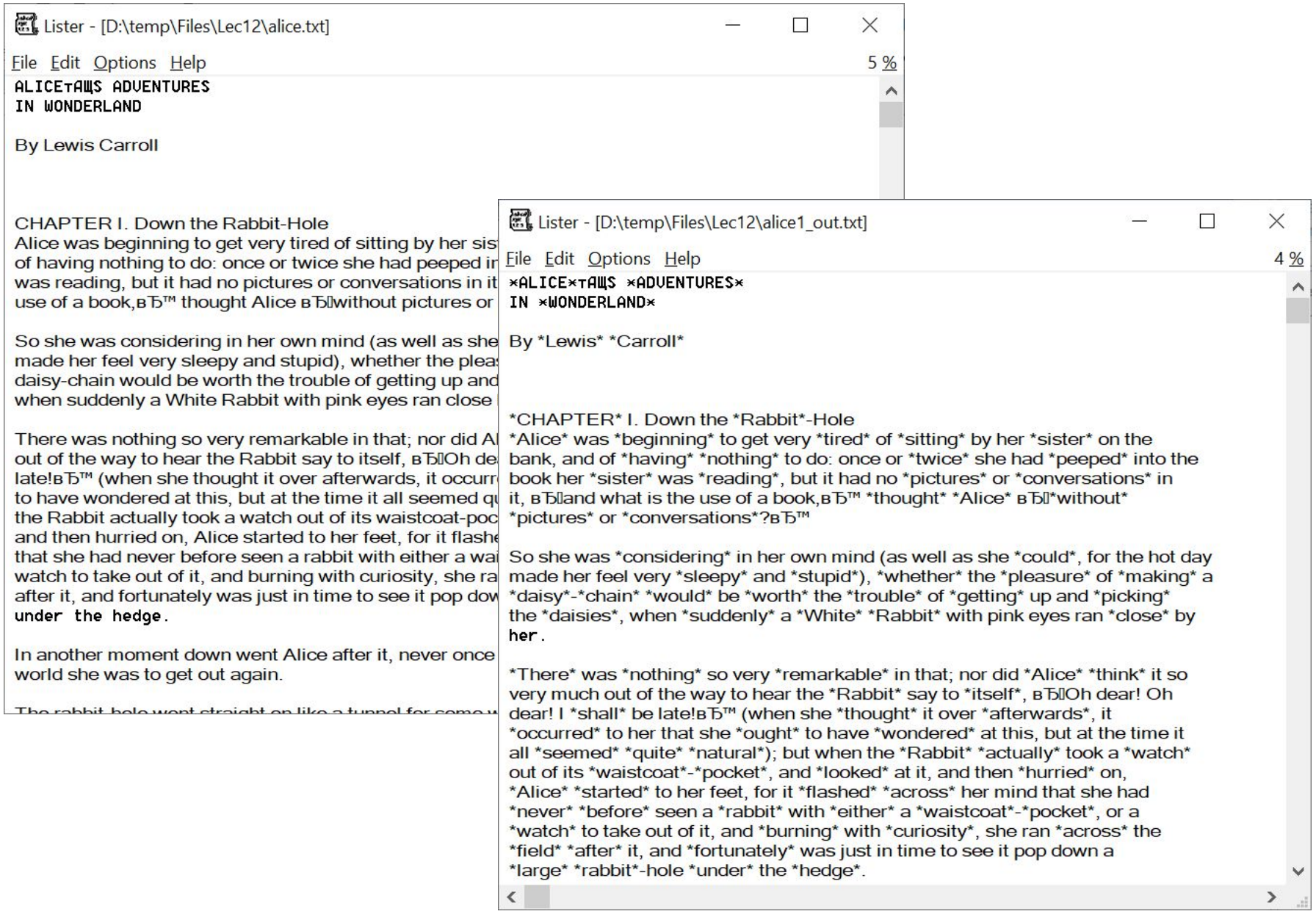
Код программы (4)

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

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

Граф состояний



Итог работы программы



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",
		"Заменив 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/russkie-simvolybukvy-pri-vvodevyvode-v-konsol-na-c/
- Msdn
- Википедия