



# Строки и структуры

Лекция 4

# Вопрос 1

## **Строковые переменные**

## Понятие строки

- ❑ Строка представляет собой массив символов, заканчивающийся нуль-символом.
- ❑ Нуль-символ – это символ с кодом, равным 0, что записывается в виде управляющей последовательности ‘\0’.
- ❑ По положению нуль-символа определяется фактическая длина строки.

## Объявление строковой переменной

```
char str[10] = "My text";
```

- ❑ В этом примере под строку выделяется 10 байт, 7 из которых занято под символы строки, а восьмой – под нуль-символ.
- ❑ Если строка при определении инициализируется, ее размерность можно опускать:

```
char str[] = "Program"; // выделено и заполнено  
// 8 байт
```

## Присваивание строк

Операция присваивания одной строки другой не определена (поскольку строка является массивом) и может выполняться с помощью цикла или функций стандартной библиотеки **<string.h>**.

## Функция `strcat`

`strcat(s1, s2)` – функция добавляет `s2` к `s1` и возвращает `s1`.

Например:

```
char s1[10]="New";
```

```
char s2[10]="Text";
```

```
strcat(s1,s2);
```

В переменной `s1` окажется значение `NewText`.

## Функция `strcpy`

`strcpy(s1, s2)` – функция копирует `s2` в `s1`  
и возвращает `s1`

Например:

```
char s1[10]="New";
```

```
char s2[10]="Text";
```

```
strcpy(s1,s2);
```

В переменной `s1` окажется значение `Text`.

## Функция `strcmp`

- ❑ `strcmp(s1,s2)` – сравнивает строки `s1` и `s2`.
- ❑ Функция возвращает положительное (если `s1` больше `s2`), нулевое (если `s1` равно `s2`) или отрицательное (если `s1` меньше `s2`).
- ❑ Строки считаются равными когда они полностью совпадают по составу символов.
- ❑ Строка считается больше другой, если в ней раньше встречается символ с кодом больше.
- ❑ Если одна строка полностью совпадает с началом другой, то большей считается более длинная строка.



## Функция `strlen`

**`strlen(s)`** – функция возвращает длину строки (при этом символ конца строки не учитывается).

## Прочие функции

- ❑ **strncat(s1, s2, n)** – функция добавляет не более **n** символов из **s2** к **s1** и возвращает **s1**.
- ❑ **strncpy(s1, s2, n)** – функция копирует не более **n** символов из **s2** в **s1** и возвращает **s1**.

## Пример

**Задание.** Пользователь вводит строку размером до 30 символов и еще один отдельный символ. Подсчитать, сколько раз этот символ встречается в строке, введенной пользователем.

```
#include "stdafx.h"  
#include <conio.h>  
#include <string.h>  
#include <locale.h>  
#include <iostream>  
using namespace std;
```

## Текст программы

```
int _tmain(int argc, _TCHAR* argv[])
{
    setlocale(LC_STYPE, "");
    char str[30];
    char s;

    cout<<"Введите строку\n";
    gets(str);
    cout<<"Введите символ\n";
    cin>>s;
```

## Текст программы

```
int k=0;
    for (int i=0; i<strlen(str); i++)
        if (str[i]==s)
            k++;
cout<<"Количество вхождений символа
"<<s<<" в строку \n"<<str<<"\n равно
"<<k<<"\n";
```

## Текст программы

```
cout<<"Для окончания работы нажмите любую  
клавишу...";  
getch();  
return 0;  
}
```



Вопрос 2

**Структуры**

## Понятие и вид структуры

В отличие от массива, все элементы которого однотипны, структура может содержать элементы разных типов.

Структура задается следующим образом:

```
struct [ имя_типа ] {  
    тип_1 элемент_1;  
    тип_2 элемент_2;  
    ...  
    тип_n элемент_n;  
} [ список_переменных ];
```



## Элементы структуры

- ❑ Элементы структуры называются *полями структуры* и могут иметь любой тип, кроме типа этой же структуры, но могут быть указателями на него.
- ❑ Описание структуры определяет новый тип, имя которого можно использовать в дальнейшем наряду со стандартными типами.

## Пример задания структуры

```
struct Worker{ // описание нового типа
    // Worker
    char fio[30];
    int age, code;
    double zarpl;
}; // описание заканчивается точкой с запятой

// определение массива типа Worker и
// указателя на тип Worker:
Worker mas_struct[100];
```

## Пример задания структуры

Можно не определять отдельный тип – тогда после описания структуры может идти список переменных, массивов или указателей. Например:

```
struct { // описание нового типа
    char fio[30];
    int age, code;
    double zarpl;
} mas_struct[100], zap;
```

## Инициализация структуры

Для инициализации структуры значения ее элементов перечисляют в фигурных скобках в порядке их описания:

```
struct {  
    char fio[30];  
    int age, code;  
    double zarpl;  
} worker = {"Миронов", 31, 215,  
18400.50};
```

## Доступ к полям структуры

### Доступ к полям структуры

выполняется с помощью операций выбора `.` (точка) при обращении к полю через имя структуры, например:

```
Worker worker, mas_struct[100];
```

```
worker.fio = "Миронов";
```

```
mas_struct[8].code = 215;
```

## Пример

### Задание.

Заполнить массив из 7 семи записей следующей структуры:

- ✓ ФИО студента;
- ✓ курс;
- ✓ группа;
- ✓ оценка по математике;
- ✓ оценка по программированию;
- ✓ оценка по физике.

## Пример

Вывести на экран массив записей в виде таблицы следующего вида:

ФИО	Курс	Группа	Матем.	Прог.	Физ.	Ср. балл

## Текст программы

```
#include "stdafx.h"  
#include <stdio.h>  
#include <conio.h>  
#include <locale.h>  
#include <math.h>  
#include <iostream>  
using namespace std;
```



## Текст программы

```
int _tmain(int argc, _TCHAR* argv[])
{
    const int n=7;
    const int m=77;
    setlocale(LC_CTYPE, "");
    struct stud {
        char fio[30];
        int kurs;
        char gruppa[10];
        int math, prog, fiz;
    };
    stud mas[n];
```

## Текст программы

```
cout<<"Введите "<<n<<" записей\n";  
for (int i=0; i<n; i++)  
{  
    cout<<"Введите ФИО студента\n";  
    fflush(stdin);  
    gets(mas[i].fio);  
    cout<<"Введите курс\n";  
    cin>>mas[i].kurs;  
    cout<<"Введите группу\n";  
    fflush(stdin);  
    gets(mas[i].gruppa);
```

## Текст программы

```
cout<<"Введите оценку по математике\n";  
cin>>mas[i].math;  
cout<<"Введите оценку по  
программированию\n";  
cin>>mas[i].prog;  
cout<<"Введите оценку по физике\n";  
cin>>mas[i].fiz;  
}
```

## Текст программы

```
printf("          Студенты\n");  
for (int j=0; j<m; j++)  
    printf("-");  
printf("\n");  
printf("|          ФИО          | Курс | Группа |  
Матем. | Прог. | Физ. | Ср. балл |\n");  
for (int j=0; j<m; j++)  
    printf("-");  
printf("\n");
```

## Текст программы

```
float sr_ball;
for (int i=0; i<n; i++)
{
    sr_ball=(mas[i].math+mas[i].prog+mas[i].fiz)/3.0;
    printf("| %20s | %4d | %8s | %6d | %5d |
           %4d | %8.2f |\n", mas[i].fio,
    mas[i].kurs, mas[i].gruppa,
           mas[i].math, mas[i].prog, mas[i].fiz,
           sr_ball);
}
```

## Текст программы

```
for (int j=0; j<m; j++)  
    printf("-");  
printf("\n");  
printf("\nДля окончания работы нажмите  
любую клавишу...\n");  
getch();  
  
return 0;  
}
```