

Строки и символы в С и С++

- Символ – элементарная единица, некоторый набор которых несет определенный смысл.
- В языке программирования С++ предусмотрено использование символьных констант.
- Символьная константа – это целочисленное значение (типа `int`) представленное в виде символа, заключённого в одинарные кавычки, например `'a'`.
- В таблице ASCII представлены символы и их целочисленные значения.

Строки в языках C и C++

```
// объявления символьной переменной  
char symbol = 'a';  
// где symbol - имя переменной типа char  
// char - тип данных для хранения символов
```

- Строки в C++ представляются как массивы элементов типа char, заканчивающиеся нуль-терминатором \0 называются C строками или строками в стиле C.

\0 — символ нуль-терминатора.

- Символьные строки состоят из набора символьных констант заключённых в двойные кавычки.
- При объявлении символьного массива необходимо учитывать наличие в конце строки нуль-терминатора, и отводить дополнительный байт под него.

```
// пример объявления строки  
char string[10];  
// где string - имя строковой переменной  
// 10 - размер массива, в данной строке может поместиться  
// 9 символов, последнее место отводится под \n
```

Строки и символы в С и С++

- Строка при объявлении может быть инициализирована начальным значением, например, так:

```
char string[10] = "abcdefghf";
```

- Количество символов в двойных кавычках - 9, а размер строки 10 символов, последнее место отводится под '\0', причём компилятор сам добавит его в конец строки.

```
// посимвольная инициализация строки:
```

```
char string[10] =
```

```
    {'a','b','c','d','e','f','g','h','f','\0'};
```

```
// десятый символ это ноль-терминатор.
```

Строки и символы в С и С++

- При объявлении строки необязательно указывать её размер, но при этом обязательно нужно её инициализировать начальным значением.
- Тогда размер строки определится автоматически и в конец строки добавится нуль-терминатор.

```
//инициализация строки без указания размера  
char string[] = "abcdefghf";  
//всё то же самое только размер не указываем.
```

Строки и символы в С и С++

- Строка может содержать символы, цифры и специальные знаки.
- В С++ строки заключаются в двойные кавычки.
- Имя строки является константным указателем на первый символ.
- Разработаем программу, с использованием строк.

```
#include <iostream>

using namespace std;

int main () {
    char name[] = "qwerty", line[500];
    cout << "Введите строку: ";
    cin.getline (line, 256);
    cout << "Присвоена строка: " << name << endl
         << "Вы ввели: " << line << endl;
    return 0;
}
```



НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ
КОРАБЛЕБУДУВАННЯ
ІМЕНІ АДМІРАЛА МАКАРОВА

Демонстрація



string.h

Функція	Пояснення
strlen (имя_строки)	определяет длину указанной строки, без учёта нуля-символа
Копирование строк	
strcpy (s1, s2)	выполняет побайтное копирование символов из строки s2 в строку s1
strncpy (s1, s2, n)	выполняет побайтное копирование n символов из строки s2 в строку s1. возвращает значение s1
Конкатенация строк	
strcat (s1, s2)	объединяет строку s2 со строкой s1. Результат сохраняется в s1
strncat (s1, s2, n)	объединяет n символов строки s2 со строкой s1. Результат сохраняется в s1

Некоторые строковые функции C++

Сравнение строк

<code>strcmp (s1, s2)</code>	сравнивает строку <code>s1</code> со строкой <code>s2</code> и возвращает результат типа <code>int</code> : 0 – если строки эквивалентны, >0 – если <code>s1<s2</code> , <0 – если <code>s1>s2</code> С учётом регистра
<code>strncmp (s1, s2)</code>	сравнивает <code>n</code> символов строки <code>s1</code> со строкой <code>s2</code> и возвращает результат типа <code>int</code> : 0 – если строки эквивалентны, >0 – если <code>s1<s2</code> , <0 – если <code>s1>s2</code> С учётом регистра
<code>stricmp (s1, s2)</code>	сравнивает строку <code>s1</code> со строкой <code>s2</code> и возвращает результат типа <code>int</code> : 0 – если строки эквивалентны, >0 – если <code>s1<s2</code> , <0 – если <code>s1>s2</code> Без учёта регистра
<code>strnicmp (s1, s2)</code>	сравнивает <code>n</code> символов строки <code>s1</code> со строкой <code>s2</code> и возвращает результат типа <code>int</code> : 0 – если строки эквивалентны, >0 – если <code>s1<s2</code> , <0 – если <code>s1>s2</code> Без учёта регистра

Некоторые строковые функции C++

Обработка символов <ctype.h>

<code>isalnum(c)</code>	возвращает значение true , если <code>c</code> является буквой или цифрой, и false в других случаях
<code>isalpha(c)</code>	возвращает значение true , если <code>c</code> является буквой, и false в других случаях
<code>isdigit(c)</code>	возвращает значение true , если <code>c</code> является цифрой, и false в других случаях
<code>islower(c)</code>	возвращает значение true , если <code>c</code> является буквой нижнего регистра, и false в других случаях
<code>isupper(c)</code>	возвращает значение true , если <code>c</code> является буквой верхнего регистра, и false в других случаях
<code>isspace(c)</code>	возвращает значение true , если <code>c</code> является пробелом, и false в других случаях
<code>toupper(c)</code>	если символ <code>c</code> , является символом нижнего регистра, то функция возвращает преобразованный символ <code>c</code> в верхнем регистре, иначе символ возвращается без изменений.

Некоторые строковые функции C++

Функции поиска

strchr (s, c)	поиск первого вхождения символа c в строке s. В случае удачного поиска возвращает указатель на место первого вхождения символа c. Если символ не найден, то возвращается ноль.
strcspn (s1, s2)	определяет длину начального сегмента строки s1, содержащего те символы, которые не входят в строку s2
strspn (s1, s2)	возвращает длину начального сегмента строки s1, содержащего только те символы, которые входят в строку s2
strprbk (s1, s2)	Возвращает указатель первого вхождения любого символа строки s2 в строке s1

Некоторые строковые функции C++

Функции преобразования `<stdlib.h>`

<code>atof (s1)</code>	преобразует строку <code>s1</code> в тип <code>double</code>
<code>atoi (s1)</code>	преобразует строку <code>s1</code> в тип <code>int</code>
<code>atol (s1)</code>	преобразует строку <code>s1</code> в тип <code>long int</code>

Функции стандартной библиотеки ввода/вывода `<stdio>`

<code>getchar (c)</code>	считывает символ <code>c</code> со стандартного потока ввода, возвращает символ в формате <code>int</code>
<code>gets (s)</code>	считывает поток символов со стандартного устройства ввода в строку <code>s</code> до тех пор, пока не будет нажата клавиша ENTER

Копирование строк

```
#include <iostream>
#include <cstring>
using namespace std;

int main()
{
    char s2[25] = "Computer Science forever";
    char s1[25];
    cout << "strcpy(s1,s2) = " << strcpy(s1,s2) << endl;
    cout << "s1=          " << s1 << endl;
    char s3[9];
    cout << strncpy(s3, s2, 8) << endl;
    return 0;
}
```

Демонстрация



Пример программы

Конкатенация строк

```
#include <iostream>
#include <cstring>
using namespace std;

int main()
{
    char s1[30] = "I am ";
    char s2[] = "programmer in the C++!!!!";
    cout << strcat(s1,s2) << endl;
    // объединяем строки s1 и s2, результат -> в s1
    char s3[23] = "I am a good ";
    cout << strncat(s3,s2,10) << "!!!" << endl;
    // объединяем 10 символов строки s2 со строкой s3
    return 0;
}
```

Конкатенация строк

Демонстрация



Сравнение строк

```
#include <iostream>
#include <iomanip>
#include <cstring>
using namespace std;

int main()
{
    char s1[] = "berkut.homelinux.com";
    char s2[] = "http://berkut.homelinux.com";
    cout << " s1 == s1 -> " << setw(2) << strcmp(s1,s1)
         << endl; // строка s1 = s1
    cout << " s1 < s2 -> " << setw(2) << strcmp(s1,s2)
         << endl; // строка s1 < s2
    cout << " s2 > s1 -> " << setw(2) << strcmp(s2,s1)
         << endl; // строка s2 > s1
    return 0;
}
```




Сравнение строк

Демонстрация



Пример программы

Обработка СИМВОЛОВ

```
#include <iostream>
using namespace std;
int main() {
    char symbol      = 'd'; // буква
    char digit       = '9'; // цифра
    char space       = ' '; // пробел
    cout << symbol    << " - is it a digit or an alpha?: ";
    cout << (isalnum(symbol) ? "true\n" : "false\n" );
    cout << symbol    << " - is it an alpha?                : ";
    cout << (isalpha(symbol) ? "true\n" : "false\n");
    cout << digit     << " - is it a digit?                  : ";
    cout << (isdigit(digit) ? "true\n": "false\n");
    cout << space     << " - is it a space?                  : ";
    cout << (isspace(space) ? "true\n": "false\n");
    cout << symbol    << " - is it a lower alpha?           : ";
    cout << (islower(symbol) ? "true\n": "false\n");
    cout << symbol    << " - is it an upper alpha?          : ";
    cout << (isupper(symbol) ? "true\n": "false\n");
    return 0;
}
```

Обработка СИМВОЛОВ

Демонстрация



Передача строк как параметров функций

- Как передать строку (она же — массив символов) в функцию в виде параметра?
- В C++ есть три способа передачи параметров:
 - По значению
 - По указателю
 - По ссылке (по адресу)



Пример (1/2)

```
#include <iostream>
using namespace std;
//определяем функции
void showText1 (char str[])
//функция принимает строку, как массив
{
    cout << str << endl;
}

void showText2 (char *str)
//указатель *str указывает на адрес первого символа строки
{
    cout << str << endl;
}

void showText3 (char (&str)[150])
// адрес строки из 150-ти символов
{
    cout << str << endl;
}
```

Пример (2/2)

```
int main()
{
    cout << " Передача строки при вызове функции: " << endl;
    showText1("~~~ ~~~ ~ berkut.homelinux.com ~ ~~~ ~~~");
    cout << endl;

    char str1[] =
        "str1 - как массив - void showText1 (char str[]);";
    showText1(str1);
    cout << endl;

    char str2[] =
        "str2 - используя указатель - void showText2 (char *str);";
    showText2(str2);
    cout << endl;

    char str3[150] =
        "str3 - используем адрес - void showText3 (char &str[]);";
    showText3(str3);
    cout << endl;

    return 0;
}
```

Демонстрация



Первый способ:

функция **void showText1 (char str[])**

принимает строку, как массив.

- При определении функции, в круглых скобках надо вписать тип массива (строки) `char`, имя строки и квадратные скобки `[]`.
- Обратите внимание, что размерность массива указывать не надо (количество символов в строке).
- С помощью символа `'\0'`, который автоматически добавляется в конец любой строки, функция сама найдет последний элемент массива и прекратит вывод элементов на экран.

Второй способ:

передача строки в функцию при помощи указателя, самый распространенный вариант.

- При определении функции, перед именем строки ставим оператор *

```
void showText2 (char *str)
```



Третий способ:

передача в функцию строки по ссылке.

В нашем примере это –

```
void showText3 (char (&str) [150])
```

- Следует отметить, что необходимо указывать точный размер строки.
- Без этого компилятор выдаст ошибку (проверим?).
- Еще важно взять в круглые скобки оператор & с именем строки — (&str).
- Строка `char str3[150]` содержит явно меньше символов. Остальным, не инициализированным, будет присвоено значение `'\0'`



Спасибо!
Вопросы?



Алгоритмизация и программирование

Программирование на C/C++
(ч.9 – строки)

Беркунский Е.Ю., кафедра ИУСТ, НУК
eugeny.berkunsky@gmail.com
<http://berkut.homelinux.com>

