**Лекция: Работа со строками.**

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.

|  |
| --- |
| // объявления символьной переменной  char symbol = 'a'; //  symbol – имя переменной типа  char    char a='A';  int n;  n=a;  cout << "a=" <<a<<" n="<<n<<endl; |
|  |

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

\0  — символ нуль-терминатора.

При объявлении строкового массива необходимо учитывать наличие в конце строки нуль-терминатора, и отводить дополнительный байт под него.

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

// вывод символов и их целочисленных значений

char c[256]; // объявление строки, как массива символов

int nc[256]; // объявление массива значений символов

for( int i=0; i<256; i++)

{ c[i]=i;

nc[i]=i;

cout<<'\t'<<c[i]<<"= "<<nc[i]<<'\n';

}

|  |  |
| --- | --- |
|  | // пример объявления строки  char string[10]; //  string – имя строковой переменной,10 – размер массива,  // то есть в данной строке может поместиться 9 символов ,  // последнее место отводится под нуль-терминатор. |

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

|  |  |
| --- | --- |
|  | char string[10] = "abcdefghf"; |

Если подсчитать кол-во символов в двойных кавычках после символа равно их окажется 9, а размер строки 10 символов,  последнее  место отводится под нуль–терминатор, 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  // нуль-терминатор. |

При объявлении строки не обязательно 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. |

Библиотека **cstring** определяет функци для обработки Cи-строк и массивов:

| Таблица 1 — Функции для работы со строками и символами | |
| --- | --- |
| **Функция** | **Пояснение** |
| [**strlen**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strlen/)**(**имя\_строки**)** | определяет длину указанной строки, без учёта нуль-символа |
| **Копирование строк** | |
| [**strcpy**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strcpy/)**(**s1,s2**)** | выполняет побайтное копирование символов из строки  s2 в строку s1 |
| [**strncpy**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strncpy/)**(**s1,s2, n**)** | выполняет побайтное копирование n символов из строки  s2 в строку s1. возвращает значения s1 |
| **Конкатенация строк** | |
| [**strcat(**s1,s2**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strcat/) | объединяет строку s2 со строкой s1. Результат сохраняется в s1 |
| [**strncat(**s1,s2,n**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strncat/) | объединяет n символов строки s2 со строкой s1. Результат сохраняется в s1 |
| **Сравнение строк** | |
| [**strcmp(**s1,s2**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strcmp/) | сравнивает строку s1 со строкой s2 и возвращает результат типа int: 0 –если строки эквивалентны, >0 – если s1<s2,  <0  — если s1>s2 С учётом регистра |
| [**strncmp(**s1,s2**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strncmp/) | сравнивает n символов строки s1 со строкой s2 и возвращает результат типа int: 0 –если строки эквивалентны, >0 – если s1<s2,  <0  — если s1>s2 С учётом регистра |
| **stricmp(**s1,s2**)** | сравнивает строку s1 со строкой s2 и возвращает результат типа **int**: 0 –если строки эквивалентны, >0 – если s1<s2,  <0  — если s1>s2 Без учёта регистра |
| **strnicmp(**s1,s2**)** | сравнивает n символов строки s1 со строкой s2 и возвращает результат типа int: 0 –если строки эквивалентны, >0 – если s1<s2,  <0 — если s1>s2 Без учёта регистра |
| **Обработка символов** | |
| [**isalnum(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-isalnum/) | возвращает значение true, если с является буквой или цифрой, и false в других случаях |
| [**isalpha(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-isalpha/) | возвращает значение true, если с является буквой,  и false в других случаях |
| [**isdigit(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-isdigit/) | возвращает значение true, если с является цифрой, и false в других случаях |
| [**islower(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-islower/) | возвращает значение true, если с является буквой нижнего регистра, и false в других случаях |
| [**isupper(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-isupper/) | возвращает значение true, если с является буквой верхнего регистра, и false в других случаях |
| [**isspace(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-isspace/) | возвращает значение true, если с является пробелом, и false в других случаях |
| [**toupper(**c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cctupe-ctupe-h/funkciya-toupper/) | если символ с, является символом нижнего регистра, то функция возвращает преобразованный символ с в верхнем регистре, иначе символ возвращается без изменений. |
| **Функции поиска** | |
| [**strchr(**s,c**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strchr/) | поиск первого вхождения символа св строке s**.**В случае удачного поиска возвращает указатель на место первого вхождения символа с**.**Если символ не найден, то возвращается ноль. |
| [**strcspn(**s1,s2**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strcspn/) | определяет длину начального сегмента строки s1, содержащего те символы, которые не входят в строку s2 |
| [**strspn(**s1,s2**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstring-string-h/funkciya-strspn/) | 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 |
| **Функции преобразования** | |
| [**atof(**s1**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstdlib-stdlib-h/funkciya-atof/) | преобразует строку s1 в тип double |
| [**atoi(**s1**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstdlib-stdlib-h/funkciya-atoi/) | преобразует строку s1 в тип int |
| [**atol(**s1**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstdlib-stdlib-h/funkciya-atol/) | преобразует строку s1 в тип long int |
| **Функции стандартной библиотеки ввода/вывода <stdio>** | |
| [**getchar(**с**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstdio-stdio-h/funkciya-getchar/) | считывает символ ссо стандартного потока ввода, возвращает символ в формате int |
| [**gets(**s**)**](http://cppstudio.com/spravochnik/standartnye-zagolovochnye-fajly-iz-si-v-s/zagolovochnyj-fajl-cstdio-stdio-h/funkciya-gets/) | считывает поток символов со стандартного устройства ввода в строку sдо тех пор, пока не будет нажата клавиша ENTER |

**// Примеры использования функций для работы со строками и символами.**

#include <iostream>

#include <cstring>

#include <ctype.h>

using namespace std;

int main()

{

char string[]="i am happy";

char instring[50];

char instring1[50];

int c;

c=strlen(string); //длина строки

cout<<c<<endl;

cout<<"\*\*\*\*\*"<<endl;

strcpy(instring,string); //копирование в 1 строку

cout<<string<<endl<<instring<<endl;

cout<<"\*\*\*\*\*"<<endl;

strncpy(instring1,instring,7); //копир в 1 строку 7 симв

cout<<instring<<endl<<instring1<<endl;

cout<<"\*\*\*\*\*"<<endl;

strcat(instring,string); //объединение строк в 1 строке

cout<<instring<<endl;

cout<<"\*\*\*\*\*"<<endl;

strncat(instring,string,3); //объединить 1 строку с з символами 2 строки

cout<<instring<<endl;

for(int i=c-1;i>=0;i--) // вывод символов строки в обратном порядке

{

cout<<string[i];

}

cout<<endl;

cout<<"\*\*\*\*\*"<<endl;

for(int i=0;i<c;i++)

{

if ( islower(string[i]) ) // если текущий символ строчная буква,

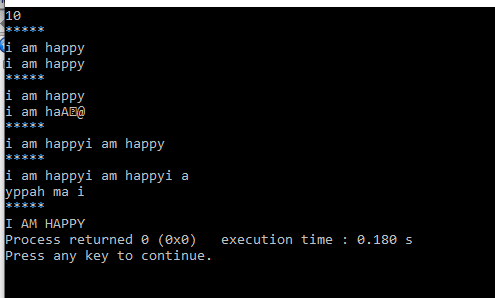
string[i] = toupper(string[i]); // перевод его в верхний регистр

cout<<string[i];

}

return 0;

}



**Функция cin.getline() и get().**

Функция **getline()** предназначена для ввода данных из потока, например, для ввода данных из консольного окна. Если формально описывать ее функционал, то она извлекает данные из входного потока до строкового разделителя, который не записывается в получившийся массив данных.

В итоге, получается извлечение одной [строки](https://proginfo.ru/strings/) и записывание ее в переменную. Сама конструкция getline  выглядит так:

**cin.getline(string, streamsize, separator);**

где string – переменная типа char, в которую запишется строка, streamsize - максимально количество символов, которое может быть записано в строку, и separator – строковый разделитель, показывающий на конец строки. Последний параметр функции можно опустить, тогда будет задан сепаратор по умолчанию - '\n'.

Функция **get()** похожа по своим действиям на **getline()** . Ей требуются такие же параметры и интерпретируются они так же. Она считывает данные до конца строки. Однако, вместо того, чтобы считать и удалить символ новой строки, она оставляет его во входной очереди. Вызов функции **cin.get()** без параметров позволяет считать только один символ, даже если это символ новой строки. Этим можно воспользоваться, чтобы избавиться от символа новой строки и приготовиться к приему следующей строки ввода.

// программа проверки введенного пользователем пароля

#include <iostream>

#include <cstring>

#include <iomanip>

using namespace std;

int main()

{

char str1[6];

cout<<"enter password: "<<endl;

cin.getline(str1,6);

char password[6] ="aassd";

while(strcmp(str1,password))

{

cout<<"Incorrect password!" <<endl;

cout << "enter password: "<<endl;

cin.getline(str1,6);

}

cout << " Access is allowed." << endl;

return 0; }

// пример использования cin.getline

#include <iostream>

#include <cstring>

using namespace std;

int main()

{

char name[10];

char surname[15];

cout<<" name:\n";

cin.getline(name,10);

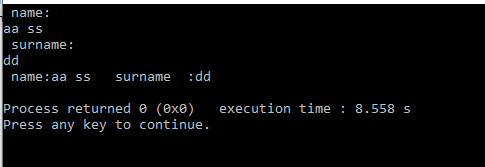
cout<<" surname:\n";

cin.getline(surname,15);

cout<<" name:"<<name<<" surname :"<<surname<<'\n';

return 0;

}



(1) // пример использования cin.get

#include <iostream>

#include <cstring>

using namespace std;

int main()

{

char name[10];

char surname[15];

cout<<" name:\n";

cin.get(name,10);

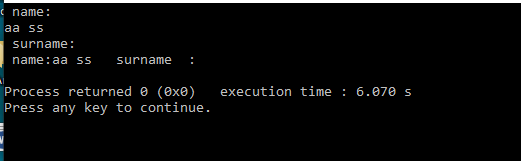
cout<<" surname:\n";

cin.get(surname,15);

cout<<" name:"<<name<<" surname :"<<surname<<'\n';

return 0;

}



(2) // пример использования cin.get()

#include <iostream>

#include <cstring>

using namespace std;

int main()

{

char name[10];

char surname[15];

cout<<" name:\n";

cin.get(name,10);

cout<<" surname:\n";

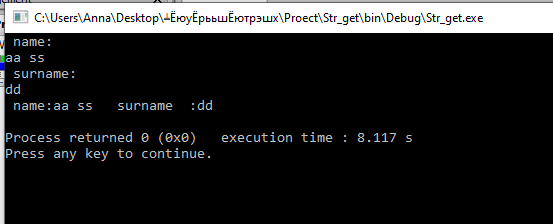
cin.get();

cin.get(surname,15);

cout<<" name:"<<name<<" surname :"<<surname<<'\n';

return 0;

}



// пример использования cin

#include <iostream>

#include <cstring>

using namespace std;

int main()

{

char name[10];

char surname[15];

cout<<" name:\n";

cin>>name;

cout<<" surname:\n";

cin>>surname;

cout<<" name:"<<name<<" surname :"<<surname<<'\n';

return 0;

}

