Символы и строки

Лектор: Артамонов Юрий Николаевич

Университет "Дубна"
филиал Котельники

Содержание

1. Понятие строки и символа
2. Описание библиотечных функций

Понятие строки и символа

Символы (символьные константы) - это величина типа int, представляемая в виде символа, заключенного в одинарные кавычки (апострофы). Как Вы знаете, с такими символьными константами программист может поступать на свое усмотрение, считая их в необходимый момент числами, или символами.

```
#include <stdio.h>
int main(){
   printf("%d\n",'z');
   printf("%c\n",122);
   return 0;}
```

Строка - это последовательность символов, с которой обращаются как с одним элементом. В С строки (строки-константы) заключаются в двойные кавычи.

```
#include <stdio.h>
int main(){
   printf("%s\n","Hello");
   char a[]={'H','e','l','l','o','\0'}; char *b = "Hello";
   printf("%s\n",a); printf("%s\n",b);
   return 0;}
```

Строка в С является массивом символов, который заканчивается *нулевым символом*. Доступ к строке осуществляется через указатель, ссылающийся на первый символ строки. Таким образом, в С можно сказать, что строка - это указатель на первый символ строки, т.е. массив символов.

Понятие строки и символа

Массиву можно присвоить строку, используя scanf. Например,

```
char word[10];
scanf("%s",word);
printf("%s\n",word);
int i;
for (i=0;i<9;i++)printf("%d - %c\n",word[i],word[i]);
```

Обратите внимание, что в scanf нет необходимости использовать &, поскольку word является указателем (адресом) на начало области памяти. Функция scanf будет читать символы до тех пор, пока не встретит символ пробела, новой строки или признак конца файла. В нашем случае длина строки не должна превышать 9 символов, чтобы оставалось место для ограничивающего строку символа нуль (NULL). Если будет введено более 9 символов, то строка может затереть область памяти с нужными данными.

Библиотека обработки символов

Таблица функций обработки символов

Для использования функций следует подключить библиотеку ctype.h.

Прототип	Описание функций
int isdigit(int c)	Возвращает значение true, если с является цифрой, и 0 (false) в других случаях.
int isalpha(int c)	Возвращает значение true, если с является буквой, и 0 (false) в других случаях.
int isalnum(int c)	Возвращает значение true, если с является буквой или цифрой, и 0 (false) в других случаях.
int islower(int c)	Возвращает значение true, если с является буквой нижнего регистра, и 0 (false) в других случаях.
int isupper(int c)	Возвращает значение true, если с является буквой верхнего регистра, и 0 (false) в других случаях.
int tolower(int c)	Если с является буквой верхного регистра, то возвращает букву нижнего, иначе возвращает букву без изменений.
int toupper(int c)	Если с является буквой нижнего регистра, то возвращает букву верхнего, иначе возвращает букву без изменений.
int isspace(int c)	Возвращает значение true, если с является пробельным символом, символом новая страница (\f), новая строка (\n), возврат каретки (\r), горизонтальная табуляция (\t), вертикальная табуляция(\v), в других случаях 0.
int iscntrl(int c)	Возвращает значение true, если с является управляющим символом, и 0 (false) в других случаях.
int isprint(int c)	Возвращает значение true, если с является отображаемым символом, и 0 (false) в других случаях.
int isgraph(int c)	Возвращает значение true, если с является отображаемым символом, исключая символ пробела, и 0 (false) в других случаях.

Задание 11.1

Введите строку с клавиатуры и преобразуйте все символы нижнего регистра в верхний, верхнего регистра в нижний.

Задание 11.2

Введите строку с клавиатуры и замените все служебные символы на символ пробела.

Задание 11.3

Введите строку с клавиатуры и оставьте в ней только числовую информацию.

Функции преобразования строк

Данные функции преобразуют строки цифр в целые значения и значения с плавающей точкой. Для их использования целесообразно подключить заголовочный файл stdlib.h. Обратите внимание, что const объявляет, что значение аргумента не будет изменяться.

Прототип	Описание функций
double atof(const char *nPtr)	Преобразует строку nPtr в тип double.
int atoi(const char *nPtr)	Преобразует строку nPtr в тип int.
long atol(const char *nPtr)	Преобразует строку nPtr в тип long int.
double strtod(const char *nPtr, char **endPtr)	Преобразует строку nPtr в тип double, сохраняя в endPtr остаток строки.
long strtol(const char *nPtr, char **endPtr, int base)	Преобразует строку nPtr в тип long, сохраняя в endPtr остаток строки.
unsigned long strtoul(const char *nPtr, char **endPtr, int base)	Преобразует строку nPtr в тип unsigned long, сохраняя в endPtr остаток строки.

Примеры использования функций atof, atoi, atol

Функции atof, atoi, atol преобразуют аргумент-строку в число соответствующего типа. Если аргумент строка начинается не цифровым символом, то поведение функций не определено.

```
#include <stdio.h>
#include<stdlib.h>
int main(){
  char *s1 = "123.5"; int n = atoi(s1); double h = atof(s1); long
  int m = atol(s1);
  char *s2 = "123.5 is number";
  printf("%d %f %ld\n", n, h, m);
  n = atoi(s2); h = atof(s2); m = atol(s2);
  printf("%d %f %ld\n", n, h, m);
  char *s3 = "First prime is 2, it's even";
  n = atoi(s3); h = atof(s3); m = atol(s3);
  printf("%d %f %ld\n", n, h, m);
  return 0;
}
```

Примеры использования функций strtod, strtol, strtoul

Функции strtod, strtol, strtoul преобразуют аргумент-строку в число соответствующего типа. При этом из строки выделяется число, а на остаток строки указывает endPtr. Однако это работает только для строк, которые начинаются с числа, иначе действие неопределено. У функций strtol, strtoul есть третий аргумент - это основание системы счисления. 0 - это указание на то, что число может быть записано в двоичной, восьмиричной, десятичной или шестнадцатиричной системе счисления, также можно указывать любое число из диапазона от 2 до 36, которое принудительно задает систему счисления.

```
#include <stdio.h>
#include<stdlib.h>
int main(){
  char *s1 = "110165 is number", *remain;
  double d = strtod(s1, &remain);
  printf("В строке %s выделено число %f и остаток %s\n",s1, d,
  remain);
  long int n = strtol(s1, &remain,2);
  printf("В строке %s выделено число %ld и остаток %s\n",s1, n,
  remain);
  unsigned long m = strtoul(s1, &remain,0);
  printf("В строке %s выделено число %lu и остаток %s\n",s1, m,
  remain);
  return 0;}
```

Задание 11.4

Выделите из строки символов все целые числа и сохраните их в динамический массив.

Задание 11.5

Выделите из текстового файла все целые числа и сохраните их в динамический массив. Преобразуйте элементы массива в строку и сохраните их в файл.

Функции библиотеки stdio для операций и символами и строковыми данными

Прототип	Описание функций
int getchar(void)	Вводит символ со станд. устройства ввода и возвращает его в формате целого.
char * gets (char * s)	Вводит символ со станд. устройства ввода в массив s до тех пор, пока не встретит символ новой строки или конца файла. В конце добавляется NULL.
int putchar(int c)	Печать символа, хранящегося в c.
int puts (const char * s)	Печать строки s с последующим символом новая строка.
int sprintf (char * s, спецификаторы, переменные)	Эквивалент printf за исключение того, что результат вывода записывается в массив s и не отображается на экране.
int sscanf (char * s, спецификаторы, переменные)	Эквивалент scanf за исключение того, что результат ввод осуществляется из массива s и не с клавиатуры.

Функции операций над строками из библиотеки string.h

Функции копирования и объединения строк

Прототип	Описание функций
char * strcpy (char * s1, const char * s2)	Копирует строку s2 в массив s1. Возвращает s1.
char * strncpy (char * s1, const char * s2, size_t n)	Копирует не более n символов строки s2 в массив s1. Возвращает s1.
char * strcat (char * s1, const char * s2)	Объединяет строку s2 со строкой s1. Первый символ s2 переписывает символ NULL строки s1. Возвращает s1.
char * strncat (char * s1, const char * s2, size_t n)	Объединяет не более n символов строки s2 со строкой s1. Первый символ s2 переписывает символ NULL строки s1. Возвращает s1.

Функции операций над строками из библиотеки string.h

Функции сравнения строк

Прототип	Описание функций
int strcmp (const char * s1, const char * s2)	Сравнивает строку s1 со строкой s2. Возвращает 0, меньше 0, больше 0, если s1 соответственно равна, меньше, больше s2.
int strncmp (const char * s1, const char * s2, size_t n)	Сравнивает до n символов строки s1 со строкой s2. Возвращает 0, меньше 0, больше 0, если s1 соответственно равна, меньше, больше s2.

Функции операций над строками из библиотеки string.h

Функции поиска

Прототип	Описание функций
char * strchr (const char * s, int c)	Находит позицию первого вхождения символа c в строку s. Если c найден, функция возвращает указатель на c в строке s, в противном случае NULL.
size_t strcspn (const char * s1, const char * s2)	Определяет и возвращает длину начального сегмента строки s1, содержащего только те символы, которые не входят в строку s2.
size_t strspn (const char * s1, const char * s2)	Определяет и возвращает длину начального сегмента строки s1, содержащего только те символы, которые входят в строку s2.
char * strprbk (const char * s1, const char * s2)	Находит в строке s1 позицию первого вхождения любого из символов строки s2. Если символ из s2 найден, возвращается указатель на этот символ в s1, иначе NULL.
char * strrchr (const char * s, int c)	Находит позицию последнего вхождения символа c в строку s. Если c найден, функция возвращает указатель на c в строке s, в противном случае NULL.
char * strstr (const char * s1, const char * s2)	Находит позицию первого вхождения строки s2 в строку s1. Если s2 найдена, функция возвращает указатель на s2 в строке s1, в противном случае NULL.
char * strtok (char * s1, const char * s2)	Последовательный вызов функции выполняет разбиение строки s1 на лексемы, разделенные символами, содержащимися в строке s2. В первом вызове передается s1, потом NULL.

Функции операций над строками из библиотеки string.h

Другие функции

Прототип	Описание функций
int strlen(const char * s)	Определяет длину строки s.