# 转载：https://www.cnblogs.com/xiangzi888/archive/2012/04/16/2451947.html

平时对字符串的操作的是很多的，了解下常用的字符串函数会使 c 编程变得很快捷！这里适当整理一下，方便以后参考。使用时，会用到大量指针的操作，注意加头文件：

#include <string.h>

一、str 系列

1.strtok

extern char \*strtok( char \*s, const char \*delim );

功能：分解字符串为一组标记串。s为要分解的字符串，delim为分隔符字符串。

说明：strtok()用来将字符串分割成一个个片段。当strtok()在参数s的字符串中发现到参数delim的分割字符时则会将该字符改为 \0 字符。在第一次调用时，strtok()必需给予参数s字符串，往后的调用则将参数s设置成NULL。每次调用成功则返回被分割出片段的指针。当没有被分割的串时则返回NULL。所有delim中包含的字符都会被滤掉，并将被滤掉的地方设为一处分割的节点。

举例：

[复制代码](javascript:void(0);)

/\* strtok example \*/

#include <stdio.h>

#include <string.h>

int main (void)

{

char str[] = "- This, a sample string.";

char \*pch;

printf("Splitting string \"%s\" into tokens:\n", str);

pch = strtok(str," ,.-");

while (pch != NULL)

{

printf("%s\n", pch);

pch = strtok(NULL, " ,.-");

}

printf("at the end: %s", str);

return 0;

}

[复制代码](javascript:void(0);)

Splitting string "- This, a sample string." into tokens:

This

a

sample

string

the end: - This

注：strtok函数会破坏被分解字符串的完整，调用前和调用后的s已经不一样了。另外貌似制表符 \t 无法充当分割字符。

2.strstr

char \* strstr( const char \* str1, const char \* str2 );

功能：从字符串 str1 中寻找 str2 第一次出现的位置（不比较结束符NULL)，如果没找到则返回NULL。

举例：

[复制代码](javascript:void(0);)

/\* strstr example \*/

#include <stdio.h>

#include <string.h>

int main ()

{

char str[] = "This is a s**i**mple string";

char \*pch;

pch = strstr(str, "s**i**mple");

strncpy(pch, "s**a**mple", 6);

puts(pch);

puts(str);

return 0;

}

[复制代码](javascript:void(0);)

sample string

This is a sample string

3.strchr

char \* strchr ( const char \*str, int ch );

功能：查找字符串 str 中首次出现字符 ch 的位置  
说明：返回首次出现 ch 的位置的指针，如果 str 中不存在 ch 则返回NULL。

举例：

[复制代码](javascript:void(0);)

/\* strchr example \*/

#include <stdio.h>

#include <string.h>

int main ()

{

char str[] = "This is a simple string";

char \*pch;

printf("Looking for the 's' character in \"%s\"...\n", str);

pch = strchr(str, 's');

while (pch != NULL){

printf("found at %d th\n", pch - str + 1);

pch = strchr(pch + 1, 's');

}

return 0;

}

[复制代码](javascript:void(0);)

Looking for the 's' character in "This is a simple string"...

found at 4 th

found at 7 th

found at 11 th

found at 18 th

4.strcpy

char \* strcpy( char \* dest, const char \* src );

功能：把 src 所指由NULL结束的字符串复制到 dest 所指的数组中。  
说明：src 和 dest 所指内存区域不可以重叠且 dest 必须有足够的空间来容纳 src 的字符串。返回指向 dest 结尾处字符(NULL)的指针。

类似的：

strncpy

char \* strncpy( char \* dest, const char \* src, size\_t num );

stpcpy

非库函数，用法跟 strcpy 完全一样

5.strcat

char \* strcat ( char \* dest, const char \* src );

功能：把 src 所指字符串添加到 dest 结尾处(覆盖dest结尾处的'\0')并添加'\0'。  
说明：src 和 dest 所指内存区域不可以重叠且 dest 必须有足够的空间来容纳 src 的字符串。  
返回指向 dest 的指针。

类似的 strncat

char \* strncat ( char \* dest, const char \* src, size\_t num );

6.strcmp

int strcmp ( const char \* str1, const char \* str2 );

功能：比较字符串 str1 和 str2。  
说明：  
当s1<s2时，返回值<0  
当s1=s2时，返回值=0   
当s1>s2时，返回值>0

类似的：

strncmp

int strncmp ( const char \* str1, const char \* str2, size\_t num );

strcasecmp

extern int strcasecmp(const char \*str1, const char \*str2);

strncasecmp

extern int strncasecmp(const char \*str1, const char \*str2, size\_t num);

7.strcspn

size\_t strcspn ( const char \* str1, const char \* str2 );

功能：在字符串 s1 中搜寻 s2 中所出现的任一个字符。  
说明：返回出现 s2 中字符时已读入的字符数，亦即在 s1 中出现而 s2 中没有出现的子串的长度。

[复制代码](javascript:void(0);)

/\* strcspn example \*/

#include <stdio.h>

#include <string.h>

int main ()

{

char str[] = "fcba73";

char keys[] = "1234567890";

int i = strcspn (str, keys);

printf ("Already read %d characters\n", i);

printf ("The first number in both str and keys is %d th\n", i + 1);

return 0;

}

[复制代码](javascript:void(0);)

类似的 strspn （Returns the length of the initial portion of *str1* which consists only of characters that are part of *str2*.）

size\_t strspn ( const char \* str1, const char \* str2 );

[复制代码](javascript:void(0);)

#include <stdio.h>

#include <string.h>

int main ()

{

char str[] = "1589fcba73";

char keys[] = "1234567890";

int i = strspn (str, keys);

printf ("the beginning %d characters are all in keys \n", i);

return 0;

}

[复制代码](javascript:void(0);)

the beginning 4 characters are all in keys

8.strlen

size\_t strlen ( const char \* str );

功能：计算字符串 str 的长度  
说明：返回 str 的长度，不包括结束符NULL。（注意与 sizeof 的区别）

类似的 strnlen

size\_t strnlen(const char \*str, size\_t maxlen);

9.strdup

extern char \*strdup( char \*str );

功能：复制字符串 str  
说明：返回指向被复制的字符串的指针，所需空间由malloc()分配且可以由free()释放。

[复制代码](javascript:void(0);)

#include <stdio.h>

#include <string.h>

int main ()

{

char \*str = "1234567890";

char \*p = strdup(str);

printf("the duplicated string is: %s\n", p);

return 0;

}

[复制代码](javascript:void(0);)

二、mem 系列

1.memset

void \* memset ( void \* ptr, int value, size\_t num );

功能：把 ptr 所指内存区域的前 num 个字节设置成字符 value。  
说明：返回指向 ptr 的指针。可用于变量初始化等操作

举例：

[复制代码](javascript:void(0);)

#include <stdio.h>

#include <string.h>

int main ()

{

char str[] = "almost erery programer should know memset!";

memset(str, '-', sizeof(str));

printf("the str is: %s now\n", str);

return 0;

}

[复制代码](javascript:void(0);)

2.memmove

void \* memmove ( void \* dest, const void \* src, size\_t num );

功能：由 src 所指内存区域复制 num 个字节到 dest 所指内存区域。  
说明：src 和 dest 所指内存区域**可以重叠**，但复制后 src 内容会被更改。函数返回指向dest的指针。

举例：

[复制代码](javascript:void(0);)

#include <stdio.h>

#include <string.h>

int main ()

{

char str[] = "memmove can be very useful......";

memmove(str + 20, str + 15, 11);

printf("the str is: %s\n", str);

return 0;

}

[复制代码](javascript:void(0);)

the str is: memmove can be very very useful.

3.memcpy

void \* memcpy ( void \* destination, const void \* source, size\_t num );

类似 strncpy。区别：拷贝指定大小的内存数据，而不管内容（不限于字符串）。

4.memcmp

int memcmp ( const void \* ptr1, const void \* ptr2, size\_t num );

类似 strncmp

5.memchr

void \* memchr ( const void \*buf, int ch, size\_t count);

功能：从 buf 所指内存区域的前 count 个字节查找字符 ch。  
说明：当第一次遇到字符 ch 时停止查找。如果成功，返回指向字符 ch 的指针；否则返回NULL。