目录

---

sizeof:

C / C language / Expressions

sizeof operator

Queries size of the object or type.

Used when actual size of the object must be known.

Syntax

sizeof(*type* **)**	(1)
sizeof *expression*	(2)

Both versions return a value of type `size_t`.

Explanation

1) Returns the size, in bytes, of the object representation of *type*
2) Returns the size, in bytes, of the object representation of the type of *expression*. No implicit conversions are applied to *expression*.

首先我要在此声明sizeof不是函数！不是函数！不是函数！而是一个操作符！（看到operator了吗？）

```
#include

int main()
{
    int a = 1;//sizeof不会读取数据，只会计算所占内存空间的大小（单位是字节）
    printf("%zd\n", sizeof(a));  //4
    printf("%zd\n", sizeof a );  //4  看吧，就算没有()只需要空格sizeof也能正确被执
行。
    printf("%zd\n", sizeof 1);   //4  这里体现了sizeof就不是函数
    return 0;
}
```

这里要强调的是如果sizeof操作对象是一个表达式，则被操作的表达式里的计算不会被执行。

```
int main()
{
    int a = 3;
    int b = 5;
    printf("%zd\n", sizeof(a += b));  //4
    printf("%d\n",a);//因为a+=b不会计算那么a的值还是3。
    return 0;
}
```

strlen：

C / Strings library / Null-terminated byte strings

strlen, strnlen_s

Defined in header <string.h>

```
size_t strlen( const char *str );                    (1)
size_t strnlen_s( const char *str, size_t strsz );   (2)  (since C11)
```

1) Returns the length of the given null-terminated byte string, that is, the number of characters in a character array whose first element is pointed to by up to and not including the first null character.str

The behavior is undefined if is not a pointer to a null-terminated byte string.str

2) Same as (1), except that the function returns zero if is a null pointer and returns if the null character was not found in the first bytes of .strstrszstrszstr

The behavior is undefined if both points to a character array which lacks the null character and the size of that character array < ; in other words, an erroneous value of does not expose the impending buffer overflow. strstrszstrsz

As with all bounds-checked functions, only guaranteed to be available if strnlen_s __STDC_LIB_EXT1__ is defined by the implementation and if the user defines __STDC_WANT_LIB_EXT1__ to the integer constant 1 before including <string.h>.

Parameters

str - pointer to the null-terminated byte string to be examined

strsz - maximum number of characters to examine

Return value

1) The length of the null-terminated byte string .str

2) The length of the null-terminated byte string on success, zero if is a null pointer, if the null character was not found.strstrstrsz

strlen是C语言中的库函数。原型是

```
size_t strlen(const char* str)
```

看的出来strlen是专门求字符串长度的，统计的是从此地址开始往后\0之前字符串的字符个数，那么它就有可能越界查找。如下所示：

```
int main()
{
    char a[] = "abc";
    char b[] = { 'a','b','c' };
    printf("%zd\n", strlen(a));//3       a在内存中是：  a b c \0 这样的strlen读取了\0前面的字符个数
    printf("%zd\n", strlen(b));//随机值   b在内存中是：  a b c 随机值…… 这样的，所以我们不知道后面什么地方会出现\0
    printf("%zd\n", sizeof(a));//4       a包括\0在内占了四个字节
    printf("%zd\n", sizeof(b));//3       b只包括了abc三个字符所以占三个字节
```

```
    return 0;
}
```

总结区别：

sizeof	strlen
1. sizeof是操作符 2. sizeof计算操作数所占内存的大小，单位是字节 3. 不关注内存中存放什么数据	1. strlen是库函数，使用需要包含头文件 `string.h` 2. srtlen是求字符串长度的，统计的是 `\0` 之前字符的隔个数 3. 关注内存中是否有 `\0` ，如果没有 `\0` ，就会持续往后找，可能会越界