<https://blog.csdn.net/zhouwei1221q/article/details/44890617>

以前一直以为sscanf只能用来简单的提取用空格隔起来的字符串，知道之前遇到了一些列字符串处理问题，详细研究起来这个函数尽然还是有很强大的有很多类似于正则表达式的功能。首先我们先来看函数定义：  
定义函数 int sscanf (const char \*str,const char \* format,........);         
 函数说明:

sscanf()会将参数str的字符串根据参数format字符串来转换并格式化数据。格式转换形式请参考scanf()。转换后的结果存于对应的参数内。  
         返回值 成功则返回参数数目，失败则返回-1，错误原因存于errno中。 返回0表示失败    否则，表示正确格式化数据的个数    例如：sscanf(str，"%d%d%s", &i,&i2, &s);    如果三个变成都读入成功会返回3。    如果只读入了第一个整数到i则会返回1。证明无法从str读入第二个整数。  
定义太抽象了，我们还是先来看一下常见的用法吧：  
（1）sscanf("zhoue3456 ", "%4s", str); //取指定长度的字符串         
          printf("str=%s\n", str);   //str="zhou";  
（2）sscanf("zhou456 hedf", "%[^ ]", str); //取到指定字符为止的字符串,取遇到空格为止字符串      
          printf("str=%s\n", str);  //str=zhou456;  
（3）sscanf("654321abcdedfABCDEF", "%[1-9a-z]", str); //取仅包含指定字符集的字符串  
          printf("str=%s\n", str);  //str=654321abcded，只取数字和小写字符  
（4）sscanf("BCDEF123456abcdedf", "%[^a-z]", str); //取到指定字符集为止的字符串         
           printf("str=%s\n", str);  //  str=BCDEF123456, 取遇到大写字母为止的字符串  
（5）int a,b,c;  
          sscanf("2015.04.05", "%d.%d.%d", &a,&b,&c); //取需要的字符串     
          printf("a=%d,b=%d,c=%d",a,b,c);  //  a=2015,b=4,c=5  
通过上面这几个例子相信大家对sscanf的用法会有一个直观的理解了,下面我们再来看一下更复杂一些的例子：  
（6）给定一个字符串“abcd&hello$why”,现在我想取出&与$之间的字符串怎么办呢  
        sscanf("abcd&hello$why", "%\*[^&]&%[^$]", str );<span style="white-space:pre"></span>printf("str=%s\n",str);  //str="hello"  
       其中%[]类似于一个正则表达式，如[a-z]表示读取所有a-z的字符，[^a-z]表示读取所有非小写字母的字符。那么在这里%\*[^&]表示先过滤掉abcd，然后以&隔开,后面还剩hello$why，然后将$之前非$的字符提取到str中。  
（7）给定一个字符串“what, time”，如果我想仅保留time,那该怎么办呢？(,后面有个空格)  
         sscanf(“what, time”, "%\*s%s", str );<span style="white-space:pre"></span>printf("str=%s\n",str);  //str="time"  
         其中%\*s表示第一个被匹配到的字符串what，被过滤掉了，如果没有空格，则结果为NULL。其实“what, time”被空格分割成了两个字符串"what,"和"time".  
然而有人会问，C语言中的scanf和sscanf有什么区别和联系呢？OK，sscanf和scanf确实很类似，两者都是用于输入。只是后者以屏幕stidin为输入源，而前者是以字符串为输入源，仅此而已。  
函数原型：int scanf( const char \*format [,argument]... );  
其中的format可以是一个或多个 {%[\*] [width] [{h | l | I64 | L}]type | ' ' | '/t' | '/n' | 非%符号}，注：{a|b|c}表示a,b,c中选一，[d],表示可以有d也可以没有d。  
width:宽度，一般可以忽略，用法如：　const char sourceStr[] = "hello, world";　char buf[10] = ;　sscanf(sourceStr, "%5s", buf); //%5s，只取5个字符  
cout << buf<< endl;　  
结果为:hello  
{h | l | I64 | L}:参数的size,通常h表示单字节size，I表示2字节 size,L表示4字节size(double例外),l64表示8字节size。　  
type :这就很多了，就是%s,%d之类。　  
特别的：　%\*[width] [{h | l | I64 | L}]type 表示满足该条件的被过滤掉，不会向目标参数中写入值。  
如：　const char sourceStr[] = "hello, world";　char buf[10] = ;　  
             sscanf(sourceStr, "%\*s%s", buf); //%\*s表示第一个匹配到的%s被过滤掉，即hello被过滤了  
         　cout << buf<< endl;　结果为:world　  
支持集合操作：　%[a-z] 表示匹配a到z中任意字符，贪婪性(尽可能多的匹配)  
                           　%[aB'] 匹配a、B、'中一员，贪婪性

                               %[^a] 匹配非a的任意字符，贪婪性

C++代码：参考链接：  http://kmplayer.iteye.com/blog/556293

**[cpp]** [view plain](https://blog.csdn.net/zhouwei1221q/article/details/44890617) [copy](https://blog.csdn.net/zhouwei1221q/article/details/44890617)

1. <span style="font-family: Arial, Helvetica, sans-serif;">1,sscanf():从一个字符串中读进与指定格式相符的数据.</span>

**[cpp]** [view plain](https://blog.csdn.net/zhouwei1221q/article/details/44890617) [copy](https://blog.csdn.net/zhouwei1221q/article/details/44890617)

1. 2,sscanf与scanf类似，都是用于输入的，只是后者以屏幕(stdin)为输入源，前者以固定字符串为输入源。
2. 3,关于正则表达式:
3. (1)%[..],当字符属于方括号里表达式表示的字符集时继续读取,否则停止.方括号里的和正则表达式差不多,^是"排除..."的意思
4. (2)%\*[..],直接跳过方括号里的字符集并继续读取

**[cpp]** [view plain](https://blog.csdn.net/zhouwei1221q/article/details/44890617) [copy](https://blog.csdn.net/zhouwei1221q/article/details/44890617)

1. <pre name="code" **class**="cpp">#include <iostream>
2. **using** **namespace** std;
4. **int** main()
5. {
6. **char** str[10];
7. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
8. cout<<str<<endl;
9. sscanf("123456","%s",str);//---------str的值为 "123456\0!!!"
10. //这个实验很简单，把源字符串"123456"拷贝到str的前6个字符，并且把str的第7个字符设为null字符，也就是\0
11. cout<<str<<endl;
13. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
14. sscanf("123456","%3s",str); //---------str的值为 "123\0!!!!!!"
15. //看到没有，正则表达式的百分号后面多了一个3，这告诉sscanf只拷贝3个字符给str，然后把第4个字符设为null字符。
16. cout<<str<<endl;
18. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
19. sscanf("aaaAAA","%[a-z]",str);// ---------str的值为 "aaa\0!!!!!!"
20. //从这个实验开始我们会使用正则表达式，括号里面的a-z就是一个正则表达式，它可以表示从a到z的任意字符，
21. //在继续讨论之前，我们先来看看百分号表示什么意思，%表示选择，%后面的是条件，比如实验1的"%s"，s是一个条件，表示任意字符，"%s"的意思是：只要输入的东西是一个字符，就把它拷贝给str。实验2的"%3s"又多了一个条件：只拷贝3个字符。实验3的“%[a-z]”的条件稍微严格一些，输入的东西不但是字符，还得是一个小写字母的字符，所以实验3只拷贝了小写字母"aaa"给str，别忘了加上null字符。
22. cout<<str<<endl;
24. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
25. sscanf("AAAaaaBBB","%[^a-z]",str);// ---------str的值为 "AAA\0!!!!!!"
26. //对于所有字符，只要不是小写字母，都满足"^a-z"正则表达式，符号^表示逻辑非。前3个字符都不是小写字符，所以将其拷贝给str，但最后3个字符也不是小写字母，为什么不拷贝给str呢？这是因为当碰到不满足条件的字符后，sscanf就会停止执行，不再扫描之后的字符。
27. cout<<str<<endl;
29. /\*
30. for (int i = 0; i < 10; i++) str[i] = '!';
31. sscanf("AAAaaaBBB","%[A-Z]%[a-z]",str);// ---------段错误
32. //这个实验的本意是：先把大写字母拷贝给str，然后把小写字母拷贝给str，但很不幸，程序运行的时候会发生段错误，因为当sscanf扫描到字符a时，违反了条件"%[A-Z]"，sscanf就停止执行，不再扫描之后的字符，所以第二个条件也就没有任何意义，这个实验说明：不能使用%号两次或两次以上
33. cout<<str<<endl;
34. \*/
36. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
37. sscanf("AAAaaaBBB","%\*[A-Z]%[a-z]",str); //---------str的值为 "aaa\0!!!!!!"
38. //这个实验出现了一个新的符号：%\*，与%相反，%\*表示过滤满足条件的字符，在这个实验中，%\*[A-Z]过滤了所有大写字母，然后再使用%[a-z]把之后的小写字母拷贝给str。如果只有%\*，没有%的话，sscanf不会拷贝任何字符到str，这时sscanf的作用仅仅是过滤字符串。
39. cout<<str<<endl;
41. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
42. sscanf("AAAaaaBBB","%[a-z]",str);// ---------str的值为 "!!!!!!!!!!"
43. //做完前面几个实验后，我们都知道sscanf拷贝完成后，还会在str的后面加上一个null字符，但如果没有一个字符满足条件，sscanf不会在str 的后面加null字符，str的值依然是10个惊叹号。这个实验也说明了，如果不使用%\*过滤掉前面不需要的字符，你永远别想取得中间的字符。
44. cout<<str<<endl;
46. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
47. sscanf("AAAaaaBC=","%\*[A-Z]%\*[a-z]%[^a-z=]",str); //---------str的值为 "BC\0!!!!!!!"
48. //这是一个综合实验，但这个实验的目的不是帮我们复习前面所学的知识，而是展示两个值得注意的地方：
49. //注意1：%只能使用一次，但%\*可以使用多次，比如在这个实验里面，先用%\*[A-Z]过滤大写字母，然后用%\*[a-z]过滤小写字母。
50. // 注意2：^后面可以带多个条件，且这些条件都受^的作用，比如^a-z=表示^a-z且^=(既不是小写字母，也不是等于号)。
51. cout<<str<<endl;
53. **for** (**int** i = 0; i < 10; i++) str[i] = '!';
54. **int** k;
55. sscanf("AAA123BBB456", "%\*[A-Z]%i", &k); //---------k的值为123
56. //首先，%\*[^0-9]过滤前面非数字的字符，然后用%i把数字字符转换成int型的整数，拷贝到变量k，注意参数必须使用k的地址。    cout<<str<<endl;
57. cout<<k<<endl;
58. **return** 0;
59. }

参考链接：

    http://blog.csdn.net/jackyvan/article/details/5349724

    http://kmplayer.iteye.com/blog/556293

版权声明：本文为博主原创文章，转载请注明出处。 https://blog.csdn.net/zhouwei1221q/article/details/44890617