**Scanf详解**

一、scanf()函数返回成功赋值的数据项数，出错时则返回EOF。

二、其控制串由三类字符构成：  
       1、格式化说明符；   
       2、空白符；   
       3、非空白符；

1、格式化说明符。

%p                 读入一个指针   
%u                 读入一个无符号十进制整数   
%n                 至此已读入值的等价字符数   
%[]                 扫描字符集合   
%%                 读%符号

附加格式说明字符表  
修饰符                           说明  
L/l 长度修饰符               输入"长"数据   
h 长度修饰符                 输入"短"数据   
W 整型常数                   指定输入数据所占宽度   
\* 星号                            空读一个数据   
hh,ll同上h,l但仅对C99有效。

2、 空白字符。

空白字符会使scanf()函数在读操作中略去输入中的一个或多个空白字符，

空白符可以是space,tab,newline等等，直到第一个非空白符出现为止。

3、 非空白字符。

一个非空白字符会使scanf()函数在读入时剔除掉与这个非空白字符相同的字符。

三、scanf()的格式控制串可以使用其它非空白字符，但在输入时必须输入这些字符。  
1：   
scanf("%d,%d",&a,&b);   
输入： 3,4 ↙（逗号与"%d,%d"中的逗号对应）

注意：如果%d,%d，在输入的时候数字需要紧跟“,”号。

           如果%d    , %d，在输入的时候数字后面可以有无数个空格。

2：

scanf("a=%d,b=%d",&a,&b);   
输入： a=3, b=4 ↙（"a=","b=",逗号与"%d,%d"中的"a=","b="及逗号对应）

注意：无论在a   = %d, b=%d有多少个空格，输入的时候只有上面的情况有效。

四、在用"%c"输入时，空格和“转义字符”均作为有效字符。  
例：   
scanf("%c%c%c",&c1,&c2,&c3);   
输入：a□b□c↙   
结果：a→c1，□→c2，b→c3 (其余被丢弃)

注意：%c %c的话，空格不看成有效字符。

scanf()函数接收输入数据时，遇以下情况结束一个数据的输入：

（不是结束该scanf函数，scanf函数仅在每一个数据域均有数据，并按回车后结束）。   
        ① 遇空格、“回车”、“跳格”键。   
        ② 遇宽度结束。 （%c%c%c）宽度为3.  
        ③ 遇非法输入。

五、scanf()函数不能正确接受有空格的字符串？如: I love you!

#include <stdio.h>   
int main()   
{   
    char str[80];   
      
    scanf("%s",str);   
    printf("%s",str);  
    return 0;   
}  
输入：I love you!   
输出：I

        scanf()扫描到"I"后面的空格就认为对str的赋值结束，并忽略后面的"love you!".这里要注意是"love you!"还在键盘缓冲区（关于这个问题，网上我所见的说法都是如此，但是，我经过调试发现，其实这时缓冲区字符串首尾指针已经相等了，也就是说缓冲区清空了，scanf()函数应该只是扫描stdin流，这个残存信息是在stdin中)。

我们改动一下上面的程序来验证一下：

#include <stdio.h>   
int main()   
{   
    char str[80];   
    char str1[80];   
    char str2[80];   
      
    scanf("%s",str);/\*此处输入:I love you! \*/   
    printf("%s",str);   
    sleep(5);/\*这里等待5秒,告诉你程序运行到什么地方\*/   
    scanf("%s",str1);/\*这两句无需你再输入,是对键盘盘缓冲区再扫描   \*/   
    scanf("%s",str2);/\*这两句无需你再输入,是对键盘盘缓冲区再扫描    \*/   
    printf("\n%s",str1);   
    printf("\n%s",str2);   
    return 0;   
}  
输入：I love you!   
输出：I   
      love   
      you!

       好了，原因知道了，那么scanf()函数能不能完成这个任务？回答是：能！别忘了scanf()函数还有一个 %[] 格式控制符（如果对%[]不了解的请查看本文的上篇）,请看下面的程序：  
#include "stdio.h"   
int main()   
{   
    char string[50];   
      
     /\*scanf("%s",string);不能接收空格符\*/   
     scanf("%[^\n]",string);   
     printf("%s\n",string);   
     return 0;   
}

六、键盘缓冲区残余信息问题

#include <stdio.h>   
int main()   
{   
    int a;   
    char c;  
    do   
    {   
        scanf("%d",&a);   
        scanf("%c",&c);   
        printf("a=%d     c=%c/n",a,c);   
        /\*printf("c=%d/n",c);\*/   
    }while(c != 'N');   
}

        每击打一下"Enter"键，向键盘缓冲区发去一个“回车”(/r),一个“换行"(/n),在这里/r被scanf()函数处理掉了（姑且这么认为吧^\_^），而/n被scanf()函数“错误”地赋给了c.

解决方法：

        可以在两个scanf()函数之后加个fflush(stdin);，还有加getch(); getchar();也可以。但是加fflush(stdin);不管什么情况都可行。

函数名: fflush   
功 能: 清除一个流   
用 法: int fflush(FILE \*stream);  
#include <stdio.h>   
int main()   
{   
    int a;   
    char c;  
    do   
    {   
        scanf("%d",&a);   
        fflush(stdin);   
        scanf("%c",&c);   
        fflush(stdin);   
        printf("a=%d     c=%c/n",a,c);  
    }while(c!='N');   
}

这里再给一个用“空格符”来处理缓冲区残余信息的示例：

运行出错的程序：  
#include <stdio.h>   
int main()   
{   
    int i;   
    char j;   
    for(i = 0;i < 10;i++)   
    {   
        scanf("%c",&j);/\*这里%前没有空格\*/   
    }   
}  
使用了空格控制符后：输入10个有效字符（把空格作为非法字符）才会退出。  
#include <stdio.h>   
int main()   
{   
    int i;   
    char j;   
    for(i = 0;i < 10;i++)   
    {   
        scanf(" %c",&j);/\*注意这里%前有个空格\*/   
    }   
}

七：如何处理scanf()函数误输入造成程序死锁或出错？

#include <stdio.h>   
int main()   
{   
int a,b,c; /\*计算a+b\*/  
scanf("%d,%d",&a,&b);   
c=a+b;   
printf("%d+%d=%d",a,b,c);   
}

scanf()函数执行成功时的返回值是成功读取的变量数,也就是说，你这个scanf()函数有几个变量，如果scanf()函数全部正常读取，它就返回几。但这里还要注意另一个问题，如果输入了非法数据，键盘缓冲区就可能还个有残余信息问题。

正确的例程：  
#include <stdio.h>   
int main()   
{   
    int a,b,c; /\*计算a+b\*/  
    while(scanf("%d,%d",&a,&b) != 2)

           fflush(stdin);  
    c=a+b;  
    printf("%d+%d=%d",a,b,c);  
}

八、scanf函数探讨

1、空白符问题

#include <stdio.h>   
main()   
{   
int a;   
printf( "input the data/n ");   
scanf( "%d/n ",&a);//这里多了一个回车符/n   
printf( "%d ",a);   
return 0;   
}

结果要输入两个数程序才结束，而不是预期的一个。why？   
原因：用空白符结尾时，scanf会跳过空白符去读下一个字符，所以你必须再输入一个数。这里的空白符包括 空格，制表符，换行符，回车符和换页符。所以如果你用scanf( "%d ",&a)也会出现同样的问题。

2、缓冲区问题

#include <stdio.h>   
main()   
{   
int n = 5;   
char c[n];   
for(int i = 0; i < n; i++)   
c[i] = scanf( "%c ",&c[i]);   
printf(c);   
return 0;   
}   
如果输入:   
a   
b   
c   
那么循环就会“提前”结束了.

原因：输入a和第一个回车后，a和这个回车符都留在缓冲区中。第一个scanf读取了a，但是输入缓冲区里面 还留有一个/n，第二个scanf读取这个/n。然后输入b和第二个回车，同样的，第三个scanf读取了b，第四个 scanf读取了第二个回车符。第五个读取了c。所以五个scanf都执行了，并没有提前结束。只不过有的scanf 读取到了回车符而已。   
解决方法：把程序改成这样就可以了：   
for( i = 0; i < n; i++){   
scanf( "%c ",&c[i]);   
fflush(stdin);//刷新缓冲区   
}   
或者不用scanf，而用gets（）函数，如：   
#include <stdio.h>   
main()   
{   
char c[5];   
gets(c);   
printf(c);   
return 0;   
}   
但要注意：这个函数自动把你最后敲的回车转换为字符 '/0 '。如果你的输入超过了数组的大小，那么就会产 生错误。

3.scanf()函数的参数输入类型不匹配问题

#include <stdio.h>   
main()   
{   
int a=123;   
char c= 't ';   
printf( "input/n ");   
scanf( "%d%c ",&a,&c);   
scanf( "%d%c ",&a,&c);   
scanf( "%d%c ",&a,&c);   
printf( "%d/n%c/n ",a,c);   
return 0;   
}

当输入a 回车后，会直接跳过下面２个scanf语句，直接输出为   
123   
t

原因：对于scanf( "%d%c ",&a,&c)，scanf语句执行时，首先试图从缓冲区中读入一个%d类型的数据，如果和第一个参数匹配，则继续从缓冲区中读取数据和第二个参数进行匹配，依次进行下去，直到匹配完所有的参 数；如果其中有一个参数不匹配，那就从这个地方跳出，忽略这个scanf后面所有的参数，而去执行下一条语句。

可以用下面的程序验证一下：   
#include <stdio.h>   
int main()   
{   
int a=123,b=1;   
char c= 't ';   
scanf( "%d%d ",&a,&b);   
scanf( "%c ",&c);   
printf( "%d/n%d/n%c/n ",a,b,c);   
return 0;   
}输入：2 回车a 回车   
结果是：   
2   
1   
a

解决方法：

#include <stdio.h>   
main()   
{   
int a=123,b;   
while(scanf( "%d%d ",&a,&b)!=2)   
fflush(stdin);   
printf( "%d/n%d/n ",a,b);   
return 0;   
}

补充：scanf中一种很少见但很有用的转换字符：[...]和[ ^...]。   
#include <stdio.h>   
main()   
{   
char strings[100];   
scanf( "%[1234567890] ",strings);   
printf( "%s ",strings);   
return 0;   
}   
运行，输入：1234werew后，结果是：1234。   
通过运行可以发现它的作用是：如果输入的字符属于方括号内字符串中某个字符，那么就提取该字符；如果 一经发现不属于就结束提取。该方法会自动加上一个字符串结束符到已经提取的字符后面。

scanf( "%[^1234567890] ",strings); 它的作用是：如果一经发现输入的字符属于方括号内字符串中某个字符 ，那么就结束提取；如果不属于就提取该字符。该方法会自动加上一个字符串结束符到已经提取的字符后面 。

注意：方括号两边不能空格，如：scanf( "%[ 1234567890 ] ",strings); scanf( "%[ ^1234567890  ] ",strings); 不让空格也会算在里面的。

用这种方法还可以解决scanf的输入中不能有空格的问题。只要用 scanf( "%[^/n] ",strings); 就可以了。很神奇吧。

百分号(%)与格式符之间的星号(\*)表示读指定类型的数据但不保存。

因此scanf( "%d %\*c %d ", &x, &y );   
对 10/20 的读入操作中，10 放入变量 x，20 放入 y。   
  
格式命令可以说明最大域宽。 在百分号(%)与格式码之间的整数用于限制从对应域读入的最大字符数。   
例如，希望向 address 读入不多于 20 个字符时，可以书写成如下形式：   
scanf( "%20s ", address );   
  
如果输入流的内容多于 20 个字符，则下次 scanf() 从此次停止处开始读入。 若达到最大域宽前已遇 到空白符，则对该域的读立即停止；此时，scanf() 跳到下一个域。   
虽然空格、制表符和新行符都用做域分割符号，但读单字符操作中却按一般字符处理。例如，对输入流 "x y " 调用：   
scanf( "%c%c%c ", &a, &b, &c );   
返回后，x 在变量 a 中，空格在变量 b 中，y 在变量 c 中。   
注意，控制串中的其它字符，包括空格、制表符和新行符，都用于从输入流中匹配并放弃字符，被匹配 的字符都放弃。例如，给定输入流 "10t20 "，调用：   
scanf( "%dt%d ", &x, &y );   
将把 10 和 20 分别放到 x 和 y 中，t 被放弃，因为 t 在控制串中。