scanf/fscanf 的%[]和%n使用方法

标准输入输出函数%[]和%n说明符的使用方法     scanf fscanf,均从第一个非空格的可显示字符开始读起！         标准输入输出函数scanf具有相对较多的转换说明符，它常常作为入门级函数出现在各种教材中。但奇怪的是，[]和n这两种都为c89/c99所规定的标准说明符却鲜少在大多数教材中出现。虽然[]和n说明符的使用频率不及其它说明符，但两者在程序设计中的作用仍然不可小视，尤其是[]说明符。

        众所周之，**scanf以空白字符为定界符，但如果输入的字符串是以其它字符为定界符的，那怎么办？[]就是专门处理这个问题的转换说明符。[]转换说明符可以通过两种方式产生结果字符集，如果第一个[字符右边没有抑扬符（^），那么处于[]之间的字符就是结果字符集，不在其中的可输入字符都作为定界符；如果左边[符号紧靠一个抑扬符（^），那么意义相反，^和]之间的字符是定界符，其余可输入字符是结果字符集。**

        在使用[]说明符之前，得先明白两个概念：一是扫描列表。扫描列表（scanlist）指的是包含在[和]两个字符之间除紧靠左边[字符的抑扬符之外的字符，例如：

scanf("%[abcd]", ptr);

abcd组成扫描列表。二是扫描字符集（scanset）。扫描字符集指的是结果字符集，例如上面的例子，结果字符集就是abcd。如果输入一个字符串“cbadkjf”，那么ptr得到的字符串是cbad，kjf三个字符都属于定界符，输入到k字符时输入字符串被截断，kjf三个字符被留在stdin里面。如果带有抑扬符，例如：

scanf("%[^abcd]", ptr);

扫描列表仍然是abcd，但扫描字符集是除abcd外的可输入字符。如果输入字符串“jksferakjjdf”，ptr得到的字符串是“jksfer”。**如果想限制输入字符串的字符数量，可以象s说明符那样，在[]前面使用位域，例如：**

**scanf("%10[^abcd]", ptr);**

**这样结果字符串最多只能包含10个字符（除'/0'字符外）。**

**[符号可以作为扫描列表中的一个成员，但]字符除紧贴最左边的[字符或抑扬符两种情况外，其余情况下都不会被看作扫描列表的成员。例如“%[]abcd]”或者“%[^]abcd]”，上述两种情况下]字符属于扫描列表的成员，但如果是“%[ab]cd]”，中间的]字符不会被看作扫描列表的成员，而且输入输出的结果会是乱七八糟的。**

**对于减号-，只有在紧贴[字符或抑扬字符以及作为扫描列表最后一个成员时，-字符才会被视为扫描列表的成员。c标准把其余情况规定为编译器相关的。大多数编译器把这种情况的减号定义为连字符，例如：**

**scanf("%[a-zA-Z]", ptr);**

**那么扫描列表由大小写各26个字母组成。少数编译器仍旧把这种情况下的减号视为扫描列表成员。** fscanf(fd,"%\*[^/n]/n");//**%\*是虚读，没有存，只是让指针跳过了这个变量！**

**%n说明符输出有效字符数量，%n在scanf和printf中都可使用。与%n相对应的形参是一个int类型的指针，%n不影响scanf和printf的返回值**。例如：

scanf("%d %d%n", &i, &j, &k);

如果输入434 6434，则k等于8，而scanf的返回值仍然为2。又如：

scanf("%c%n", &ch, &k);

输入“sbcdefdg”后，k等于1，而不是8，因为**%c只取一个字符，%n输出的是有效字符数量**。

**%n用在printf函数里，表示输出的字符数量**，例如：

printf("i=%d, j=%d/n%n", i, j, &k);

在i=343、j=123的情况下，k=12，同时**%n不影响printf的返回值**，其返回值仍然为12，而不是14。

==============================================================

这个用法是在参H264 jm82考代码上看到的，用来从解码器参数配置文件中读取配置参数，代码如下：

// read the decoder configuration file if((fd=fopen(config\_filename,"r")) == NULL) { snprintf(errortext, ET\_SIZE, "Error: Control file %s not found/n",config\_filename); error(errortext, 300); }

fscanf(fd,"%s",inp->infile);                // H.26L compressed input bitsream fscanf(fd,"%\*[^/n]");

fscanf(fd,"%s",inp->outfile);               // YUV 4:2:2 input format fscanf(fd,"%\*[^/n]");

fscanf(fd,"%s",inp->reffile);               // reference file fscanf(fd,"%\*[^/n]");

对应的配置文件内容如下：

test.264                 ........H.26L coded bitstream test\_dec.yuv             ........Output file, YUV 4:2:0 format test\_rec.yuv             ........Ref sequence (for SNR)

通过这种方式

inp->infile = "test.264"

inp->outfile = "test\_dec.yuv"

inp->reffile = "test\_rec.yuv"

而相应的配置文件中的一些注释则不会被读入，这是相当简便的用法，比起通过严格约定注释符并进行一个字符一个字符来解析，这种方式简单了许多！值得借鉴！

===================================================================

scanf  语法:    #include <stdio.h>   int scanf( const char \*format, ... );  类似函数有        int scanf(const char \*format, ...);        int fscanf(FILE \*stream, const char \*format, ...);//指定输入流        int sscanf(const char \*str, const char \*format, ...);//指定缓存区 scanf()函数根据由format(格式)指定的格式从stdin(标准输入)读取,并保存数据到其它参数. 它和printf()有点类似. format(格式)字符串由控制字符,空白字符和非空白字符组成. 控制字符以一个%符号开始,如下:

**控制字符 说明  %c 一个单一的字符  %d 一个十进制整数  %i 一个整数  %e, %f, %g 一个浮点数  %o 一个八进制数  %s 一个字符串  %x 一个十六进制数  %p 一个指针  %n 一个等于读取字符数量的整数  %u 一个无符号整数  %[] 一个字符集  %% 一个精度符号**

1.scanf()读取匹配format(格式)字符串的输入. 当读取到一个控制字符, 它把值放置到下一个变量. 空白(tabs, 空格等等)会跳过. 非空白字符和输入匹配, 然后丢弃. 如果是一个在%符号和控制符间的数量, 那么只有指定数量的字符转换到变量中. 如果scanf()遇到一个字符集(用%[]控制字符表示), 那么在括号中的任意字符都会读取到变量中. scanf()的返回值是成功赋值的变量数量, 发生错误时返回EOF.  2.scanf()函数的一般格式为：scanf("格式字符串"，输入项首地址表) 3.scanf的格式控制的一般形式为：%[\*][宽度][F|N][h|l]类型字符 []中的控制字符为可选项  4."\*"表示该输入项读入后不赋予任何变量，即跳过该输入值。 5."宽度"表示输入读入字符的长度，对于整型表示截取相应宽度的数字赋给后面列表中的相应变量；对于字符型表示读入相应长度的字符后把第一个字符赋给相应的变量，其余的自动舍弃。例如scanf("%2d%3d",&a, &b);如果输入为12345则将12赋给a，将45赋给b；scanf("%2c%3c",&a, &b);如果输入为12345则将'1'赋给a，将'3'赋给b . "%s" 整个输入作为一个串,并设置末尾的'/0' "%ns",n为整数,读入的串最长不超过n,然后在末尾补'/0' %nf 读入的浮点数最多有n位整数,位数多于n,会截断。 "%n[a-z]" 读入最多n个字符,如果遇到非a-z的字符,停止 "%[^=]" 读入任意多的字符,直到遇到"="停止 "%n[^=]" 读入"="号前的至多n 个字符 6.F 、N、h、l分别表示远指针、近指针、短整和长整型。 7.对于输入字符串还有一些比较有用的控制。     例如经常**需要读入一行字符串，而这串字符里面可能有空格、制表符等空白字符，如果直接用%s是不可以的，于是有些人就想到用gets(),当然这也是一种选择，但是懂C的人基本上都知道gets()是一个很危险的函数，而且很难控制，特别是与scanf()交替使用时前者的劣势更是一览无余，所以gets()一般是不推荐用的**，其实用%[^/n]就可以很好的解决这个问题了，^表示"非"，即读入其后面的字符就结束读入。这样想**读入一行字符串直接用scanf("%[^/n]%\*c",str);就可以了， %\*c的作用是读入/n,否则后面读入的将一直是/n。**所有对%s起作用的控制都可以用%[],比如**%[0-9]表示只读入'0'到'9'之间的字符，%[a-zA-Z]表示只读入字母**， '-'是范围连接符，当然也可以直接列出你需要读入的字符。 如果你只需要读"abc"里面的字符就可以用%[abc] (或者%[cab]、%[acb]、%[a-c]、%[c-a].....), 如果想读入某个范围之外的字符串就在前面加一个'^',如：%[^a-z]就表示读入小写字母之外的字符。 例如从键盘输入的"1235ab86"中读取1235、86给n,有如下方法：

#include <stdio.h>

bool skip(){

scanf("%\*[^0-9]");

return true;

}

void main()

{

int n;

while(skip() && scanf("%d", &n)!=EOF)

printf("%d/n", n);

}

输出为：

1235 86