**使用scanf函数时应该注意的问**

**1、sacnf()中的变量必须使用地址。**  
**2、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。**

**3、在用"%c"输入时，空格和“转义字符”均作为有效字符。**

  问题一：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()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。  
  ③ 遇非法输入。  
  所以，上述程序并不能达到预期目的，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;  
  }  
  问题二：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');  
  }  
  scanf("%c",&c);这句不能正常接收字符,什么原因呢？我们用 printf("c=%d\n",c);将C用int表示出来，启用printf("c=%d\n",c);这一句，看看scanf()函数赋给C到底是 什么，结果是c=10 ,ASCII值为10是什么？换行即\n.对了，我们每击打一下"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.解决办法：可以在两个scanf()函数之 后加个fflush(stdin);，还有加getch() , getchar()也可以，但是要视具体scanf()语句加那个，这里就不分析了，读者自己去摸索吧。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：  
  #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\*/  
  }  
  }  
  可以运行看看两个程序有什么不同。  
  问题三： 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);  
  }  
  如上程序，如果正确输入a,b的值，那么没什么问题，但是，你不能保证使用者每一次都能正确输 入，一旦输入了错误的类型，你的程序不是死锁，就是得到一个错误的结果,哈哈，这可能所有人都遇到过的问题吧？解决方法：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);  
  }  
  补充： fflush(stdin)这个方法在GCC下不可用。（在VC6.0下可以）  
  以下是 C99 对 fflush 函数的定义：  
  int fflush(FILE \*stream);  
  如果stream指向输出流或者更新流（update stream），并且这个更新流  
  最近执行的操作不是输入，那么fflush函数将把任何未被写入的数据写入stream  
  指向的文件（如标准输出文件stdout）。否则，fflush函数的行为是不确定的。  
  fflush（NULL）清空所有输出流和上面提到的更新流。如果发生写错误，fflush  
  函数会给那些流打上错误标记，并且返回EOF，否则返回0。  
  由此可知，如果 stream 指向输入流（如 stdin），那么 fflush 函数的行为是不确定的。故而使用  
  fflush(stdin) 是不正确的，至少是移植性不好的。  
  可采用如下方法：  
  /\* 此函数完全可以和scanf函数一起使用，无论输入何种类型的数据，但不能和gets一起使用 \*/  
  void flush()  
  {  
  char c;  
  while ((c=getchar()) != '\n'&&c!=EOF) ;  
  }  
  #include <stdio.h>  
  int main()  
  {  
  int a,b,c; /\*计算a+b\*/  
  while(scanf("%d,%d",&a,&b)!=2) flush();  
  c=a+b;  
  printf("%d+%d=%d",a,b,c);  
  }