以下引用D题蒸羊的代码：

因为一个小地方写的有问题所以是WA

#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

int main()

{

char a;

int b[100001],i=0,hold;

while(scanf("%d",&b[i])!=EOF)

{

scanf("%c",&a); //这里用scanf(“%c”,&a);会出错。。

用a=getchar();就会AC 呵呵哒

if(a==EOF)

printf("YES\n");

if(a==' ')

i++;

else

{

i++;

for(int s=0;s<i;s++)

{

for(int k=0;k<i-1-s;k++)

{

if(b[k]<b[k+1])

{

hold=b[k+1];

b[k+1]=b[k];

b[k]=hold;

}

}

}

for(int j=0;j<i;j++)

cout<<b[j]<<" ";

cout<<endl;

memset(b,0,sizeof(b));

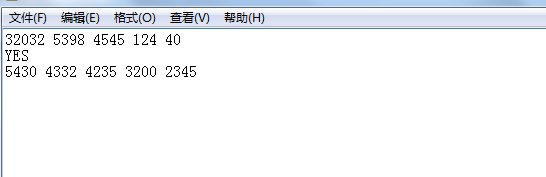
i=0;

}

}

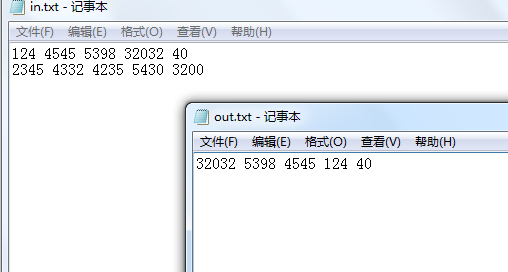
}

这个文件的输出结果为



这说明第二组数据到达了文件末尾，所以c读到了EOF，所以输出了YES。注意YES是第二组数据遇到了文件末尾，所以才输出的。

而将a=getchar();改成scanf(“%c”,&a);以后再测试输出结果。



结果是只输出了一组数据的结果。即第一组数据的结果。所以这就是这个程序出现WA的原因。

因此说明scanf(“%c”,&a);和a=getchar();在读取字符的时候是不同的。

通过这个测试说明scanf(“%c”,&c);是不能读到EOF这个字符的。

还有这个程序:

#include <stdio.h>

int main()

{

char ch;

while(scanf("%c", &ch) != EOF) {

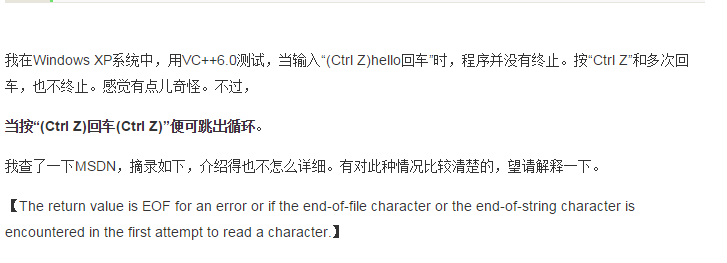
printf("233");

}

return 0;

}

你按回车是不会跳出循环的。。而是会一直执行。。



这个在机房的XP系统上实测确实按Ctrl Z也无法跳出循环。但是我的win7的电脑按了Ctrl Z却可以跳出。但是只按回车两个系统都是无法跳出循环的。

所以大家只需要记住，scanf(“%c”,&c);不能读入EOF。那我们用scnaf(“%c”,&a);在遇到文件末尾时到底读入的是什么呢？

我们把程序略加修改，输出这个字符的值。

#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

int main()

{

char a;

freopen("in.txt","r",stdin);

freopen("out.txt","w",stdout);

int b[100001],i=0,hold;

while(scanf("%d",&b[i])!=EOF)

{

//a=getchar();

scanf("%c",&a);

**printf("%d\n",(int)a);**

if(a==EOF)

printf("YES\n");

if(a==' ')

i++;

else

{

i++;

for(int s=0;s<i;s++)

{

for(int k=0;k<i-1-s;k++)

{

if(b[k]<b[k+1])

{

hold=b[k+1];

b[k+1]=b[k];

b[k]=hold;

}

}

}

for(int j=0;j<i;j++)

cout<<b[j]<<" ";

cout<<endl;

memset(b,0,sizeof(b));

i=0;

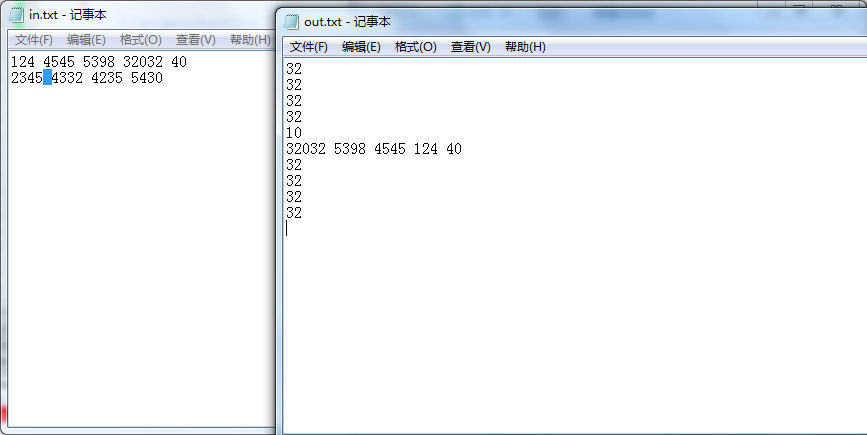
}

}

if(a=='\0')

printf("YES\n");

}



这就是神奇的地方，我们发现第一个数据之间有四个空格，一个回车符。所以第一次输出了四个32，32是空格的ASCII码。10是’\n’的ASCII码。而而第二组却不对。第二组数据我没有去掉最开始的这个数据末尾的3200的时候会输出5个空格。而我在数据3200末尾没有打空格符号。

去掉3200以后，我保证5400这个数字之后没有空格。但是scanf(“%c”,&a);这个程序还会输出4个32，也就是4个空格。这是为什么呢？

明明只有三个空格，可是为何会出现4个呢？

今天之前我也不知道，但是现在我测试完，我彻底懂了。。等我讲完，我希望你们透彻的明白scanf的机理。

其实这第4个32是第3个32的值。因为scanf会吧数据输入到缓冲区中，而printf函数则会从缓冲区中取出这些值。

最后一次遇到EOF的时候，scanf读取字符失败了，所以printf从缓冲区中只能拿上一次输入的那个字符。而那个字符就是第三个空格。

这是怎么测试出来的呢？

我教你们。

#include<cstdio>

#include<iostream>

#include<cstring>

using namespace std;

int main()

{

char a;

freopen("in.txt","r",stdin);

freopen("out.txt","w",stdout);

int b[100001],i=0,hold;

while(scanf("%d",&b[i])!=EOF)

{

//a=getchar();

scanf("%c",&a);

**if(scanf("%c",&a)==-1)**

**printf("jhljx is doubi\n");**

printf("%d\n",(int)a);

if(a==EOF)

printf("YES\n");

if(a==' ')

i++;

else

{

i++;

for(int s=0;s<i;s++)

{

for(int k=0;k<i-1-s;k++)

{

if(b[k]<b[k+1])

{

hold=b[k+1];

b[k+1]=b[k];

b[k]=hold;

}

}

}

for(int j=0;j<i;j++)

cout<<b[j]<<" ";

cout<<endl;

memset(b,0,sizeof(b));

i=0;

}

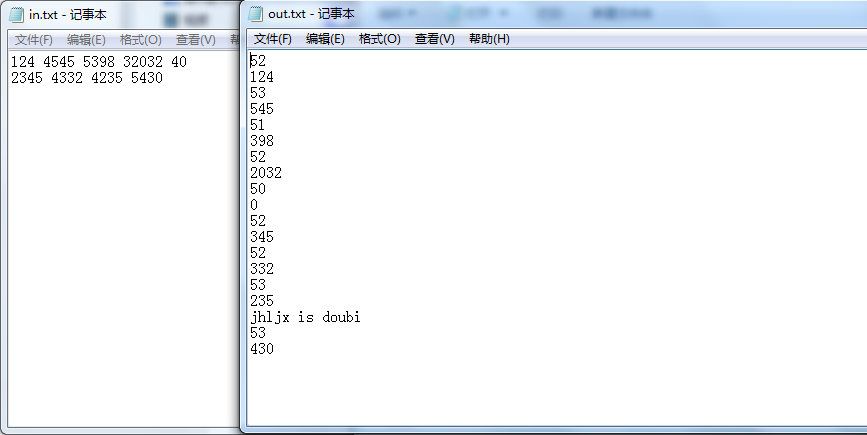
}

if(a=='\0')

printf("YES\n");

}

加了那行语句。



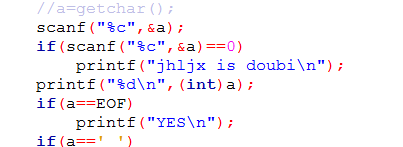
结果确实输出了jhljx is doubi

这就说明scanf读取字符失败。说明我的猜测是正确的。

现在我们知道了scanf的机理。我刚刚百度了一下，scanf读取数据是有返回值的，这个返回值是scanf函数接受到的变量的个数。



但是scanf读取到EOF的时候是返回-1的，不会返回0.所以你写成



这样是不会输出jhljx is doubi的。

Scanf读到EOF说明读取失败。这时就应该退出循环。也就是scanf就读不到EOF,所以第二组数据的排序结果不会输出。

所以好多人只是判断了’\n’，结果莫名其妙WA掉了。

因为最后一组数据没有’\n’，所以会跪。

这就是被EOF坑了。。

而getchar是可以读取所有字符的。所以这个就可以AC。

所以你们说你在本地测试正确，交上去就WA。原因就在这里。这就说明控制台测试，即用小黑框测试你的代码是否正确是不全面的。要学会用文件读写来测试自己的结果。

因为在控制台下面我们无法读入到文件结束。而大多数时候是以回车来结束一次输入的。这就是控制台和文件的区别。有文件就要考虑文件末尾的EOF，而要牢记EOF不能用scanf读入。

你们大多数人被坑的原因就是这个。

这是一种方法，还有一种方法是全部用字符串读入。

这样又会出现不同。

所以在读取字符的时候就会出现这些神奇的地方，这也是你们掌握不好字符的原因之一。我希望我能尽可能解释清楚。

#include<cstdio>

#include<iostream>

#include<cstring>

#include<algorithm>

#define maxn 1000100

using namespace std;

char a[maxn];

int b[maxn];

bool cmp(int c,int d)

{

return c>d;

}

int main()

{

//ios\_base::sync\_with\_stdio(0);

freopen("in1.txt","r",stdin);

freopen("out.txt","w",stdout);

while(gets(a))

{

int k=strlen(a),i=0,num=0,s=0;

bool flag=0;

printf("233 %d\n",(int)a[k]);

if(a[k]=='\0')

printf("YES\n");

while(i<k)

{

if(a[i]>='0'&&a[i]<='9')

{

num=num\*10+a[i]-'0';

if(i==k-1)

b[s++]=num;

i++;

}

else

{

b[s++]=num;

num=0;

i++;

}

}

sort(b,b+s,cmp);

// for(int i=0;i<s-1;i++)

// {

// for(int j=0;j<s-1-i;i++)

// {

// if(b[j]<b[j+1])

// {

// swap(b[j],b[j+1]);

// }

// }

// }

for(int i=0;i<s;i++)

printf("%d ",b[i]);

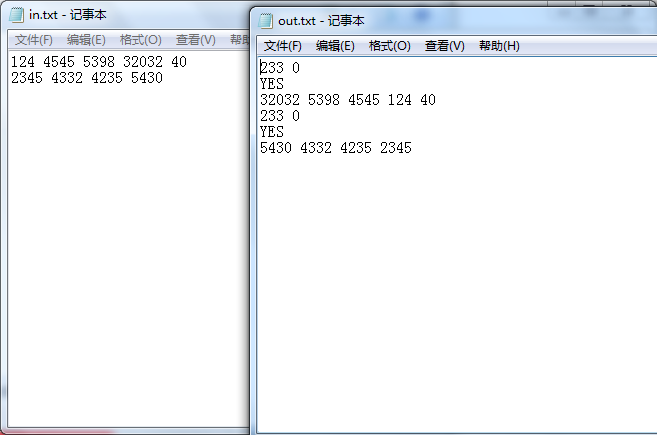
printf("\n");

}

}

这是D题用字符串读入的代码，是最简单粗暴的做法。

输出结果是这个：



这说明两次字符串末尾都会有一个’\0’存在。这个是输入字符串的时候系统自动添加的。

所以在你们期末笔试可能会出现这样的题，

char a[]=”abcde”和char b[]=｛’a’,’b’,’c’,’d’,’e’,’f’｝;这两个的长度实际上是一样的。PS:这个知识点去年期末笔试考试选择题出现过。。当时我就跪了。。Orz

所以要考虑第一个字符串的长度要考虑’\0’



这就说明两种做法在读入数据的时候差异会很大，第一种做法在读到末尾会读到’\n’而用字符串则会读到’\0’,而不是’\n’。

通过比较我们也发现了很多的问题。

尤其是在第一种做法中要对末尾EOF处理。所以getchar来读入单个字符当然是正确的。。因为它可以读入EOF。可以读入所有字符。

这也就是我通过本题测试出来的一些结论。通过几个小时的探究也发现了不少东西，在这里和大家分享。我觉得这种探究的方法和自我学习的方法还是很重要的，因为你会发现你在自我探索，你也会很快乐。

还有这个有用的东西。



还有一些博客：

<http://blog.csdn.net/zy799894671/article/details/7757764>

<http://blog.csdn.net/hanchengxi/article/details/8591663>（很好，推荐看一看）