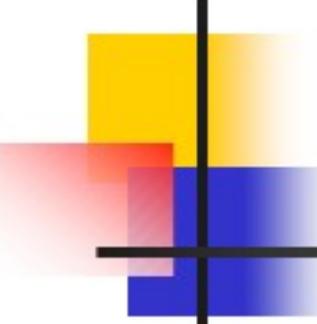


OnlineJudge基本输入输出

湖南师范大学 瞿绍军
powerhope@tom.com

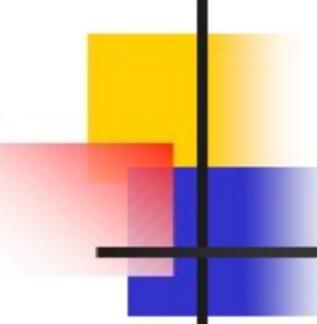




评测系统反馈：

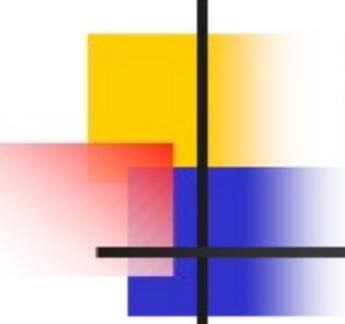
ACM国际大学生程序设计竞赛中，自动评测系统给你的反馈信息极少，主要包括：

- Accepted (AC)：正确。祝贺你！你的程序是正确的，并且运行时间和内存开销均不超过相应的限制。
- Presentation Error (PE)：格式错。你的程序输出了正确的结果，但是输出格式并非严格满足题目说明。请检查空格、换行、左右对齐等。
- Wrong Answer (WA)：答案错。你的程序对一组或者多组评测系统内部的非公开测试数据给出了错误的结果，因此还需要更进一步的调试。
- Time Limit Exceeded (TLE)：超时。你的程序在某一组或多组数据上运行的时间过长，该程序可能存在效率问题。



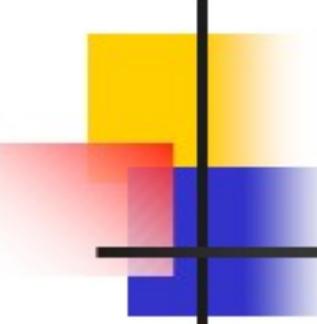
评测系统反馈：

- **Runtime Error (RE)** : 运行错。你的程序在运行结束之前由于段错误(Segmentation fault)、访问冲突(ACCESS VIOLATION)、数组下标超出范围(ARRAY BOUNDS EXCEEDED)、浮点异常(FLOAT POINT EXCEPTION)或其他类是错误、除零错误(DIVISION BY ZERO)、栈溢出(STACK OVERFLOW)等。请仔细检查程序中类似的错误。
- **Memory Limit Exceeded (MLE)** : 超内存。你的程序试图使用超过评测系统规定大小的内存。
- **Output Limit Exceeded (OLE)** : 超输出。你的程序输出了过多的信息。这通常意味着你的程序陷入了一个带输出的无限循环中。



评测系统反馈：

- **Compile Error (CE)**：编译错。编译器无法成功地编译你的程序，错误信息将返回给用户。编译过程中产生的警告信息不会导致此错误。
- **System Error**: 系统错。比如你的程序需要超过了硬件限制的内存。

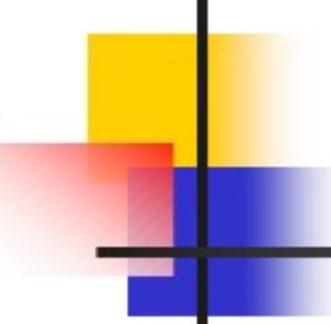


先看一个超级简单的题目：

- 10000
- Sample input:
 - 10 20
- Sample output:
 - 30



```
■ #include <iostream>
■ using namespace std;
■ int main()
■ {
■     int a, b;
■     cin >> a >> b;
■     cout << a + b << endl;
■     return 0;
■ }
```



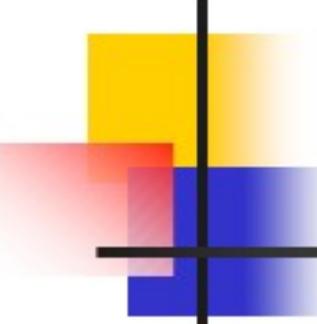
输入_第一类:

- 输入不说明有多少个Input Block, 以EOF为结束标志。
参见: 10361



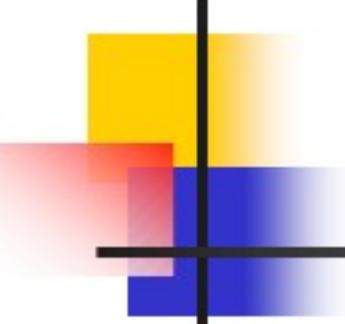
C源代码：

```
# include <stdio.h>
int main()
{
    int a,b;
    while(scanf("%d %d",&a, &b) != EOF)
        printf("%d\n",a+ b);
    return 0;
}
```



C++ 源代码：

- #include<iostream>
- using namespace std;
- int main()
- {
- int a, b;
- while(cin >> a >> b)
- cout<< a + b<< endl;
- return 0;
- }



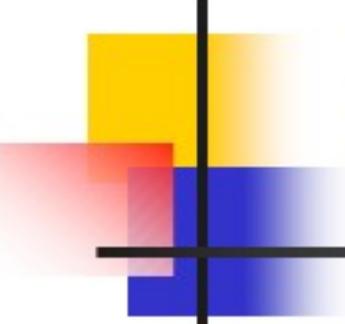
本类输入解决方案：

- C语法：

```
while(scanf("%d %d",&a, &b) != EOF)
{
    ....
}
```

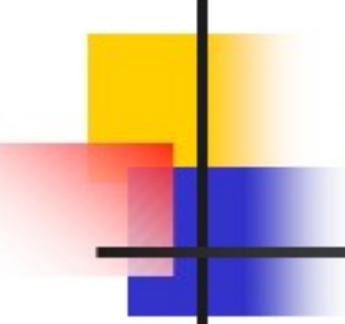
- C++ 语法：

```
while( cin >> a >> b )
{
    ....
}
```



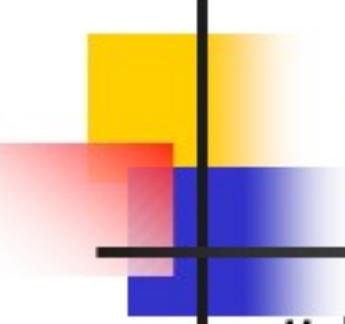
说明(1)：

1. **Scanf**函数返回值就是读出的变量个数，
如： `scanf("%d %d", &a, &b);`
如果只有一个整数输入， 返回值是1，
如果有两个整数输入， 返回值是2， 如
果一个都没有， 则返回值是-1。
2. **EOF**是一个预定义的常量， 等于-1。



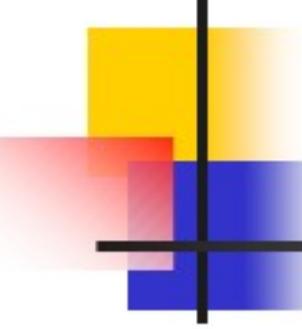
输入 第二类：

- 输入一开始就会说有N个Input Block, 下面接着是N个Input Block。
参见：10362



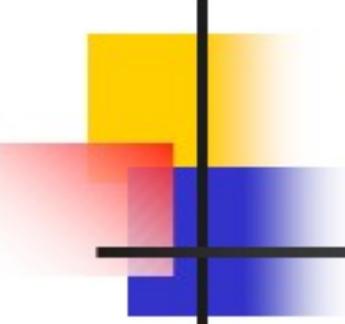
C源代码：

```
# include < stdio.h>
int main()
{
    int n,i,a,b;
    scanf("%d",&n);
    for(i=0;i<n;i+ +)
    {
        scanf("%d %d",&a, &b);
        printf("%d\n",a+ b);
    }
    return 0;
}
```



C++源代码：

```
# include< iostream>
using namespace std;
int main()
{
    int n,a, b;
    cin>>n;
    while(n--)
    {
        cin >> a >> b;
        cout<< a + b<< endl;
    }
    return 0;
}
```



本类输入解决方案：

- C语法：

```
scanf("%d",&n) ;  
for( i=0 ; i<n ; i++ )  
{
```

```
....
```

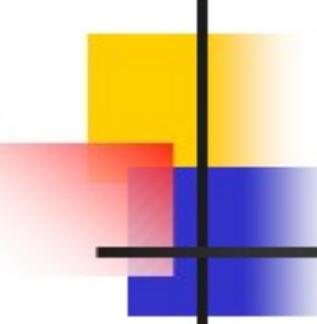
```
}
```

- C++ 语法：

```
cin >> n;  
for( i=0 ; i<n ; i++ )  
{
```

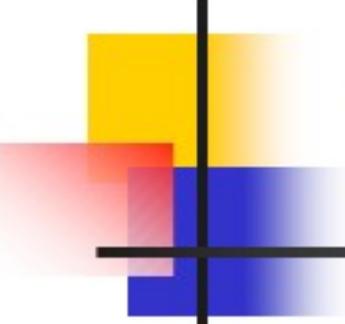
```
....
```

```
}
```



输入_第三类:

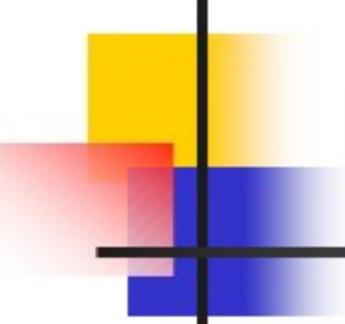
- 输入不说明有多少个Input Block,但以某个特殊输入为结束标志。
参见: 10363
- 3



源代码：

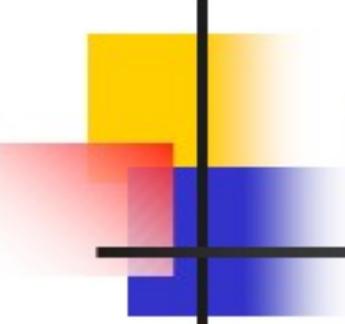
```
# include <stdio.h>
int main()
{
    int a,b;
    while(scanf("%d %d",&a, &b) &&(a!= 0 && b!= 0))
        printf("%d\n",a+ b);
    Return 0;
}
```

上面的程序有什么问题？



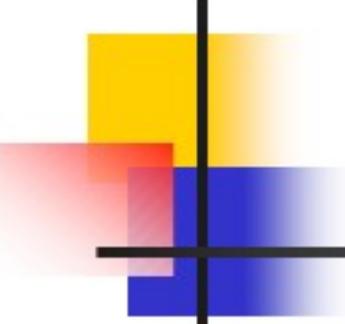
C源代码：

```
# include <stdio.h>
int main()
{
    int a,b;
    while(scanf("%d %d",&a, &b) &&(a||b))
        printf("%d\n",a+ b);
    return 0;
}
```



C++源代码：

```
# include< iostream>
using namespace std;
int main()
{
    int a,b;
    while(cin>>a>>b && (a||b) )
        cout<<a+b<<endl;
    return 0;
}
```



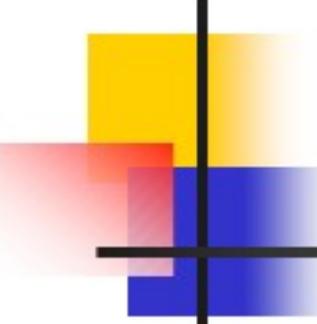
本类输入解决方案：

- C语法：

```
while( scanf("%d",&n) && n!=0 )  
{  
    ....  
}
```

- C++ 语法：

```
while( cin >> n && n != 0 )  
{  
    ....  
}
```



输入_第四类:

- 以上几种情况的组合
- 10364
- 10365
- 10366

4源代码：

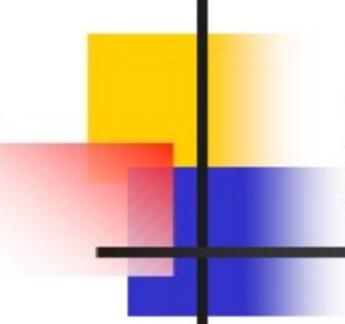
```
#include<stdio.h>
int main()
{
    int n,sum;
    while( scanf( "%d" , &n ) && n )
        //每组case前有一个控制该组输入数据的数，为0结束
    {
        int x;
        sum = 0;
        while( n-- ) //控制该组输入个数
        {
            scanf( "%d" , &x );
            sum += x;
        }
        printf( "%d\n" , sum ); //一行一个结果
    }
    return 0;
}
```

5源代码：

```
# include <stdio.h>
int main()
{
    int casnum,n,sum;
    scanf( "%d" , &casnum ); //控制总的输入case的数
    while( casnum-- ) //控制总的输入个数
    {
        int x;
        sum = 0;
        scanf( "%d" , &n ); //每个case中控制该组输入个数
        while( n-- )
        {
            scanf( "%d" , &x );
            sum += x;
        }
        printf( "%d\n" , sum ); //一行一个结果
    }
    return 0;
} 2010-12-23
```

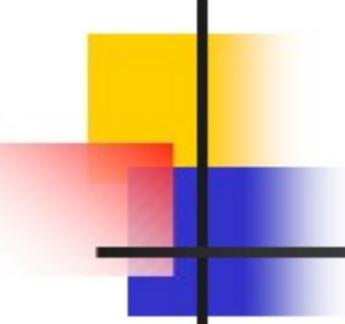
6源代码：

```
#include <stdio.h>
int main()
{
    int n,sum;
    while( scanf( "%d" , &n ) != EOF ) //输出到文件结尾
    {
        int x;
        sum = 0;
        while( n-- ) //控制该组输入个数
        {
            scanf( "%d" , &x );
            sum += x;
        }
        printf( "%d\n" , sum ); //一行一个结果
    }
    return 0;
}
```



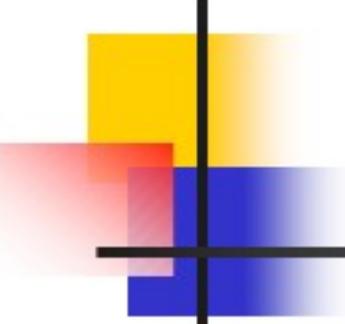
输入 第五类：

- 输入是一整行的字符串的
参见：10367



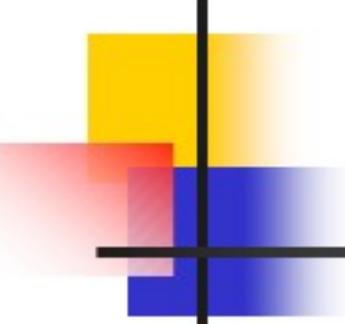
本类输入解决方案：

```
■ #include <stdio.h>
■ #include <string.h>
■ int main(void){
■     int i;
■     char a[15],b[205];
■     while(1){
■         gets(a);
■         if(strcmp(a,"ENDOFINPUT") == 0)
■             break;
■         gets(b);
■         gets(a);
■     }
■ }
```



本类输入解决方案：

```
■     for(i= 0;b[i] != '\0';i+ +){  
■         if(b[i]> = 65&&b[i]< = 90){  
■             b[i]-= 5;  
■             if(b[i]< 65) b[i]+ = 26;  
■         }  
■         printf("% c",b[i]);  
■     }  
■     printf("\n");  
■ }  
■ return 0;  
■ }
```



本类输入解决方案：

- C语法：

```
char buf[20];  
gets(buf);
```

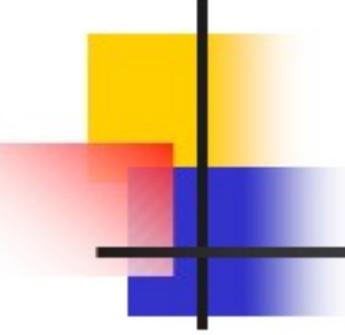
- C++ 语法：

如果用string buf; 来保存：

```
getline( cin , buf );
```

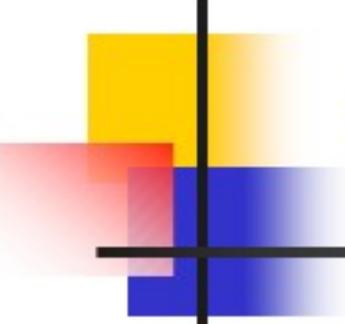
如果用char buf[255]; 来保存：

```
cin.getline( buf, 255 );
```



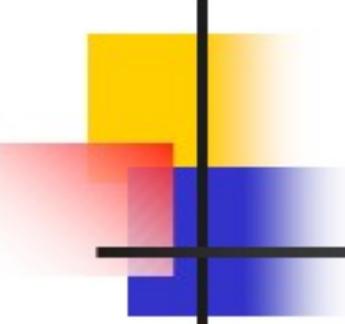
说明：

- `scanf(" %s%s",str1,str2)`, 在多个字符串之间用**一个或多个空格**分隔;
- 若使用**gets**函数, 应为`gets(str1); gets(str2);`; 字符串之间用**回车符**作分隔。
- 通常情况下, 接受**短**字符用**scanf**函数, 接受**长**字符用**gets**函数。
- 而**getchar**函数每次只接受**一个**字符, 经常`c=getchar()`这样来使用。



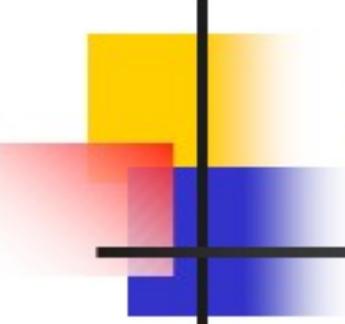
说明： cin.getline的用法：

- **getline** 是一个函数，它可以接受用户的输入的字符，直到已达指定个数，或者用户输入了特定的字符。它的函数声明形式（函数原型）如下：
`istream& getline(char line[], int size, char endchar = '\n');`
- 不用管它的返回类型，来关心它的三个参数：
- **char line[]**: 就是一个字符数组，用户输入的内容将存入在该数组内。
- **int size** : 最多接受几个字符？用户超过size的输入都将不被接受。
- **char endchar** :当用户输入endchar指定的字符时，自动结束。默认是回车符。



说明续

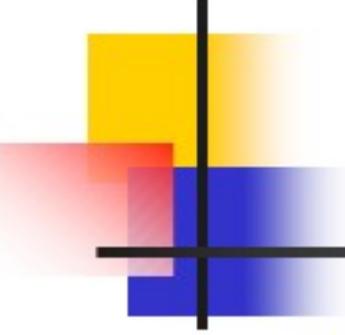
- 结合后两个参数，`getline`可以方便地实现：
用户最多输入指定个数的字符，如果超过，
则仅指定个数的前面字符有效，如果没有超
过，则用户可以通过回车来结束输入。
- `char name[4];`
- `cin.getline(name,4,'\\n');`
- 由于 `endchar` 默认已经是 '`\\n`'，所以后面那
行也可以写成：
- `cin.getline(name,4);`



输出_第一类:

- 一个Input Block对应一个Output Block，Output Block之间**没有**空行。

参见：10361



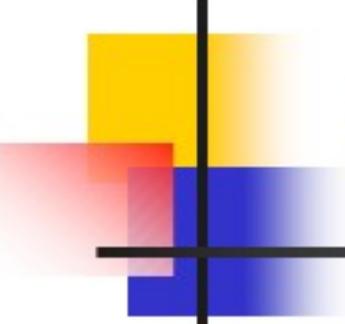
解决方案：

- C语法:

```
{  
    ....  
    printf("%d\n",ans);  
}
```

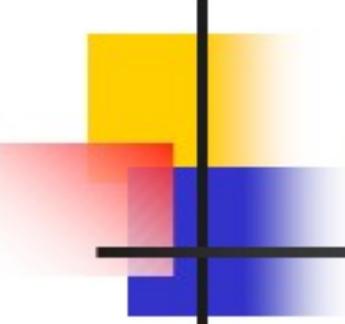
- C++ 语法:

```
{  
    ...  
    cout << ans << endl;  
}
```



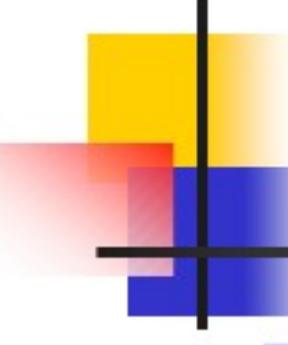
输出_第二类：

- 一个Input Block对应一个Output Block，
每个Output Block之后都有空行。
参见：10368



8源代码

```
# include <stdio.h>
int main()
{
    int a,b;
    while(scanf("%d %d",&a, &b) != EOF)
        printf("%d\n\n",a+b);
    return 0;
}
```



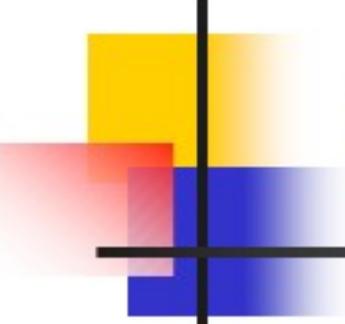
解决办法：

- C语法：

```
{  
    ....  
    printf("%d\n\n",ans);  
}
```

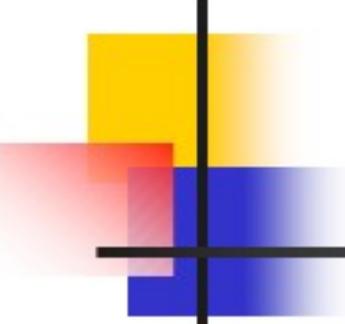
- C++ 语法：

```
{  
    ...  
    cout << ans << endl << endl;  
}
```



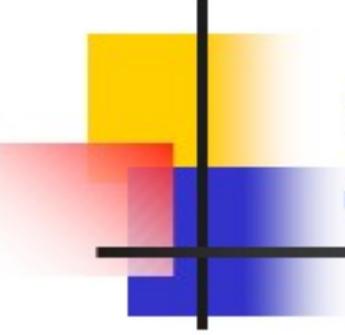
输出_第三类:

- 一个Input Block对应一个Output Block, Output Block**之间有**空行。
参见: 10369



9 源代码

```
#include <stdio.h>
int main()
{
    int icase,n,i,j,a,sum;
    scanf("%d",&icase);
    for(i=0;i<icase;i++)
    {
        sum=0;
        scanf("%d",&n);
        for(j=0;j<n;j++)
        {
            scanf("%d",&a);
            sum+=a;
        }
        if(i<icase-1)
            printf("%d\n\n",sum);
        else
            printf("%d\n",sum);
    }
    return 0;
}
2010-12-23
```



三、C语言处理“混合数据”的问题

10370



C++源代码：

- # include<iostream>
- # include<iomanip>
- using namespace std;
- int main()
- {
- char a;
- int n,b,c;
- double sum=0;
- while((cin>>n)&&n)
- {
- while(n--)
- {

```
cin>>a>>b>>c;
switch(a)
{
    case '+':sum=b+c;cout<<sum<<endl;break;
    case '-':sum=b-c;cout<<sum<<endl;break;
    case '/':if(b%c==0)
                cout<<b/c<<endl;
    else
        cout << setiosflags(ios::fixed)
            << setprecision(2)
            << float(b)/float(c)<< endl;
    break;
```

```
■         case '*':sum=b*c;cout<<sum<<endl;break;  
■     }  
■ }  
■ }  
■ return 0;  
■ }
```