1 关于 ACM 的输入输出（一）

写给第一次参加现场赛的同学们

一般来说 ACM 的现场赛会规定输入输出

或者是文件输入标准输出

也可能是文件输入文件输出

如果没有规定的话那么一般就是标准的输入输出了

那说一下输入输出的重定向

一般用下面两种方法

c++常用:

```
#include <fstream.h>
ifstream filein("data.in");  // 定义一个文件输入流
ofstream fileout("data.out"); //cout<< --> fileout<<
filein.eof() //文件到末尾,返回非零值
```

data.in 表示输入的数据文件

本地测试的话本来输入的数据就要在这个文件里面测试了

建一个本地的文本 data.in,可以用记事本的方式打开

注意:文件输入的话,以后的 cin>>都要改成 filein>>, cout<<都要改成 fileout<<

c 语言常用:

```
freopen("date.in","r",stdin);  //重定向所有标准的输入为文件输入
freopen("date.out","w",stdout);//重定向所有标准的输出为文件输出
fclose(stdout);//输出结束
```

```
freopen("date.in","r",stdin); //重定向所有标准的输入为文件输入
freopen("date.out","w",stdout);//重定向所有标准的输出为文件输出
fclose(stdout);//输出结束
```

第一句的意思就是文件输入,以"读状态",去替换标准的输入

以上如果只是规定用文件输入输出 的某一种,那么就只用其中的一种

2 关于 ACM 的输入输出（二）

ACM 题目特点: 由于 ACM 竞赛题目的输入数据和输出数据一般有多组（不定），并且格式多种多样，所以，如何处理题目的输入输出是对大家的一项最基本的要求。这也是困扰初学者的一大问题。 ACM 的输入输出要求严格按照规定来，所以你不需要输出像"Please input the data"这类的提示语。否则将会被判 Wrong Answer。

1、输入

初学者一般有个误区：如果题目包含多组测试数据，他们就会把输入的内容全部保存起来，然后再依次处理。

其实程序的输入/输出是相互独立的，因此，每当处理完一组测试数据，就应当按题目要求进行相应的输出操作。而不必将所有结果储存起来一起输出。

下面来介绍一下 ACM 中常见的一些输入情况。

只有一组测试数据

这类题目是最简单的，比如第 1000 题。参考代码：

```
#include
int main(void)
{
```

```
int a, b;
scanf("%d %d", &a, &b);
printf("%d/n", a + b);
return 0;
}
```

没有明确指出输入什么时候结束

如果是这种情况，我们默认是以“文件结束”(EOF)为结束标志。

这是 ACM 的默规，例如 1076 题。参考代码：

```
#include
int main(void)
{
int a, b;
while (scanf("%d %d", &a, &b) != EOF)
printf("%d/n", a + b);
return 0;
}
```

指定数据量

有时会在数据的第一行提供数据量大小，比如第一行是 100，则表示有 100 组数据。比如第 1077 题。参考代码：

```
#include
int main(void)
{
int n, a, b;
scanf("%d", &n);
```

```
while (n--)
{
scanf("%d %d", &a, &b);
printf("%d/n", a + b);
}
return 0;
}
```

以特定元素作结束符

这种输入和第一种类似。常见的是规定以 0 作为结束符。

比如第 1078 题。参考代码：

```
#include
int main(void)
{
int a, b;

while (scanf("%d %d", &a, &b), a || b)
printf("%d/n", a + b);

return 0;
}
```

输出

输出格式统一

这种比较简单，只要按要求来就没问题的。

比如每组输出占一行，或者每组输出后面加一个空行。比如 1000 题。

数据之间有空行

对于这种输出，有时候还会告诉你有几组输入，这样你就可以自己判断一下是不是最后一组。是就不输出空行，否则多输出一个空行。而有时候连共有几组数据都不会告诉你。其实不论知不知道有几组数据，我们都可以这样处理。

第一组数据后面不加空行。

第二组开始，每组前面加空行。 比如第 1079 题，参考代码：

```
#include
int main(void)
{
int a, b, i = 0;
while (scanf("%d %d", &a, &b), a || b)
printf((i++? "/n%d/n": "%d/n"), a + b);
return 0;
}
```

3 关于 ACM 的输入输出（三）

在线判决系统是机器判题系统，也就是俗称的 OJ（Online Judge），机器判决的一个特点就是必须 100%的吻合才能判为正确，否则要么 WA,PE。同时对于提交的程序还有一定的时间限制，如果超过时间则会判超时。OJ 一般采用的是标准输入输出，所以提交的

时候我们不必要使用文件读入输出（这与高中的信息学是不同的），机器判决只针对程序结果，不针对程序，所以很多时候直接提交数据也是可以的，俗称打表。

下面介绍常用的处理输入的方法

几种常用的处理输入方法(C 语言)

感觉新人对于处理输入输出存在一些问题,这里写出几个常用到的处理方法:

1.知道输入数据组数 n

```
scanf("%d",&n);
whlie(n--){
   这里处理每一组输入.然后直接按格式输出,没必要开数组存储答案.
}
```

2.没有数据总数,以 EOF 结束

可能用的几个函数:

scanf():

```
while(scanf("%s|%d")!=EOF){
  处理每一组数据,并输出.
}
```

getchar():读入一个字符

```
whlie((ch=getchar())!=EOF){
```

}

gets():读入一行

while(gets(buf)!=NULL) {

}

用 getchar,gets 注意读入换行符.

3.以 0 或-1 结束的输入.

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

}

关于 C++的输入输出处理：

cin 读字符串时遇到空白符（空格，换行等）结束

```
char str[BUFFER];
while (cin >> str) {
}
```

getline 读字符串时遇到换行符结束,用于读一整行

```
char str[BUFFER];
while (cin.getline(str, BUFFER)) {
}
string str;
while (getline(cin, str)) {
}
```

cin/cout 要比 scanf/printf 慢一些，尽可能使用 scanf/printf 以避免测试大量数据时因为输入输出慢而导致 TLE. putchar/getchar 要比 scanf/printf 更快

关于 java 的输入输出处理：

如果使用 BufferedReader(jdk1.1 或以后的版本，一次读一整行字符串，类似于 gets)

```
BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));
String s;
while ((s = stdin.readLine()) != null) {
   可以用 StringTokenizer st = new StringTokenizer(s);来按空格切词
   int n = Integer.parseInt(st.nextToken());
   double b = Double.parseDouble(st.nextToken());
}
```

如果使用 Scanner(仅限于 jdk1.5 或以后的版本，一般用于从字符串中切词，类似于 cin)

```
Scanner stdin = new Scanner(System.in);
while (stdin.hasNext()) {
   String s = stdin.next();
   int n = stdin.nextInt();
   double b = stdin.nextDouble();
}
```

至于输出，很多新手总会选择先将答案存储在一个数组里，等程序运行完再输出，其实这是没有必要的，机器判决是逐个字符匹配，所以完全可以处理一组输入后，便输出结果。

ACM 技巧 使用文件输入输出方便测试的方法

把下面两块宏语句分别嵌在 main 函数的开始和结束，这样在本地调试的时候，cin/cout 和 scanf/printf 直接对应到指定的文件流，但提交到 OJ 时，此两句不被编译，所以仍为标准 I/O 流，因此不用提交前改代码。

后面一块宏不用也可以，前面一块宏根据自己的输入文件改变"in.txt","out.txt"，也可以只用其一。

```
#include <iostream>
#include <cstdio>

using namespace std;

#ifndef ONLINE_JUDGE
    freopen("in.txt","r",stdin);
    freopen("out.txt","w",stdout);
#endif

#ifndef ONLINE_JUDGE
    fclose(stdin);
    fclose(stdout);
```

#endif

用这种方法，cin/cout 和 scanf/printf 都可以转化为文件流

C 语言输入输出函数详解

C 语言中基本的输入输出函数有：

putchar ():把变量中的一个字符常量输出到显示器屏幕上;

getchar ();从键盘上输入一个字符常量,此常量就是该函数的值;

printf ();把键盘中的各类数据,加以格式控制输出到显示器屏幕上;

scanf ();从键盘上输入各类数据,并存放到程序变量中;

puts ():把数组变量中的一个字符串常量输出到显示器屏幕上;

gets ():从键盘上输入一个字符串常量并放到程序的数组中.

sscanf(); 从一个字符串中提取各类数据。

putchar() 和 getchar() 顾名思议就是从输入流中获取一个字符和输出一个字符，比较简单，不再多讲。

例子如下：

```
char c = getchar();
putchar(c);
```

格式化输入输出 scanf()和 printf()是最有用的，所以重点讲一下。

printf():

一般形式:

printf("格式控制".输出列表);

eg : printf("a=%d,b=%f,c=%c/n",a,b,c);

1;格式控制.

格式控制是用双引号括起来的字符串,也称"转换控制字符串",它包含以下两部分信息.

格式说明：由"%"和格式字符组成,如%d,%f,%c,他的作用是把输出数据转换为指定格式输出,格式的说明总是由"%"字符开始的.

普通字符：需要原样输出的字符,或者是一些有特殊含义的字符，如/n,/t。

2;输出列表

就是需要输出的一些数据,也可以是表达式,如果在函数中需要输出多个变量或表达式,则要用逗号隔开.

一些特殊字符的输出：

单引号，双引号，和反斜杠的输出在前面加转义字符”/”

如：”/”’，“/””，“//”

%的输出用两个连在一起的%%，即 printf(“%%”);

常用的格式说明如下:

格式字符

d 以十进制形式输出带符号整数(正数不输出符号)

o 以八进制形式输出无符号整数(不输出前缀 O)

x 以十六进制形式输出无符号整数(不输出前缀 OX)

u 以十进制形式输出无符号整数

f 以小数形式输出单精度实数

lf 以小数形式输出双精度实数

e 以指数形式输出单、双精度实数

g 以%f%e 中较短的输出宽度输出单、双精度实数

c 输出单个字符

s 输出字符串

这里强调一下：网上很多文章都说 f 和 lf 是一样的，即不管单精度，双精度浮点数，都可以用 f, 但我在 POJ 上做过测试，输出 Double 时用 f 确实也可以 ，但读入时，用 f 就报 WA，所以大家如果对 Double 进行读写的话，都用 lf 吧。说到 Double，再啰嗦一句，建议大家要用到浮点数时都用 Double，不要用 float，因为在很多情况下，float 精度不够会导致 WA。

特殊：

对 64 位整数的输入输出，在 POJ 上的 C++环境下(即 VC)，64 位整数是：

__int64 （注意 int 前面是两个下划线）

输入输出格式为"%I64d".

在 G++环境下(即 Dev C++) 64 位整数是

long long

输入输出格式为"%lld".

输出宽度

用十进制整数来表示输出的最少位数。 注意若实际位数多于定义的宽度，则按实际位数输出， 若实际位数少于定义的宽度则补以空格或 0。

精度

精度格式符以"."开头，后跟十进制整数。意义是：如果输出数字，则表示小数的位数；如果输出的是字符， 则表示输出字符的个数；若实际位数大于所定义的精度数，则截去超过的部分。

标志格式字符

- 结果左对齐，右边填空格

+ 输出符号(正号或负号)空格输出值为正时冠以空格，为负时冠以负号

例如：

double c=24212345.24232;

printf(“%020.4”); 表示输出精确到小数点后 4 位，输出占 20 位，若有空余的位补 0.

scanf：

scanf 的很多用法都是和 printf 对应的，故不再赘述。

说一下 scanf 一个特别好用的地方，就是可以滤去一些不想要的东西。

举例说明如下：

比如输入为日期 yyyy-mm-dd，就可以这样写：

int year,moth,day;

scanf(“%d-%d-%d”,&year,&moth,&day);

再比如：

scanf("%3d %*3d %2d",&m,&n);　输入 113 118 69 回车(系统将 113 赋予 m,将 69 赋予 n,因为*号表示跳过它相应的数据所以 118 不赋予任何变量)

puts()用的不多，且基本都能用 printf()代替，故不再多说。

gets()是从输入流中获取一行字符串放入字符数组中:

char in[100];

gets(in);

大家可能最容易出错的地方就是字符串的输入，所以强调一下：

能进行字符，字符串输入的有：

getchar(), scanf(“%c”); scanf(“%s”), gets()

其中 getchar() 和 scanf(“%c”)的功能是一样的。

需要注意的是，这两个函数读入的是输入流中当前位置的字符，

比如：

scanf(“%d”,&n);

c = getchar();

假设输入 67/ (假设“/”代表回车)，则第一个 scanf 读入一个整数 67 后，当前输入流的位置是 67 之后，即指向回车符，所以第二个 getchar()读入的就是一个回车符了，即 c = ‘/n’。

同样，gets()也是从当前位置读入一行字符串。

比如：

scanf(“%d”,&n);

gets(str);

此时读入字符数组中的字符串就是“/n” 了

所以通常在用 scanf 读入一个非字符串的类型之后，如果要读入字符，或字符数组，都用一个额外的 getchar()把回车符读掉，若后面跟的不止一个回车符，可能还有多余的空格的话，就用 gets()读掉。

和以上不同的是，scanf(“%s”) 读入的时候是会忽略掉空格，回车和制表符的。并且以空格，回车和制表符作为字符串结束的标志。

经常会有这样的题，输入第一行是一个整数，接下来每行的第一个是一个字符，用来表示某种操作，后面再跟一些数据，比如：

4

A 100 2

B 23

A 23 89

B 34

像这种输入就需要小心，读入字符时不要读成回车符。

为了防止意外，我一般是这样处理这类输入的：

```
char model[2];
Scanf(“%d”,&n);
for(…,…,…){
   scanf(“%s”,model);
   if(model[0] == ‘A’){
}
else{
}
}
```

sscanf():

sscanf()经常用来分解字符串，功能非常强大，但很多功能都需要正则表达式的知识，所以就介绍一下最简单的几种用法，大家如果想了解更多的话，自己去网上找吧。

1、

```
char str[100],str1[100],str2[100];
gets(str);
sscanf(str,”%s%s”,str1,str2);
```

将读入的一整行字符串按空格，制表符或回车符分割成两个字符串。

2、

取指定长度的字符串。如在下例中，取最大长度为 4 字节的字符串。

```
sscanf("123456 ", "%4s", str);
```

对于 C++的输入输出就不再详细的讲了，因为 cin,cout 的速度实在太慢，不推荐使用，我一般都是到万不得已时才用。比如当你要读入字符串到 string 对象中时，就只能用

cin 了，这时候还有一个常见的问题，就是如何将一整行字符串读入一个 string 中，这就要用到 getline 函数了。

用法为:

getline(cin, str);

第一个参数就是标准输入流 cin ，第二个参数是接收读入数据的 string 对象，本来还有第三个参数，是结束符的标志，但通常用它默认的就可以了，所以不用管。

注意区分这个 getline 和 cin.getline 的区别：

cin.getline 的用法如下：

char str[20];

cin.getline(str,20); 表示从读入的一行字符串中，取最多 20 各字符放入字符数组 str 中，注意此处的 str 是字符数组，而上面的 str 是 string 对象。

另外需要注意的是，千万不要把 cout 和 printf 混用，因为 cout 是带缓冲的而 printf 不带，所以会使得输出的数据顺序混乱。

下面是几个比较大的在线提交系统（Online Judge）里面有大量历年的竞赛题目，注册一个 ID，然后用自己熟悉的语言（一般有 Pascal/C/C++/Java）写好源代码提交即可，会实时返回信息告诉你是否正确。采用黑箱测试，系统里有一套标准的输入输出数据（对外保密，而且通常数据很多很怪），你的程序的输出和标准输出完全符合即可。常见的返回信息有 AC（Accepted，通过）WA（Wrong Answer，输出有错误）TLE（Time Limit Exceeded，超时）MLE（Memory Limit Exceeded，内存溢出）RE（Runtime Error，发生实时错误）等，只有 AC 了才算做对一题。这里只是一个简要介绍，请大家在做题时先看看各网站上的 FAQ，Enjoy it~~~

浙江大学 Online Judge（ZOJ）http://acm.zju.edu.cn

国内最早也是最有名气的 OJ，有很多高手在上面做题。特点是数据比较刁钻，经常会有你想不到的边界数据，很能考验思维的全面性，现在我主要在这个 OJ 上做题

北京大学 Online Judge（POJ）http://acm.pku.edu.cn/JudgeOnline/

建立较晚，但题目加得很快，现在题数和 ZOJ 不相上下，特点是举行在线比赛比较多，数据比 ZOJ 上的要弱，有时候同样的题同样的程序，在 ZOJ 上 WA，在 POJ 上就能 AC

同济大学 Online Judge http://acm.tongji.edu.cn/index.php

这个 OJ 题数上不能与上两个相比，推荐这个 OJ 的原因是它是中文的，这对很多对英文不太感冒的兄弟是个好消息吧。它也因此吸引了众多高中的 OIer，毕竟他们的英文还差一些呵呵，上面的题目也更偏向高中的信息学竞赛一些。

西班牙 Valladolid 大学 Online Judge（UVA）http://online-judge.uva.es/problemset/

世界上最大最有名的 OJ，题目巨多而且巨杂，数据也很刁钻，全世界的顶尖高手都在上面。据说如果你能在 UVA 上 AC 一千道题以上，就尽管向 IBM、微软什么的发简历吧，绝对不会让你失望的。

俄罗斯 Ural 立大学 Online Judge（URAL）http://acm.timus.ru/

也是一个老牌的 OJ，题目不多，但题题经典，我在高中的时候就在这上面做题的。

UsacoGate Online Judge（USACO）http://ace.delos.com/usacogate

全美计算机奥林匹克竞赛（USACO）的训练网站，特点是做完一关才能继续往下做，与前面的 OJ 不同的是测试数据可以看到，并且做对后可以看标准解答，所以如果大家刚开始的时候在上面那些 OJ 上总 WA 却找不到原因的话，可以试着来这里做做，看看测试数据一般是从什么地方阴你的。