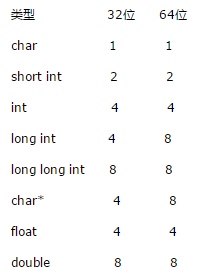
C:\Users\E470\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.pnghttp://acm.zjut.edu.cn/onlinejudge/problemset.php  
  
心舞 2019/3/18 8:05:18

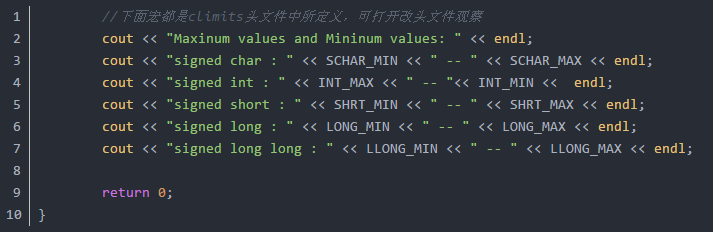
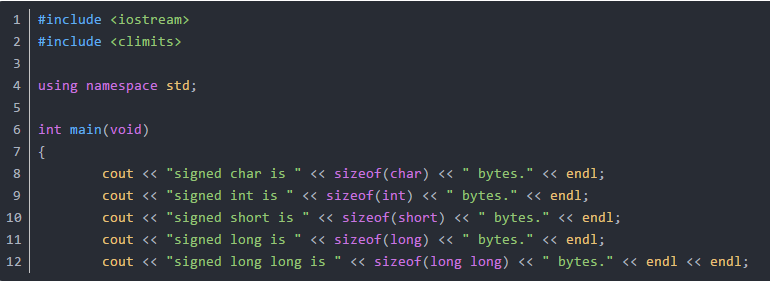
C++中可用的整型类型有五种,由小到大分别是char(也可以看成一种单字节的整型)、short、int、long、long long。它们使用不同数目的位来存储整型值。char在任何系统中都是单字节8位

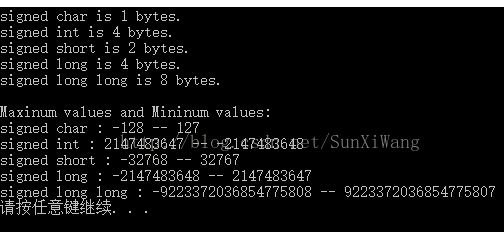
（1）short至少16位

（2）int 至少short一样长

（3）long 至少32位，且至少与int一样长

（4）long long 至少64位，且至少与long一样长





辗转相除法

**两**个整数的最大公约数是能够同时整除它们的最大的[正整数](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=70214187&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)。辗转相除法基于如下原理：两个整数的最大公约数等于其中较小的数和两数的差的最大公约数。例如，252和105的最大公约数是21（252 = 21 × 12；105 = 21 × 5）；因为252 − 105 = 21 × (12 − 5) = 147，所以147和105的最大公约数也是21。在这个过程中，较大的数缩小了，所以继续进行同样的计算可以不断缩小这两个数直至其中一个变成零。这时，所剩下的还没有变成零的数就是两数的最大公约数。由辗转相除法也可以推出，两数的最大公约数可以用两数的整数倍相加来表示，如21 = 5 × 105 + (−2) × 252。这个重要的结论叫做[裴蜀定理](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=73423552&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)。

最小公倍数=2数之积/最大公约数

//2法(1)辗转相除(2)从某一数递减验证公约数

#include <iostream>

using namespace std;

int main()

{ 1 int n=1,N,a,b,r;

cin>>N;

while(n<=N)

{ cin>>a>>b;

if(a<b){r=a;a=b;b=r;}

12 while(r=a%b){a=b;b=r;}

/\*int x=a;

while(!(a%x==0&&b%x==0))

{x--;}换12行,17行x替代b,删去1行r \*/

17 cout<<"case"<<n<<":"<<b<<endl;

n++;

}

return 0;

}

汉诺塔递归算法/实验4-7

**汉**诺塔是一个发源于印度的益智游戏，也叫河内塔。相传它源于印度神话中的大梵天创造的三个金刚柱，一根柱子上叠着上下从小到大64个黄金圆盘。大梵天命令婆罗门将这些圆盘按从小到大的顺序移动到另一根柱子上，其中大圆盘不能放在小圆盘上面。当这64个圆盘移动完的时候，世界就将毁灭。

汉诺塔问题源于印度神话

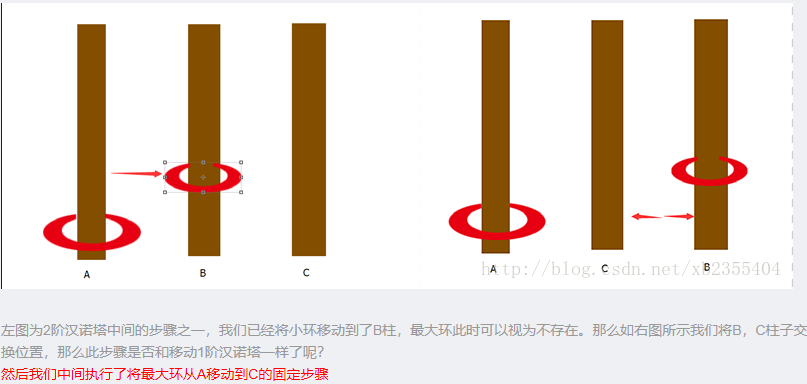
**那**么好多人会问64个圆盘移动到底会花多少时间？那么古代印度距离现在已经很远，这64个圆盘还没移动完么？我们来通过计算来看看要完成这个任务到底要多少时间？

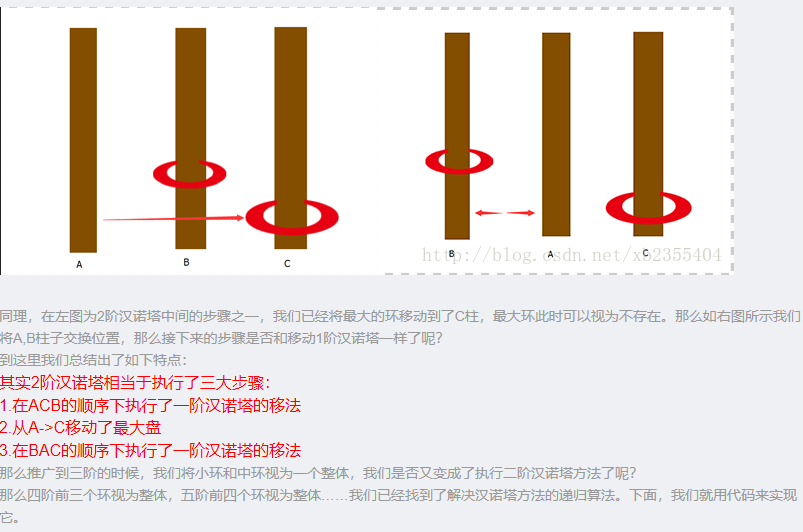
**我**们首先利用数学上的数列知识来看看F(n=1)=1,F(n=2)=3,F(n=3)=7,F(n=4)=15……F(n)=2F(n-1)+1;

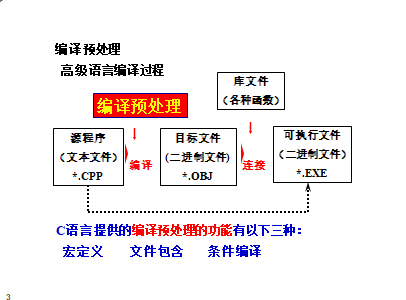
**我**们使用数学归纳法可以得出通项式：F(n)=2^n-1。当n为64时F(n=64)=18446744073709551615。

**我**们假设移动一次圆盘为一秒，那么一年为31536000秒

那么18446744073709551615/31536000约等于584942417355天，换算成年为5845.54亿年。

**目**前太阳寿命约为50亿年，太阳的完整寿命大约100亿年。所以我们整个人类文明都等不到移动完整圆盘的那一天





**ACM的江湖**

浙江大学（ZJU）：http://acm.zju.edu.cn/  
浙江工业大学（ZJUT）：http://acm.zjut.edu.cn/  
浙江师范大学（ZJNU）：http://acm.zjnu.cn/  
浙江工商（ZJGSU）：http://acm.zjgsu.edu.cn/  
宁波理工（NIT）：http://acm.nit.net.cn  
上海：  
华东师范大学（ECNU）：http://acm.cs.ecnu.edu.cn/  
华东理工大学（ECUST）：http://acm.ecust.edu.cn/  
同济大学（TJU）：http://acm.tongji.edu.cn/  
江苏：  
南京航空航天大学： http://acm.nuaa.edu.cn

**fabs()和abs()区别**

**相同点**

1.都是获取绝对值  
2.头文件都需调用#include<cmath>

不同点

函数名: abs  
功 能: 求整数的绝对值  
用 法:  
int abs(int i);  
返回的是|x|的值

函数名：fabs  
功能：求浮点数x的绝对值  
用法：  
float fabs(float j);  
double fabs(double j);  
返回的是|x|的值

sizeof():主要负责计算数组的长度

strlen():主要负责计算字符串的长度

1.sizeof():数组占用字节除以数组类型所占字节，结果为数组元素个数

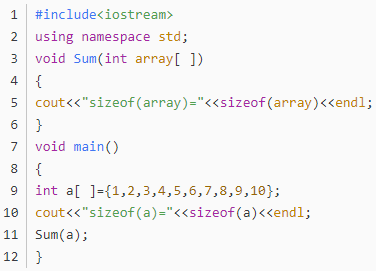
使用：sizeof（数组名)/sizeof（数组类型名）

2.strlen():strlen，求字符串有效长度

使用：strlen（字符数组名）//结果为字符数组有效字符长度，不包括末尾的'\0'

注意：

当数组作为函数参数传递时，数组名代表的是数组的首址，而非数组内容，故无法使用sizeof和strlen；所以，在传址时，应提供2个参数：1、数组名，代表数组首地址；2、数组元素个数，以便确定传递的次数。



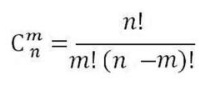
运算结果为：sizeof(a)=40   sizeof(array)=4

看到这样的结果，是不是感觉很奇怪？同样的数组，在不同的位置，输出却不一样。

分析：当数组作为函数参数传递时，数组名代表的是数组的首址，即指针，而非数组内容。如果传递整个数组，会导致栈溢出的。

所以在主函数中使用sizeof计算出的是准确的数组长度。而在调用函数中，由于传递的数组不再是数组本身，而是其地址，所以用sizeof计算出的，实际上是数组地址的长度，这时的sizeof(array)，实际上是sizeof(int)。

因此，要想计算实际的数组长度，还是在主函数中求好，要是在调用时想用，可以作为参数传递过去

**组合数**

算法：

1.写代码时不能用此公式直接算.语言中的数据类型(int)在表示范围上是有限的.更何况还有中间结果溢出的现象(18!溢出)，千万要小心

2.(1)不越界：a[i][j]=a[i-1][j-1]+a[i-1][j]注意边界要先赋值1

(2)越界：static a[][];没有先赋值

string s=“...”;

s.substr(i,n);//从下标i开始取n个字符