第五次上机题解

by：Ausar

----------祝大噶线代考试顺利噢----------

Mogg的简单数论这道题目，由于比较特别，所以单独写题解，pdf之后会给大家

A 淇淇的九九乘法表

时间限制：1000ms 内存限制：65536kb

通过率：1006/1015 `(99.11%)` 正确率：1006/2117 `(47.52%)`

> ## 题目描述
>
> 正在努力学习乘法的淇淇希望你能帮她打印一份九九乘法表，这样她就能快乐地学习乘法啦！
>
> ## 输入
>
> 无
>
> ## 输出
>
> 按照样例的格式输出九九乘法表。
>
> ## 输出样例

```
1 * 1 = 1
2 * 1 = 2 2 * 2 = 4
3 * 1 = 3 3 * 2 = 6 3 * 3 = 9
...
9 * 1 = 9 9 * 2 = 18 ... 9 * 9 = 81
```

提示

数字和符号之间都有空格

——Ausar

这题挺水的了，就是循环加输出，别忘了空格就好

```
#include<stdio.h>
//By:xsy
int main()
{
    int i,j;
    for (i = 1; i <= 9; ++i)
        for (j = 1; j <= i; ++j)
            printf("%d * %d = %d%c", i, j, i * j, " \n"[j == i]);
    return 0;
}
```

B 再遇士谔数

时间限制：1000ms 内存限制：65536kb

通过率：954/980 `(97.35%)` 正确率：954/1406 `(67.85%)`

题目描述

士谔书院18级的编号是1873，所以以1,8,7,3结尾的十进制正整数，都是士谔数。 现在要你输出[L,R][L,R]中所有的士谔数

输入

共一行，2个正整数L,R。

输出

若干行，一行一个整数，表示一个士谔数，从小到大输出。

输入样例

```
1 10
```

输出样例

```
1
3
7
8
```

limit

L,R∈[1,100000]

这题是猪脚头子为了各位开心，专门放的一道弱化版的水题给大家玩耍。由于弱化过数据，所以直接for循环就能搞定。

```
#include <stdio.h>
//By:pmxm
int main(){
    int L,R;
    int i;
    scanf("%d%d",&L,&R);
    for (i=L;i<=R;i++)

        if ( ( i%10 == 1) + ( i%10 == 8)  + (i%10 == 7) + (i%10 == 3) )
            printf("%d\n",i);
    return 0;
}
```

C 考试得分

时间限制：1000ms 内存限制：65536kb

通过率：970/977 `(99.28%)` 正确率：970/2235 `(43.40%)`

题目描述

今有一次考试，共10题，每题只有两个选项0/1，现有4份试卷，已知其选项和前3份试卷的得分，试求第4份的得分。

number	1	2	3	4	5	6	7	8	9	10	score
std1	0	0	1	0	1	0	0	1	0	0	70
std2	0	1	1	1	0	1	0	1	1	1	50
std3	0	1	1	1	0	0	0	1	0	1	30
std4	0	0	1	1	1	0	0	1	1	1	?

> 解是唯一的。
>
> ## 输入
>
> 无
>
> ## 输出
>
> 一个整数，第4份的得分
>
> ## 提示
>
> 逃跑很有效
>
> ——艾克臣
>
> ---
>
> 每题的答案可能是1或者0，每题占10分
>
> 也就是说，前三个学生分别答对了7题、5题和3题
>
> 所有答案的可能情况只有2^10=1024种，所以完全可以用循环暴力枚举
>
> 对于每一位只有0和1的数字串枚举问题，可以尝试用二进制的方法来解决
>
> ——Ausar

这题关键是如何遍历所有的答案可能，然后把答案的可能性与学生的解答对比

如果这个答案与前三个学生回答正确数吻合，那么这个就是正确答案。与第四位学生的答题对比一下就能得出分数了。

这题用**二进制**是最好的选择，一共有10道题，那么所有的答案可能在二进制里面表示就是：

0000000000到**1111111111**

在**10进制**中，就是**0**到**1023**，其中，1023可以很简单地表示成

```
(1<<10)-1
```

那么我们只用一个简单的

```
for(i=0;i<(1<<10);i)
```

就可以完成对所有可能答案的遍历。

而如何完成对答案和学生试卷的比较呢?

以第一位同学为例，他分别在3、5、8这几题回答了1，其他题都是0

那么我们可以在二进制中这么表示：

0010100100

由于在二进制存储的时候，右边是低位，左边是高位，所以我们不妨反转一下，得到：

0010010100

而如何快速得到这个数字呢，我们可以用这个式子：

```
a=(1<<2)+(1<<4)+(1<<7);
```

然后，我们可以按位比较这个同学与答案是否相同，以此算出他得了几分。

这里我们可以用for循环暴力比较。（利用右移，按位与，按位异或来得到）

但是更巧妙的方法是：

直接把答题与标准答案按位异或

这样，如果有某一位的答题与标准答案不同，那一位就会被置为1

例如：

如果学生答案是：0010010100

标准答案是： 0011011000

异或结果： 000**1**00**11**00

可以看出，结果为1的题目就是**做错**的题目。

c语言里面自带了一个统计二进制中，1的个数的函数，叫做

__builtin_popcount（）

所以我们得到：

总题数 - __builtin_popcount（异或结果）= 正确题数

只需要看看是否与实际情况匹配就好

代码如下：

```
#include<stdio.h>
//by:cmx
int a=(1<<2)+(1<<4)+(1<<7);
int b=(1<<1)+(1<<2)+(1<<3)+(1<<5)+(1<<7)+(1<<8)+(1<<9);
int c=(1<<1)+(1<<2)+(1<<3)+(1<<7)+(1<<9);
int d=(1<<2)+(1<<3)+(1<<4)+(1<<7)+(1<<8)+(1<<9);
//4个学生分别的答题情况
int main()
{
    for(int i=0;i<(1<<10);i++)//遍历所有的可能标准答案
        if(10-__builtin_popcount(i^a)==7)
```

```
            if(10-__builtin_popcount(i^b)==5)
                if(10-__builtin_popcount(i^c)==3)//如果三位的答题得分均吻合
                {
                    printf("%d",100-10*__builtin_popcount(i^d));
                    return 0;
                }
    return 0;
}
```

D 格式化倒序输出

时间限制：1000ms 内存限制：65536kb

通过率：893/945 `(94.50%)` 正确率：893/2301 `(38.81%)`

题目描述

将输入的数按照要求格式倒序输出。

格式要求：每个数所在的字段宽度为10，有前导0和正负符号（正数前面有正号）。

输入

第一行，一个整数n；

接下来n行，每行5个整数，a1，a2，a3，a4，a5。（-10000<ai<10000）

输出

n行，每行5个整数，a5，a4，a3，a2，a1，两两之间以空格间隔。

输入样例

```
2
1 2 3 4 5
-10 -5 0 5 10
```

输出样例

```
+000000005 +000000004 +000000003 +000000002 +000000001
+000000010 +000000005 +000000000 -000000005 -000000010
```

Hint

使用printf()函数的格式控制符号即可完成整数的格式化；

具体请查看printf()函数的格式标志表（书上或百度）。

> 尹宝林老师写的《c程序设计导引》的p95上有详细介绍
>
> ——Ausar

这题就很简单了，因为只有5个数，所以不会数组也无妨

按照书上给的方法，控制printf按照制定格式倒序输出即可

此处给出的代码运用到了数组，但是不用数组也完全可以做。只是代码稍微长了一点而已。

```
#include<stdio.h>
//By:lzz
int main()
{
    int i, a[5], n;
    scanf("%d", &n);
    while (n--){
        for (i=0; i<5; i++) scanf("%d", &a[i]);
        for (i=4; i>0; i--) printf("%+010d ", a[i]);//带符号，有前导0，10位整数
        printf("%+010d\n", a[0]);
    }
}
```

PS：好多助教其实也忘了printf可以输出数字的符号，hhh

E Ausar的非格式化倒序输出

时间限制：1000ms 内存限制：65536kb

通过率：619/716 `(86.45%)` 正确率：619/1393 `(44.44%)`

> ## 题目描述
>
> 有一个猪脚出了要格式化到序输出的题目
>
> 奥萨不喜欢这种条条框框，他喜欢自由的感觉
>
> 于是他出了一道很自由的到序输出问题
>
> 他会给你若干个在int范围内的数，请你将他们倒序输出
>
> （正常输出就好，别加什么奇奇怪怪的格式）
>
> 不过由于这题比较自由，所以每组数据的数字个数是不定的
>
> **请大家尽可能用递归完成**
>
> ## 输入
>
> 多行输入，每行一个数字
>
> ## 输出
>
> 多行输出，每行一个数字，和输入顺序相反

输入样例

```
123
223
4399
6666
1212121
```

输出样例

```
1212121
6666
4399
223
123
```

提示

我们可以这样设计一个递归函数

```
1.尝试读取一个数字，读取失败直接返回
2.如果读取成功，那么调用自身
3.调用自身结束之后，输出读取到的数字
```

——Ausar

如果实在怂，用循环也可以做，数据非常弱

——ArcheyChen

这题数据非常弱，最多也就10来个数字，用数组是轻轻松松就可以过的。

但是我出这题，是为了让大家知道递归还能有奇妙的用法。

递归函数如何设计我已经写在提示里面了。我在这就直接贴出代码，希望各位能领悟一下里面的含义。

```c
#include<stdio.h>
//By:ljf
void Rua()
{
    int x;
    if(~scanf("%d",&x))//和多组数据读入类似，尝试读入一个数据
    {
        Rua();
        printf("%d\n",x);//递归结束之后再输出
    }
    return;
```

```
}

int main()
{
    Rua();//RUA!
    return 0;
}
```

鉴于有些同学并不知道这个程序到底做了些什么，*“莫名其妙就AC了”*，所以我再解释一下。

每一层调用函数，函数都会读取一个值，然后记录下来，然后再次调用下一层函数。

最后一层函数读入失败之后，开始返回。

各层函数与调用顺序相反地返回，每层函数返回的时候都把自己读到的值给输出出来。

所以就是输出顺序与读入顺序正好相反了。

PS：对于多组读入，windows系统需要用ctrl+z来模拟EOF，而苹果系统需要ctrl+q，然后ctrl+d才能模拟

F 大水题

时间限制：1000ms 内存限制：65536kb

通过率：678/844 `(80.33%)` 正确率：678/2345 `(28.91%)`

> ## 题目描述
>
> 为了补偿被WA，TLE，REG……等折磨得痛苦不堪的同学们，Max决定出一道大水题，并且非常确信它的通过率是最高的良心题。
>
> 现在请你来帮助Max来实现这个Flag（或者打脸?）。
>
> 输入多组二元组<xi,yi>，求所有满足 **1** 1<=xi<=2000，1<=yi<=2000，**2** xi和yi不能互相整除，**3** xi，yi都不等于1837 的所有二元组的xi之和与yi之和。
>
> ## 输入描述
>
> 多行，每行一个二元组，xi和yi。
>
> 保证数据不超过100000行。
>
> ## 输出描述
>
> 一行两个整数 分别表示符合要求的xi的和与yi的和
>
> ## 输入样例

```
2 1
5 3
1837 7
```

输出样例

```
5 3
```

提示

不能互相整除的意思是，xi除以yi的余数，以及yi除以xi的余数都不等于0

——Ausar

这体是真滴水，主要就是考一个多组数据的读入，然后一个if判断一下就行了，直接贴代码

```
#include<stdio.h>
//By:cmx
int x,y,a,b;

int main()
{
    while(scanf("%d%d",&x,&y)==2)
        if(x>=1&&x<=2000&&y>=1&&y<=2000&&x!=1837&&y!=1837&&x%y&&y%x)
            a+=x,b+=y;
    printf("%d %d",a,b);
    return 0;
}
```

在c语言里，一个表达式的值只有在**等于0**的情况下才会被判断为**假**，**别的时候**都判断为**真**

所以里面的 **x%y** 实际上是等价于**x%y!=0**

G lx的简单题

时间限制：1000ms 内存限制：65536kb

通过率：48/297 `(16.16%)` 正确率：48/995 `(4.82%)`

题目描述

lx不想出题面。于是他出了一道简单题。

给定nn个整数，求相差最小的两个数。

输入

两行。第一行一个正整数n(2<=n<=100000)，表示有n个整数。第二行n个整数a[i]。

保证a[i]在int范围内。

保证对∀i≠j，有a[i]≠a[j]。(1<=i,j<=n)

输出

一行。输出相差最小的两个数，小的数在前，大的数在后。

如果相差最小的两个数不是唯一的，那么输出其中最小的两个数。

输入样例

```
5
1 2 3 4 5
```

输出样例

```
1 2
```

Hint

使用Selection Sort/Bubble Sort/Insertion Sort, etc.可以拿到大部分分数。

想拿满的同学需要使用stdlib.h里的qsort函数或者使用其他效率更高的算法。^_^

对于qsort的用法，可以参照这个连接

但是里面的cmp函数最好不要写成直接相减的模式，否则可能会产生溢出，最好是判断两个数字大小再选择返回-1或1或0

——Ausar

这题主要是要用数组加排序（对的，就是超纲了，你们貌似还没学到数组）

最朴素的冒泡排序是会超时的，所以必须用快速排序，堆排序之类的比较快的排序方法

c自带了一个快速排序函数**qsort()** ,这个函数的具体用法我已经贴一个博客的连接

但是这个博客里面的cmp函数写得不太好，我也在提示中说明了这点。请大家参考我贴的标程。

排序好了之后，只需要扫一遍数组，然后记录两个相邻数字的最小值，还有这两个数字是什么，最后输出即可

```
#include <stdio.h>
#include <stdlib.h>
//By:lx
int cmp(const void *a, const void *b)//这个比较函数的格式是qsort规定的
{
    return *(int *)a > *(int *)b ? 1 : -1;//强制转换成int指针类型，然后解指针
}

int main()
```

```
{
    int n, a[100005];
    int i, j, x, y;
    scanf("%d", &n);
    for (i = 0; i < n; i++) {
        scanf("%d", a + i);
    }
    qsort(a, n, sizeof(a[0]), cmp);
    //待排序的元素首地址，待排序的元素个数，待排序的元素大小，比较函数

    long long min = a[1] - a[0];//初始化min为前两个数的差
    x = a[0]; y = a[1];//初始化记录数据为前两个元素
    for (i = 1; i < n-1; i++) {
        long long t = (long long)a[i+1] - a[i];
        if (min > t) {//出现了更小的差，则更新min,x还有y
            min = t; x = a[i]; y = a[i+1];
        }
    }
    printf("%d %d\n", x, y);
    return 0;
}
```

H 酸奶吃羊排

时间限制：1000ms 内存限制：65536kb

通过率：159/317 (50.16%) 正确率：159/882 (18.03%)

题目描述

酸奶非常喜欢玫瑰红这家店的羊排，曾经十分钟解决过一份羊排，算一份羊排十根的话，酸奶一分钟就能吃掉一根羊排。这天酸奶和小伙伴又来了这家店排了两个小时的队之后终于吃上了，一共点了n（<=1000）根羊排，小伙伴食量不及酸奶，x分钟才能吃掉一根，而且两个人每次都会选择当前肉量最多的羊排开吃，请问小伙伴能吃到多少肉呢，以及两人吃了多少分钟？（当两人同时选择羊排时，酸奶会让小伙伴先选）

输入

第一个行两个正整数n，x

接下来一行n个正整数，表示每根羊排的肉量（<=1000）

输出

一行两个数，依次表示**小伙伴**吃到的总肉量和吃的时间，用空格隔开

输入样例

```
7 3
1 2 3 4 5 6 7
```

输出样例

```
10 5
```

Hint

```
第一分钟小伙伴吃掉了7，酸奶吃了6
第二分钟酸奶吃了5
第三分钟酸奶吃了4
第四分钟小伙伴吃了3，酸奶吃了2
第五分钟酸奶吃了1，吃完了
```

这里的数字指的是每**根**羊排的肉量，别被题面里面的“根”和“份”给搞迷糊了

这题排序好之后就很容易做了

——Ausar

这题的意思其实就是，酸奶和他的小伙伴会从大到小吃。然后小伙伴每x分钟吃一根。

那我们就可以先排序，然后从大到小遍历。每x分钟让小伙伴“吃”一根。最后把结果输出即可。

```
#include <stdio.h>
//By:ljh
int n,x,a[2000],ans;

int main()
{

    scanf("%d%d",&n,&x);
    for(int i=1;i<=n;i++)
        scanf("%d",&a[i]);
    for(int i=1;i<=n;i++)
        for(int j=i+1;j<=n;j++)
            if(a[i]<a[j])//这种是最朴素的冒泡排序，效率很低，但是容易写。应对这题的小数据足够了。
            {
                a[i]=a[i]^a[j];
                a[j]=a[i]^a[j];
                a[i]=a[i]^a[j];
            }//这个地方是将a[i]和a[j]交换。在之前的题目“Ausar的Switch”里面有介绍过这个方法
    int cnt=0;
    for(int i=1;i<=n;i++)
    {
        if(cnt% x==0)//小伙伴开始吃了
        {
            ans+=a[i];
            i++;//吃了一根，所以i额外+1
        }
        cnt++;
    }
```

```
    printf("%d %d",ans,cnt);
    return 0;
}
```

I HugeGun学姐的魔法

时间限制：2000ms 内存限制：130000kb

通过率：44/313 (14.06%) 正确率：44/1233 (3.57%)

题目描述

HugeGun学姐被魔鬼MountVoom追杀了，于是她逃进了平面直角坐标系，没想到MountVoom也会这招。

于是HugeGun使用了一手魔法把直角坐标系改成了极坐标系。

MountVoom心态爆炸，给了你nn个点的直角坐标，问你它们对应的极坐标是多少。

输入

第一行，一个整数，为点数n(1≤n≤1000000)n(1≤n≤1000000)

接下来n行，每行2个整数a,b(−500000≤a,b≤500000)a,b(−500000≤a,b≤500000)表示一个点的直角坐标。

输出

对于每个点，输出一行，两个77位小数A,B表示对应的极坐标。其中A为极径，B为极角，B应满足0≤B<2π

对于特殊点(0,0)，应输出0.0000000 0.0000000

输入样例

```
2
0 0
0 1
```

输出样例

```
0.0000000 0.0000000
1.0000000 1.5707963
```

提示

这是一道基础几何题，方法有很多，math.h里有很多三角函数可以用哦

这题就是考你极坐标还记不记得是个啥东西。然后会不会用math.h函数库。直接贴代码吧

```
#include<stdio.h>
```

```
#include<math.h>
//By:cmx
double a,b,d,s;
int n;

int main()
{
    scanf("%d",&n);
    while(n--)
    {
        scanf("%lf%lf",&a,&b);
        d=sqrtl(a*a+b*b);
        if(a==0&&b==0)
            s=0;
        else
            s=acosl(a/d);
        if(b<0||b==0&&a<0)
            s=2*acosl(-1)-s;
        printf("%.7lf %.7lf\n",d,s);
    }
    return 0;
}
```

J login走迷宫

时间限制：1000ms 内存限制：8192kb

通过率：51/120 `(42.50%)` 正确率：51/240 `(21.25%)`

题目描述

login玩游戏的时候遇到了一个恶意的迷宫，路线是呈螺旋形顺时针向中心靠近的。假设地图是一个n*m的矩形方格图（n行m列），从左上角出发，移动到相邻一格需要一步，他想知道走到第p行q列需要几步。

（你可以理解为墙是没有厚度的）

以下是一个4*5的地图上，到每一格的步数所形成的矩阵

```
1  2  3  4  5
14 15 16 17 6
13 20 19 18 7
12 11 10 9  8
```

其中，数字表示步数。如：第2行3列需要16步

输入描述

一行四个数，分别为n,m,p,q

输出描述

一个数，表示需要的步数

样例输入

```
4 5 2 3
```

样例输出

```
16
```

数据范围和约束

1≤n,m≤5000

1≤p≤n

1≤q≤m

HINT

把步数填到矩阵中就是一个螺旋矩阵哦。

该题不需要数组，因此内存限制很小，如果用数组存下步数会MLE哦。

其他助教友情提示

这题可以手动推出公式，加上一些判断就可以直接输出结果

或者是用变量x，y记录现在的坐标，模拟走一遍

——Ausar

这道题目，比较厉害的同学可以试着推出公式。但其实模拟是个比较好的方法。

虽然我们开数组会炸内存，但是我们可以完全不用数组。只用x和y变量记录坐标，模拟着走一走就好了

在各位助教写的程序中，我挑了一份比较直观的供大家参考。

```
#include <stdio.h>
int main()
{
    int n, m, p, q;
    scanf("%d%d%d%d", &n, &m, &p, &q);
    int x = 1, y = 1, direct = 0, cir = 0, ans = 0; // 0: right 1: down 2: left 3: up
    while (y != p || x != q) {
        ans++;
        switch(direct) {
            case 0: x++;
```

```
                if (x+1 == m-cir+1) {
                    direct = 1;
                }
                break;
            case 1: y++;
                if (y+1 == n-cir+1) {
                    direct = 2;
                }
                break;
            case 2: x--;
                if (x-1 == cir) {
                    direct = 3;
                }
                break;
            case 3: y--;
                if (y-1 == cir+1) {
                    direct = 0;
                    cir++;
                }
                break;
        }
    }
    printf("%d\n", ans+1);
    return 0;
}
```

K Ackermann函数

时间限制：1000ms 内存限制：65536kb

通过率：947/969 `(97.73%)` 正确率：947/1198 `(79.05%)`

题目描述

login正在学数学。

他要尝试计算阿克曼函数，然而计算量太大了，他口算不出来，因此，他拜托你来帮助他计算。

阿克曼函数的定义如下：

$$A(m,n) = \begin{cases} n+1 & m = 0 \\ A(m-1,1) & m > 0 \wedge n = 0 \\ A(m-1,A(m,n-1)) & m > 0 \wedge n > 0 \end{cases}$$

输入描述

一行两个数，m和n。

输出描述

一行一个数，A(m,n)

样例输入

```
2 2
```

样例输出

```
7
```

数据范围和约束

0≤m≤3

0≤n≤10

提示

你会递归了吗?

尹老师的书p100

希望别的助教别打死我

——Ausar

这题妥妥的是书上的原题。我已经在提示里面给了你们书本上的页数了。希望大家能通过这题，体会一下递归到底是个怎么回事。

```
#include <stdio.h>
//By:lyt
int ack(int, int);
//这个东西叫做函数原型，告诉别的函数，我的函数写在后面。函数原型只需要指明参数类型，不一定要写上参数名称

int main()
{
    int m,n;
    scanf("%d%d",&m,&n);
    printf("%d",ack(m,n));
}

int ack(int m,int n)
{
    if(m==0)
        return n+1;
    else if(n==0)
        return ack(m-1,1);
```

```
    else
        return ack(m-1,ack(m,n-1));
}
```

L 悠唯的复读序列

时间限制：1000ms 内存限制：65536kb

通过率：335/513 `(65.30%)` 正确率：335/1577 `(21.24%)`

题目描述

悠唯很喜欢水群，而众所周知的是，群里有很多复读机。

群里最开始的话题由3个群友一人说一句话开始。之后新加入的都是复读机。第ii个群友由于网络延迟的缘故来不及复读第i−1个群友的发言，因此ta会复读第i−2和i−3个群友的发言，所以他说的话是二者相加那么多。

现在，悠唯想知道对于n个编号为i的群友，他们分别说了几句话。由于这个数字过于庞大，悠唯希望将结果对100000007取模后输出。

输入

第一个数为数据组数n(n<=100) 接下来n行，每行1个整数i(i<=500000)

输出

对于每组数据，输出一行，表示F(i)对100000007取模后的结果

输入样例

```
6
1
2
3
4
5
6
```

输出样例

```
1
1
1
2
2
3
```

> ## HINT
>
> 斐波那契数列的简单变化。 **注意时间限制**，请尽量减少代码运算量。
>
> ————
>
> 别用递归，会gg的
>
> ——Ausar

其实这题就是稍微修改过的斐波那契数列。只不过递推公式变成F(n)=F(n-2)+F(n-3);

这题数据量比较大，如果写递归的话会超时，希望大家用循环递推

（据说尾递归也能过，具体什么叫做尾递归，学有余力的同学可以自己百度）

为了照顾没自学过数组的同学，这里贴的代码是没有用数组的

```
#include<stdio.h>
By:lyt
int MOD=100000007;

int main()
{
    int T;
    scanf("%d",&T);
    while(T--)
    {
        int a=1,b=1,c=1,temp;
        int n;
        scanf("%d",&n);
        for(int i=4;i<=n;i++)
        {
            temp=a+b;
            if(temp>=MOD)
                temp-=MOD;//既然是加法，就没必要用%了，可以直接用减法，减法运算还比取模运算快
            a=b;
            b=c;
            c=temp;
        }
        printf("%d\n",c);
    }
}
```

M ArcheyChen的汉诺塔

时间限制：1000ms 内存限制：65536kb

通过率：816/871 `(93.69%)` 正确率：816/1306 `(62.48%)`

> ## 题目描述

艾克臣摸鱼摸太久了，他感觉再不出点题目就要被炒鱿鱼。正巧，他手头有一个汉诺塔，于是他决定让你写一个帮他解决汉诺塔问题的程序。

汉诺塔，相信大家小时候都玩过。它由三根柱子和若干个从大到小的圆盘组成。

汉诺塔的规则是，在任何时候，大盘都不能压在比他小的盘上面，每次移动只能取一根柱子最上面的一个盘移动到另一根柱子上。

现在，有n个圆盘组成的汉诺塔放在第一根柱子上，请帮ArcheyChen输出最快把所有盘都移动到第三根柱子上的步骤

输入

第一行，一个正整数n（n<13)

输出

若干行，第k行代表第k次移动的步骤，输出格式如下

```
  "%c --> %c"
```

三根柱子分别用大写字母ABC来表示

输入样例

```
3
```

输出样例

```
A --> C
A --> B
C --> B
A --> C
B --> A
B --> C
A --> C
```

Hint

我们可以在一次移动中，把柱子分为“from” “to”和“tmp”

```
如果我们这次只用移动一片盘子，那么直接从from移动到to就行了
```

如果我们要移动一摞的盘子（共n片），我们可以

> ```
> 1.先将n-1片从from移动到tmp
> 2.然后将剩下的那1片从from移动到to
> 3.再将之前的n-1片从tmp移动到to
> ```
>
> 其实尹老师的书p101也写有详细的代码
>
> ——Ausar

这题真的差不多就是原题了。

汉诺塔差不多是递归的精华题目了。

如果只有一层，那么直接移动就好。

如果有多层，我分解成两部分。一部分有n-1层，另一部分有1层。

把n-1层先移动到tmp（这部分是递归），然后移动一层（这部分可以递归也可以不递归）到目的地，最后把n-1层从tmp移动到目的地就行了。

```
#include<stdio.h>
//By:ljf
char t[]={"XABC"},in[]={"in1.txt"},out[]={"out1.txt"};
long long rua(int layer,int from,int to)//顺手写了统计移动次数。在此题实际上用不上。
{
    if(layer==1)
    {
        printf("%c --> %c\n",t[from],t[to]);
        return 1;
    }
    long long ans=0;
    ans+=rua(layer-1,from,6-from-to);//三个塔的编号，1，2，3加起来刚好为6，所以可以这么算出tmp
    //先把n-1层挪到tmp
    ans+=rua(1,from,to);
    //把剩下的那1层直接挪到to
    ans+=rua(layer-1,6-from-to,to);
    //把之前的n-1层从tmp挪到to
    return ans;
}
int main()
{
    int n;
    scanf("%d",&n);
    rua(n,1,3);

}
```

O lx放球

时间限制：1000ms 内存限制：65536kb

通过率：209/341 `(61.29%)` 正确率：209/645 `(32.40%)`

题目描述

lx最近在研究组合数。他手上有n个有标号的球，标号依次为1，2，…，n。将这n个球放入r个相同的盒子里，不允许有空盒，其不同放置方法的总数记为F(n,r)。 现在给定n和r，lx对F(n,r)很感兴趣，但是他不会求。所以他想请你求出F(n,r)。

输入

一行。两个正整数n和r。其中0<r<=n<=20

输出

一行。一个正整数，表示F(n,r)

输入样例

```
4 2
```

输出样例

```
7
```

样例解释

F(4,2)=7

这7种不同的放置方法依次为 {(1) , (234)} , {(2) , (134)} , {(3) , (124)} , {(4) , (123)} , {(12) , (34)} , {(13) , (24)} , {(14) , (23)} 。

Hint

如果你解决了r<=3的情况，就可以拿到70分。^_^

——

请推导F(n,r)的递推公式。

什么是递推公式？就是F(n,r)与它前一项或几项的关系可以用一个式子来表示，那么这个式子就是递推公式。

通项也是可以推出来的，但不建议去尝试。

这题需要一定的思考。

把n个球放到r个盒里面，不允许有空盒。

那么：

1.如果球不够盒子多，或者不存在盒子，那么此时方案数为0

2.如果只有一个盒子，或者是球和盒子的数量正好相等，那么只有一种方案

3.其他的情况。我们考虑第n个球所在盒子有一个球和有多个球的情况。

a)如果第n个球的盒子里面只有这一个球，那么这种情况的数量等于F(n-1,r-1)

b)如果第n个球的盒子里面不只有一个球，那么我们可以先吧剩下的n-1个球放在r个盒子里面，然后把第n个球随便扔到一个盒子里面，共有F(n-1,r)*r种情况

代码如下：

```
#include <stdio.h>
//By:lx
long long f(int n, int r)
{
    if (n < r || r == 0)
        return 0;
    if (r == 1 || r == n)
        return 1;
    return f(n-1, r-1) + r*f(n-1, r);
}

int main()
{
    int n, r;
    scanf("%d%d", &n, &r);
    printf("%lld\n", f(n, r));
    return 0;
}
```