

# 内容提纲

- ❖ 认识递推
  - 递推的概念
  - 生活中的递推示例
- ❖ 递推关系的建立与求解方法
  - 递推关系的建立
  - 递推的求解方法
- ❖ 算法示例
  - 模拟示例
  - 练习系统
  - 蓝桥真题

*You are here!*

你在这儿！

# 认识递推

- ❖ 递推算法是从问题的规模（项数）出发，找到大规模问题与小规模问题之间的关系（或前后项之间的关联），然后根据他们之间的联系逐步求解。
- ❖ 例如：
  - 数列0, 3, 6, 12, 15, ...
  - $$\begin{cases} a_n = a_{n-1} + 3 & (n > 1) & \text{(递推公式)} \\ a_1 = 0 & (n = 1) & \text{(边界条件)} \end{cases}$$

# 认识递推

- ❖ 这种在规定的初始条件下，找出后项对前项的依赖关系的操作，称为**递推**。表示某项和它前面若干项的关系式就叫**递推公式**。初始条件称为**边界条件**。
- ❖ 递推法分为顺推和倒推两种，**顺推**适用于已知起始条件（小规模问题的解），然后根据关联逐步推算，而**倒推**则相反。

# 认识递推

- ❖ 在实际问题中类似的很多，处理这类问题的理想方法是用归纳法求出通项公式。上例中的通项公式为 $a_n = (n-1) \cdot 3$  ( $n \geq 1$ )。
- ❖ 但是大多情况下，要想得到通项公式是比较困难的，而通过已知条件归纳出一个递推关系则相对容易。
- ❖ 这时我们可以采用递推技术，**避开求通项公式**的麻烦。

# 递推与递归的关系

- ❖ 可用递推求解的题目一般有以下两个特点：
  - (1) 问题可以划分成多个状态；
  - (2) 除初始状态外，其它各个状态都可以用固定的递推关系式来表示。
- ❖ 在我们实际解题中，题目不会直接给出递推关系式，而是需要通过分析各种状态，找出递推关系式。

# 递推与递归的关系

---

- ❖ 递归与递推既有相似点又有不同点
- ❖ 相同点：数据元素之间的关系可以用抽象的、严格的公式表示出来，且不论是递推关系式还是递归公式，都具有边界条件。



# 递推与递归的关系

## ❖ 不同点：

- **递推是从边界条件出发，通过递推关系式求 $f(n)$ 的值。**从边界到目标解的过程很清楚直观；
- **而递归是从函数自身出发来达到边界条件**，在通往边界的递归过程中，系统需要用堆栈把每次调用的现场(中间结果)保存下来，供递归返回时使用(出栈、恢复现场)。
- **当然，递归与递推也可以相互转换**，具体应用时主要看哪种解法更加简便高效。

# 生活中的递推示例

- ❖ 超简单的例子：有5人坐在一起，当问第5个人多少岁，他说比第4个人大2岁，问第4个人多少岁，他说比第3个人大2岁，依此下去，问第一个人多少岁，他说他10岁，最后求第5个人多少岁？
- ❖ 如果所坐的不是5人而是 $n$ 人，写出第 $n$ 个人的年龄表达式。



# 写出递推公式或通项公式

## ❖ 递推公式

$$f(n) = \begin{cases} 10(n = 1) \\ f(n - 1) + 2(n \geq 2) \end{cases}$$

## ❖ 通项公式

通项公式:

$$F(n) = 10 + (n - 1) * 2$$

# 内容提纲

- ❖ 认识递推
  - 递推的概念
  - 生活中的递推示例
- ❖ 递推关系的建立与求解方法
  - 递推关系的建立
  - 递推的求解方法
- ❖ 算法示例
  - 模拟示例
  - 练习系统
  - 蓝桥真题

*You are here!*  
你在这儿！

# 递推关系的建立

- ❖ 解决递推问题有三个重点：
  - **如何建立正确的递推关系 --- 核心**
  - 递推关系有何性质
  - 递推关系式如何求解
- ❖ 按照推导问题的方向，分为顺推法和逆推法
  - 顺推法：从已知条件出发，逐步推出要解决的问题。
  - 逆推法：从问题出发，逐步推到已知条件。

# 递推的求解方法

## ❖ 递推算法的求解方法

- 数组：从已知条件入手，填充数组。
- 递归：从问题入手，递归调用函数。
- 用递推式的通项表达式：
  - 迭加法
  - 待定系数法
  - **特征方程法**
  - **生成函数法**

# 内容提纲

- ❖ 认识递推
  - 递推的概念
  - 生活中的递推示例
- ❖ 递推关系的建立与求解方法
  - 递推关系的建立
  - 递推的求解方法
- ❖ 算法示例
  - **模拟示例**
  - 练习系统
  - 蓝桥真题

*You are here!*  
你在这儿！

# 递推的求解方法

## ❖ 递推算法的求解方法

- **数组：从已知条件入手，填充数组。**
- 递归：从问题入手，递归调用函数。
- 用递推式的通项表达式：
  - 迭加法
  - 待定系数法
  - **特征方程法**
  - **生成函数法**



## 1

## 一只蜜蜂

水

- ❖ 有一只经过训练的蜜蜂只能爬向右侧相邻的蜂房，不能反向爬行。试求出蜜蜂从蜂房a爬到蜂房b的可能路线数。
- ❖ 输入：多组测试数据，每组两个整数a和b，且 $0 < a < b < 50$ 。
- ❖ 输出：蜜蜂从蜂房a爬到蜂房b的可能路线数

## Sample Input

```
2
1 2
3 6
```

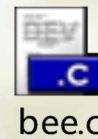


## Sample Output

```
1
3
```

# 算法分析

- ❖ 这是一道很典型的Fibonacci数列类题目，其中的递推关系很明显。
- ❖ 由于“蜜蜂只能爬向右侧相邻的蜂房，不能反向爬行”的限制，决定了蜜蜂到b点的路径只能是从b-1点或b-2点到达的，故
- ❖ **递推公式**为： $f(n)=f(n-1)+f(n-2)$  ( $a+2 \leq n \leq b$ )
- ❖ **边界条件**为： $f(1)=1$ ， $f(2)=1$ 。



## 2

## 蟠桃记

水

- ❖ 猴子第1天摘下若干个桃子,当即吃了一半又一个。第2天又把剩下的桃吃了一半又一个,以后每天都吃前一天剩下的桃子的一半又一个,到第n天猴子想吃时,只剩下一个桃子。问猴子第1天一共摘了多少桃子?
- ❖ 输入：多组测试数据，每组一个正整数 $n(1 < n < 30)$
- ❖ 输出：第一天开始吃的时候桃子的总数。

## Sample Input

2  
4

## Sample Output

4  
22

# 算法分析

- ❖ 已知条件第  $n$  天剩下 1 个桃子，隐含条件每一次前一天的桃子个数等于后一天桃子的个数加 1 的 2 倍。
- ❖ **递推公式**为： $f(i)=(f(i-1)+1)*2$       ( $1<i<n$ )
- ❖ **边界条件**为： $f(i)=1$       ( $i=n$ )
- ❖ 逆推、顺推都可。



# 递推的求解方法

## ❖ 递推算法的求解方法

- 数组：从已知条件入手，填充数组。
- **递归：从问题入手，递归调用函数。**
- 用递推式的通项表达式：
  - 迭加法
  - 待定系数法
  - **特征方程法**
  - **生成函数法**

### 3 童年生活二三事

易

- ❖ Redraiment小时候走路喜欢蹦蹦跳跳，他最喜欢在楼梯上跳来跳去。但年幼的他一次只能走上一阶或者一下子蹦上两阶。现在一共有N阶台阶，请你计算一下Redraiment从第0阶到第N阶共有几种走法。
- ❖ 输入：多组测试数据。每组一个 N
- ❖ 对应每个输入包括一个输出。为redraiment到达第n阶不同走法的数量。

| 样例输入 |
|------|
| 1    |
| 2    |
| 0    |

| 样例输出 |
|------|
| 1    |
| 2    |



# 算法分析

- ❖ 当 $n=1$ ，一级台阶，上法有1种
- ❖ 当 $n=2$ ，二级台阶，上法有2种
- ❖ 当 $n=3$ ，三级台阶，上法有3种
- ❖ ...
- ❖ 当 $n-1$ ， $n-1$ 级台阶，只能从 $n-2$ 台阶和 $n-3$ 台阶上来，上法有 $n-2$ 的上法+ $n-3$ 的上法。
- ❖ 当 $n$ ， $n$ 级台阶，只能从 $n-1$ 台阶和 $n-2$ 台阶上来，上法有 $n-1$ 的上法+ $n-2$ 的上法。
- ❖ **递推公式**： $f(n)=f(n-1)+f(n-2)$
- ❖ **边界条件**： $f(1)=1$ ， $f(2)=2$



baby1.c



baby2.c



baby3.c

# 递推之精髓

- ❖ 首先确认：能否容易的得到简单情况的解？
- ❖ 然后假设：规模为 $n-1$ 的情况已经得到解决。
- ❖ 最后分析：**当规模扩大到 $n$ 时，如何枚举出所有的情况，并且要确保对于每一种子情况都能用已经得到的数据解决。**
- ❖ **强调：**
  - 1、编程中的空间换时间的思想
  - 2、并不一定只是从 $n-1$ 到 $n$ 的分析

# 经验之谈

---

- ❖ 蓝桥杯如果是填空题，直接写出答案
  - 所以用递归、数组均可，不存在超时问题。
  - 当然，你甚至可直接通过数学的方法完成。

## 4

## 切蛋糕

易

- ❖ 直线分割平面
- ❖ 今晚是GG的生日，大家的口福又来了哦，蛋糕可是少不了的，不过这蛋糕可不是人人都有的吃的哦，你得费点脑筋呢，只要你能用 $n$ 刀切出最多块数的蛋糕(刀子每次都垂直于桌面)，你就可以吃到美味的蛋糕了哦，（注意：切出的蛋糕大小可以不等）。怎么样，相信聪明的你吃到这块蛋糕应该不是问题吧？
- ❖ 输入：要切的刀数 $n$
- ❖ 输出：每组输出切成的蛋糕数 $m$

样例输入

1  
10

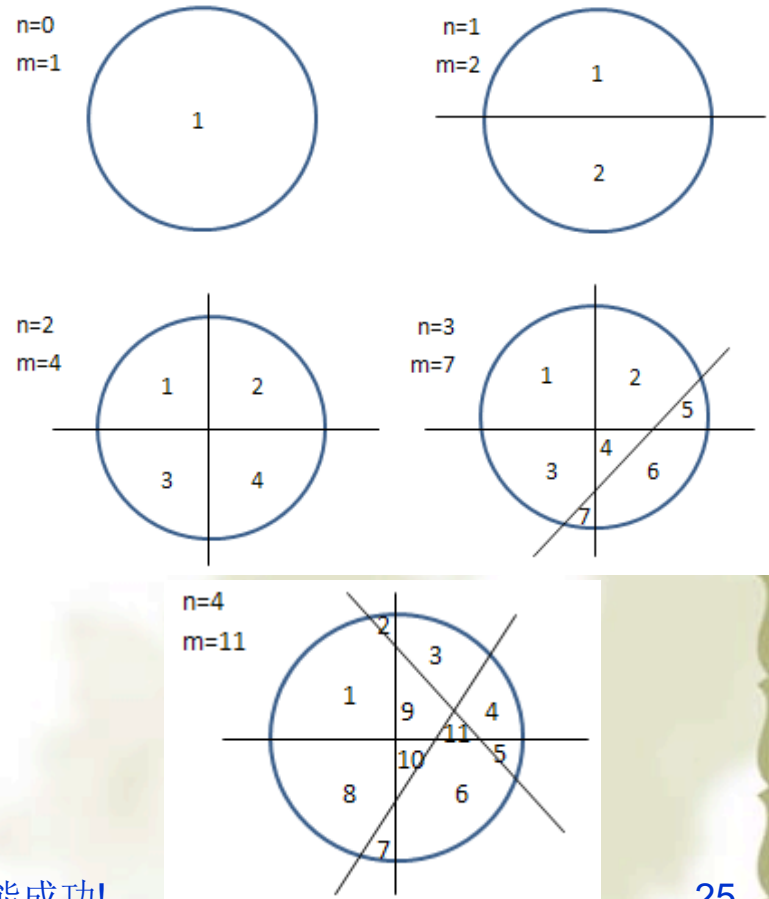
样例输出

2  
56

# 算法分析

## ❖ 刀数 $n$ ，蛋糕数 $m$

- $f(n)$ 为求刀数为 $n$ 的蛋糕数
- $n=0$ 时， $m=1$
- $n=1$ 时， $m=2$
- $n=2$ 时， $m=4$
- $n=3$ 时， $m=7$
- $n=4$ 时， $m=11$



# 算法分析

❖ 归纳法，会推导吗？

–  $n(n+1)/2+1$ ;

❖ 递推公式

–  $f(n)=f(n-1)+n$       能想到吗？ //  $f(n)=m+n$

❖ 边界条件

–  $f(n)=1$        $n=0$



# 算法分析

- ❖ 当有 $n-1$ 条直线时，平面最多被分成了 $f(n-1)$ 个区域。则第 $n$ 条直线要是切成的区域数最多，就必须与每条直线相交且不能有同一交点。这样就会得到 $n-1$ 个交点。
- ❖ 这些交点将第 $n$ 条直线分为2条射线和 $n-2$ 条线段。而每条射线和线段将已有的区域一分为二。
- ❖ 这样就多出了 $2+(n-2)$ 个区域。故， $f(n)=f(n-1)+n$

# 递推的求解方法

## ❖ 递推算法的求解方法

- 数组：从已知条件入手，填充数组。
- 递归：从问题入手，递归调用函数。
- 用递推式的通项表达式：

- **迭加法**
- 待定系数法
- **特征方程法**
- **生成函数法**

**迭加法适用范围**

$$f(n)=f(n-1)+g(n)$$

# 算法分析

❖ 迭加，化简后，

$$f(n) = f(n-1) + n$$

$$f(n-1) = f(n-2) + n-1$$

$$= \dots$$

$$f(1) = f(0) + 1$$

$$f(0) = 1 + 0$$



cake1.c



cake2.c



cake3.c



cake4.c

---

$$f(n) = 1 + 1 + 2 + \dots + n$$
$$= 1 + n(n+1)/2$$

# 5 折线分割平面

中

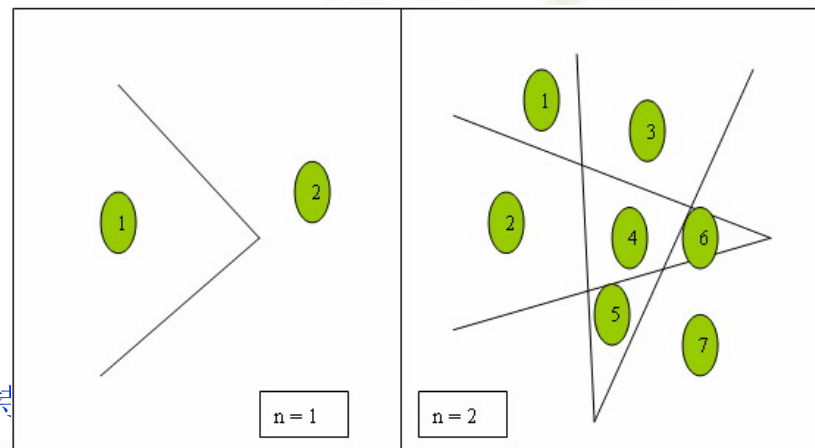
- ❖ 平面上有 $n$ 条折线，问这些折线最多能将平面分割成多少块？
- ❖ 输入：多组测试数据，每组一个整数 $n$  ( $0 < n \leq 10000$ )，表示折线的数量。
- ❖ 输出：输出平面的最大分割数。

## Sample Input

```
2
1
2
```

## Sample Output

```
2
7
```



# 算法分析

❖ 归纳法，会推导吗？

–  $2n^2 - n + 1$

❖ 递推公式：

–  $f(n) = f(n-1) + 4(n-1) + 1$

❖ 边界条件：

–  $f(n) = 1 \quad n = 0$

# 算法分析

- ❖ 根据直线分平面可知，由交点决定了射线和线段的条数，进而决定了新增的区域数。当  $n-1$  条折线时，区域数为  $f(n-1)$ 。
- ❖ 为了使增加的区域最多，则折线的两边的线段要和  $n-1$  条折线的边，即  $2*(n-1)$  条线段相交。那么新增的线段数为  $4*(n-1)$ ，射线数为 2。
- ❖ 但要注意的是，折线本身相邻的两线段只能增加一个区域。故， $f(n)=f(n-1)+4(n-1)+2-1$



# 算法分析

❖ 迭加，化简后，

$$f(n) = f(n-1) + 4(n-1) + 2 - 1$$

$$f(n-1) = f(n-2) + 4(n-2) + 1$$

$$= \dots$$

$$f(2) = f(1) + 4 + 1$$

$$f(1) = f(0) + 1$$

$$f(0) = 1$$

---

$$f(n) = 4(1 + 2 + \dots + (n-1)) + n + 1$$

$$= 4 * n * (n-1) / 2 + n + 1$$

$$= 2n^2 - n + 1$$



line1.c



line2.c



line3.c

## 6

## 汉诺塔I

易

- ❖ 三柱问题。圣庙里有三根宝石针。其中一个针上有由大到小的金片。一次只移动一片，小片必须在大片上面。请问有 $m$ 个金片的汉诺塔金片全部移动到另外一个针上时需要移动的最少步数是多少？
- ❖ 输入：一个整数 $n$ ，表示起始时金片的个数。
- ❖ 输出：把起始针上的金片移动到另外一个针上需要移动的最少步数。

假如输入

4

应当输出

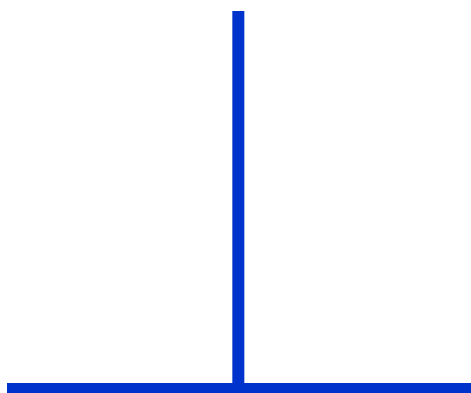
15

# 执行过程

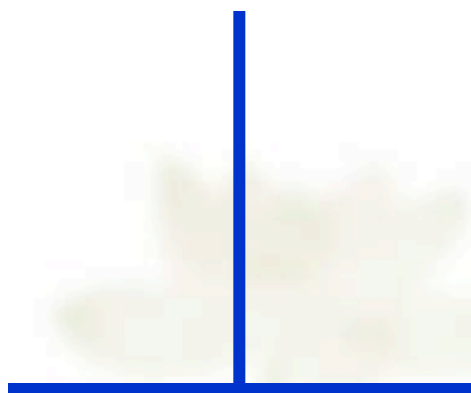
---



A



B



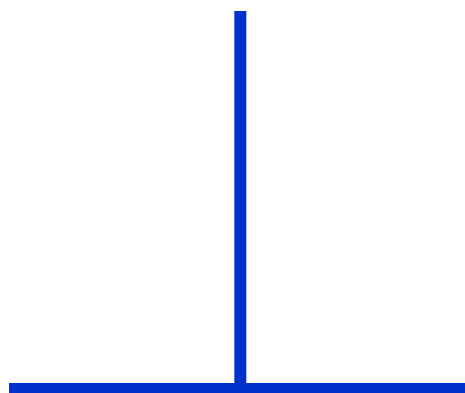
C

# 执行过程

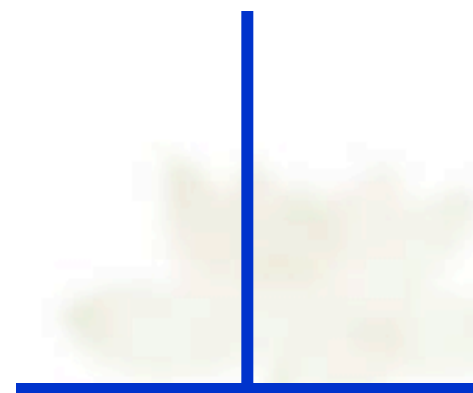
---



A

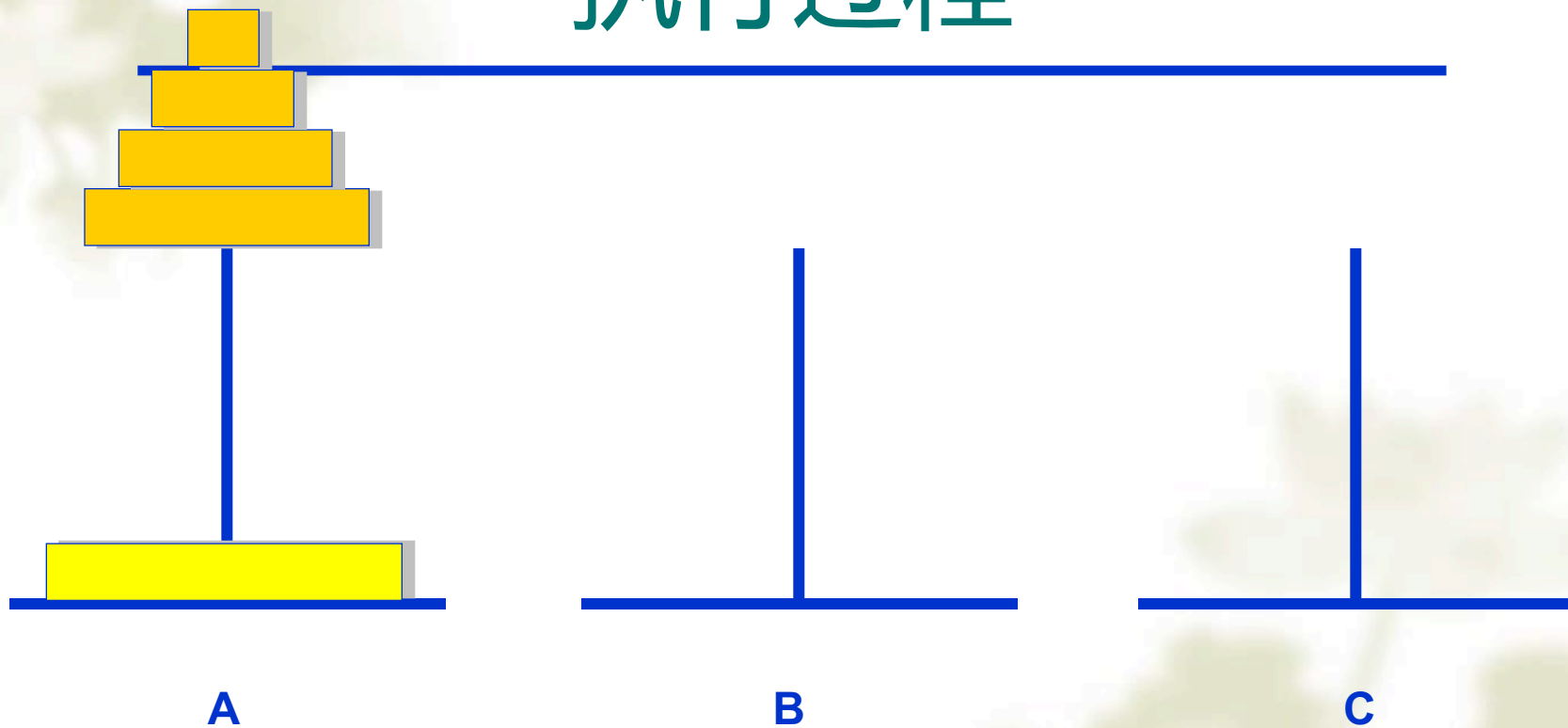


B

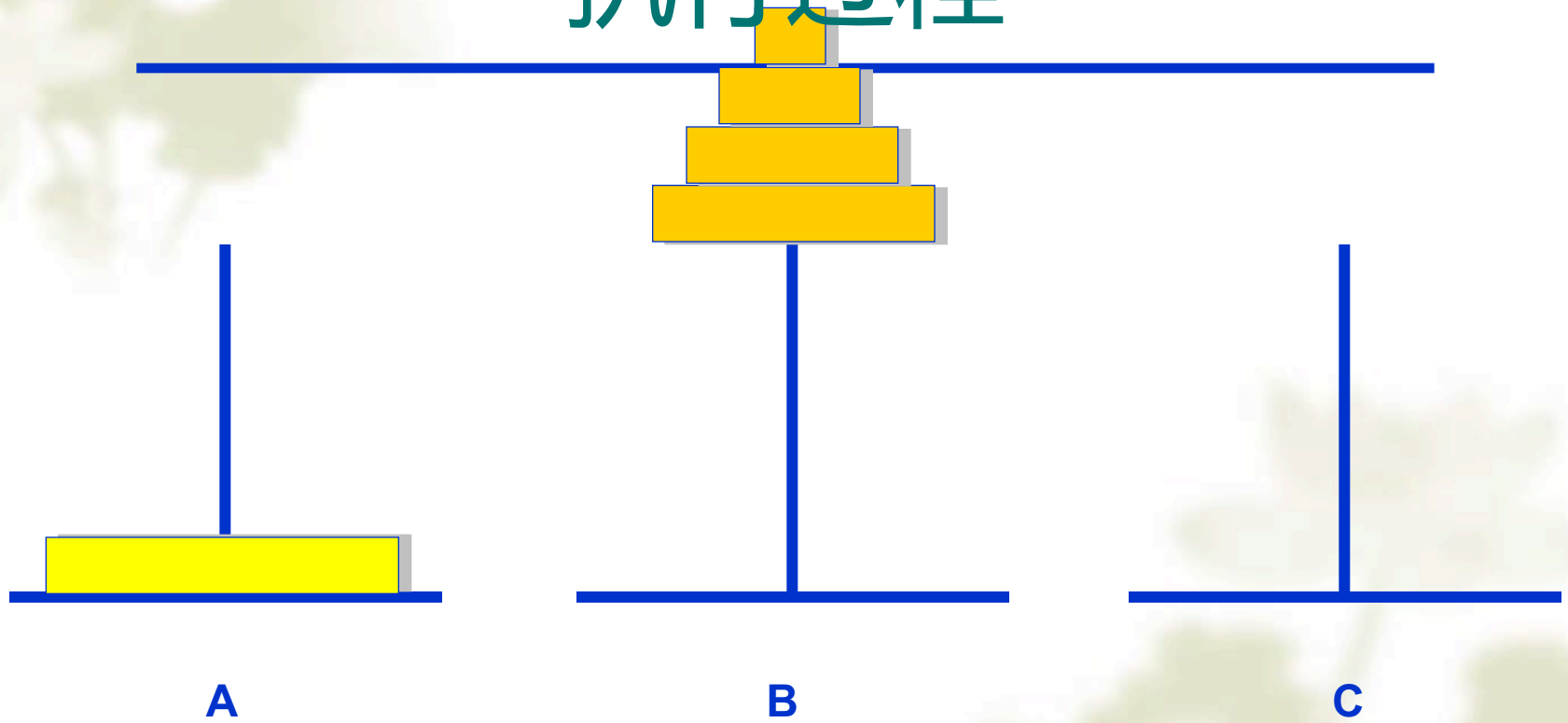


C

# 执行过程

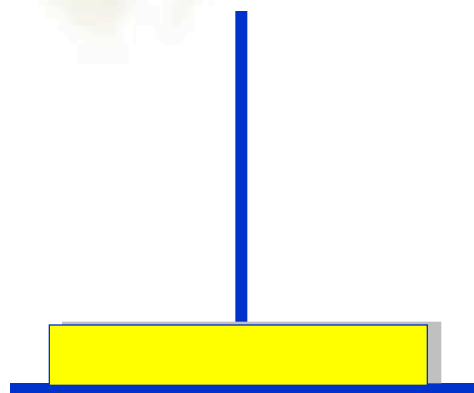


# 执行过程

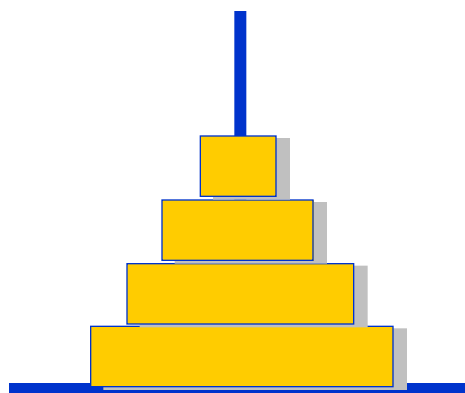


# 执行过程

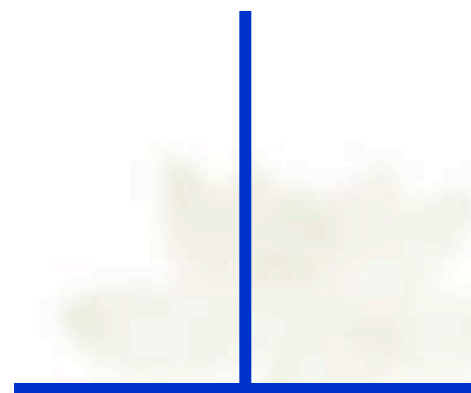
---



A



B

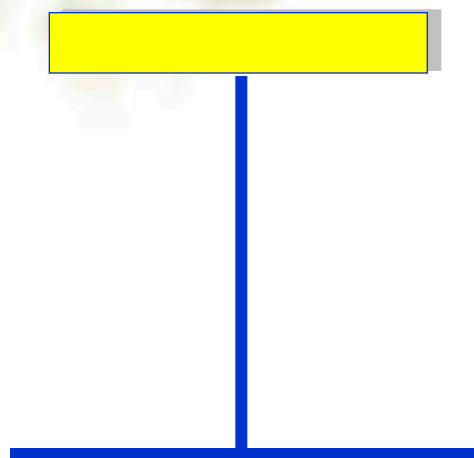


C

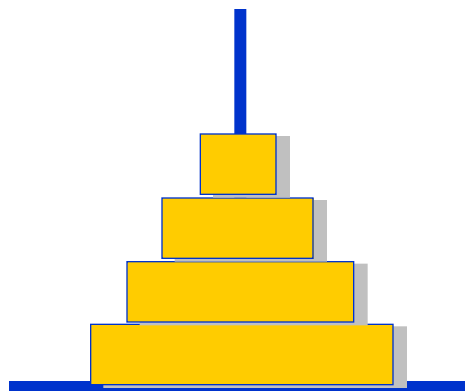


# 执行过程

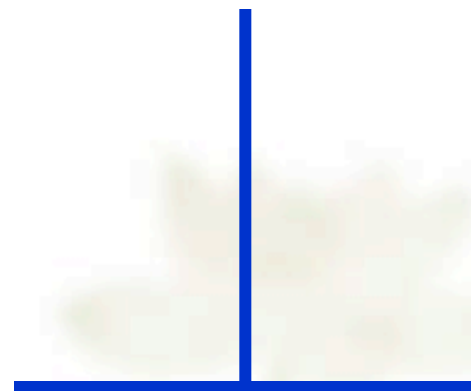
---



A



B

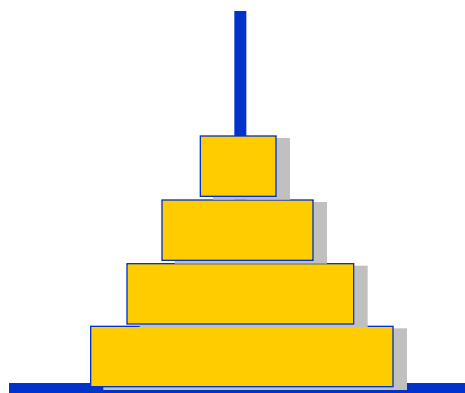


C

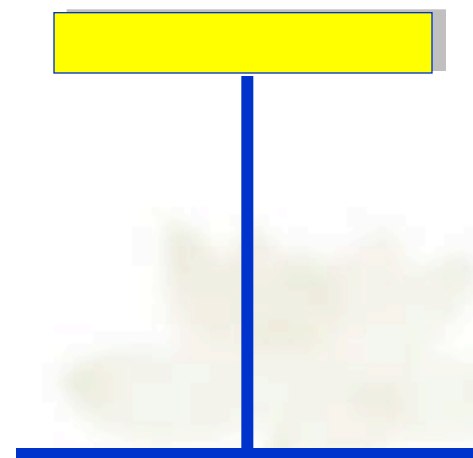
# 执行过程



A



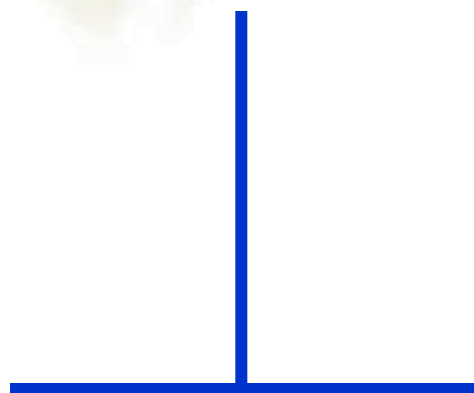
B



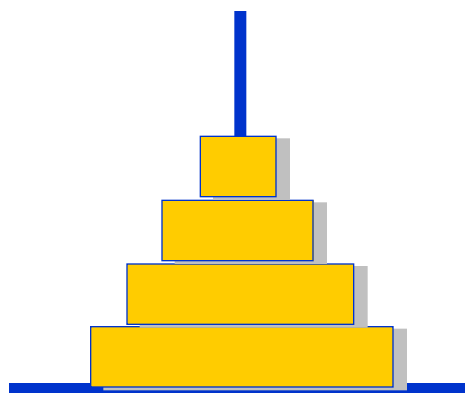
C

# 执行过程

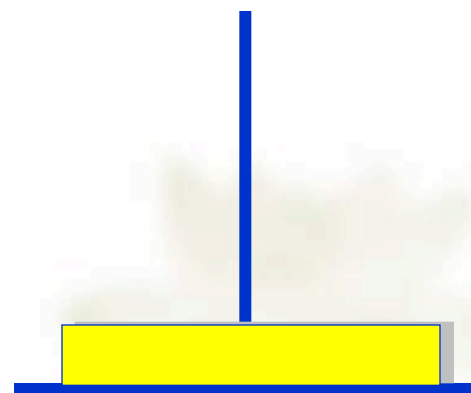
---



A

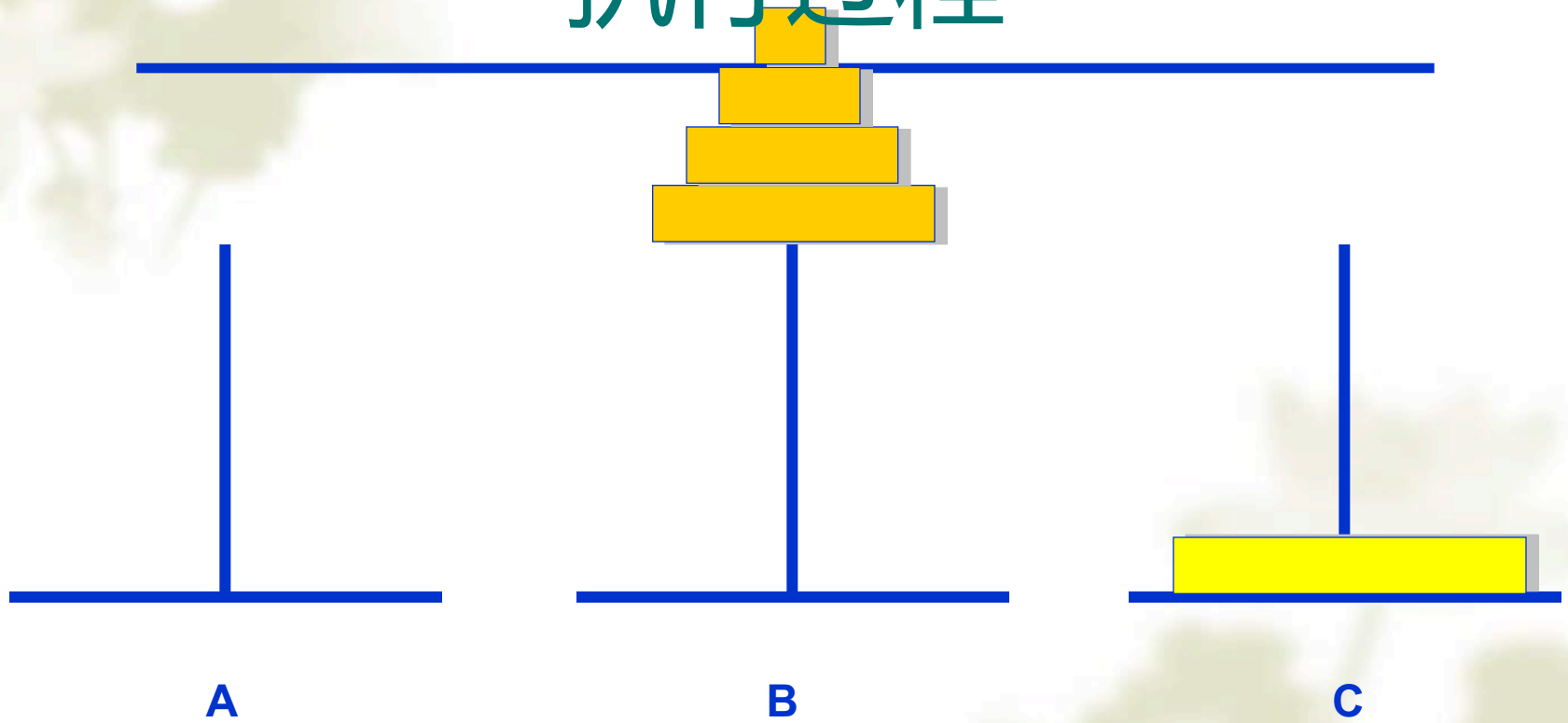


B

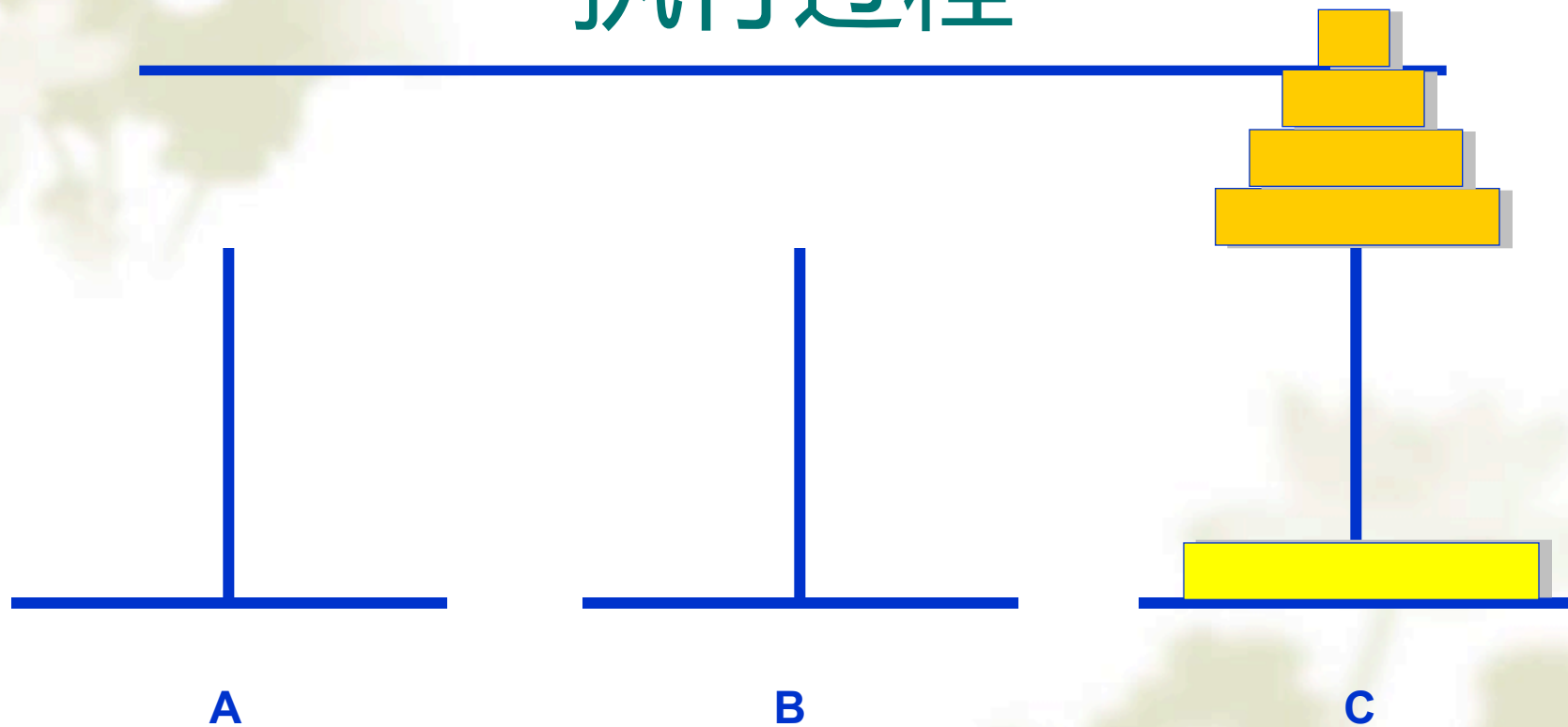


C

# 执行过程

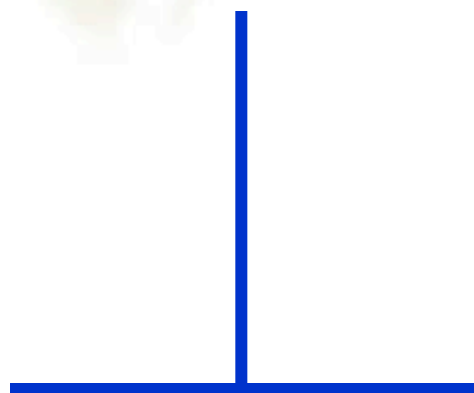


# 执行过程

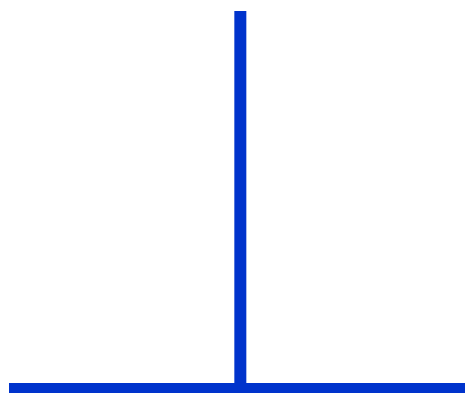


# 执行过程

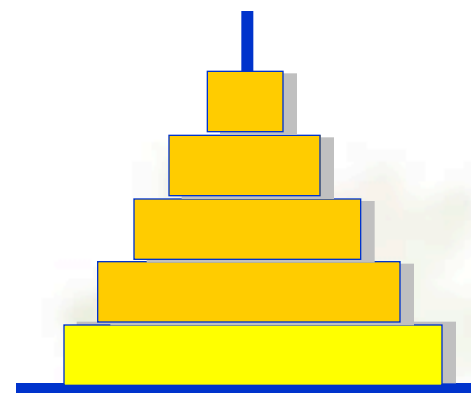
---



A



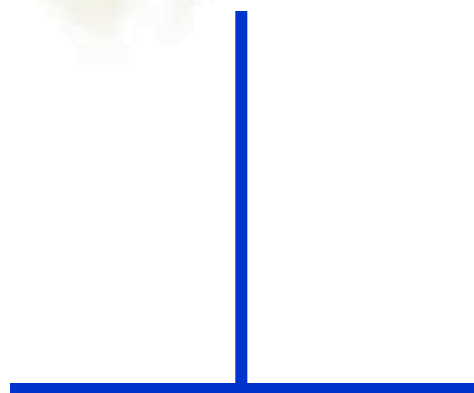
B



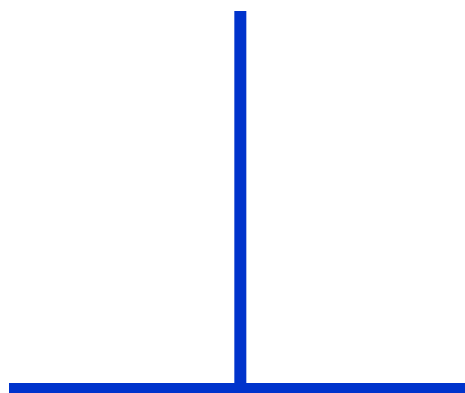
C

# 执行过程

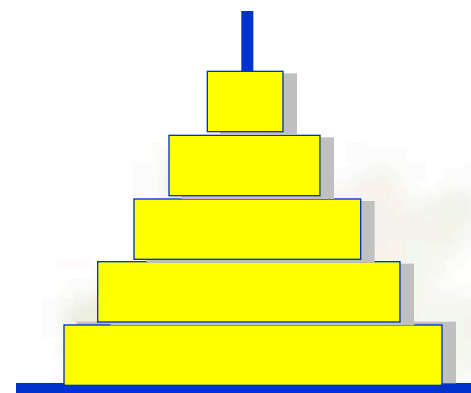
---



A



B



C



# 算法分析

- ❖ 要想把 $n$ 个盘子从A移动到C上，必须
  - 先将 $n-1$ 个盘子从A移到B盘，需要步数 $f(n-1)$ 。
  - 移动第 $n$ 个盘子从A到C，需要步数1。
  - 将 $n-1$ 个盘子从B盘移到C盘，需要步数 $f(n-1)$ 。
- ❖ 最少执行步数为：
  - $f(n) = 2*f(n-1)+1$

# 算法分析

❖ 归纳法，会推导吗？

–  $2^n - 1$

❖ **递推公式：**

–  $f(n) = 2f(n-1) + 1$

❖ **边界条件：**

–  $f(n) = 1 \quad n = 1$

# 递推的求解方法

## ❖ 递推算法的求解方法

- 数组：从已知条件入手，填充数组。
- 递归：从问题入手，递归调用函数。
- 用递推式的通项表达式：

- 迭加法
- 待定系数法 →
- 特征方程法
- 生成函数法

**待定系数法适用范围**  
$$f(n)=a*f(n-1)+g(n)$$

# 算法分析

- ❖ 待定系数法：其基本原理是递推关系两边加上相同的数或相同性质的量，构成数列的每一项都加上相同的数或相同性质的量，使之成为等差或等比数列。
- ❖ 由递推公式： $f(n)=2*f(n-1)+1$ 
  - 令： $f(n)+A=2(f(n-1)+A)$        $A$ 为待定系数
  - 求得 $A=1$ ，即： $f(n)+1=2(f(n-1)+1)$
  - 由等比数列性质（ $a_n=a_1q^{n-1}$ ）得出：
  - $f(n)+1=2^{n-1}(f(1)+1)$
  - 代入 $f(1)=1$ ，所以： $f(n)=2^n - 1$



Hanoi1.c



Hanoi2.c



Hanoi3.c

# 7 阿牛的EOF牛肉串

中

- ❖ 阿牛从家里拿来一块上等的牛肉干，准备在上面刻下一个长度为 $n$ 的只由“E”“O”“F”三种字符组成的字符串（可以只有其中一种或两种字符，但绝对不能有其他字符），阿牛同时禁止在串中出现O相邻的情况，他认为，“OO”看起来就像发怒的眼睛，效果不好。你能帮阿牛算一下一共有多少种满足要求的不同的字符串吗？
- ❖ 输入：一个整数 $n$ 。
- ❖ 输出：全部的满足要求的涂法。
- ❖ 分析：
  - 1：E、O、F
  - 2：EO、EF、EE、FE、FO、FF、OE、OF

| Sample Input |  |
|--------------|--|
| 1            |  |
| 2            |  |

| Sample Output |  |
|---------------|--|
| 3             |  |
| 8             |  |

# 算法分析

- ❖ 每次都在原字符串的最后面再加一个字符，让它仍然是合法的字符串。这就会出现最后一个字符是O和不是O两种情况。不是O肯定没问题，是O则需要看前一个字符。
- ❖ 设 $f(n)$ 求解末尾是O的字符串个数，而 $g(n)$ 求解不是O的字符串个数。
- ❖ 在原来的字符串上再加个O，让它依然合法，则原来的字符串末尾必须不为O，即 $f(n) = g(n-1)$ ；
- ❖ 在原来的字符串上再加非O，则它对前面字符串的末尾没有要求，而且它还有E、F两种。因此 $g(n) = 2 * (f(n-1) + g(n-1))$

# 算法分析

## ❖ 递推公式：

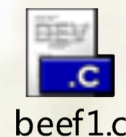
- $f(n) = g(n-1)$  ;
- $g(n) = 2 * (f(n-1) + g(n-1))$

## ❖ 边界条件：

- $f(n)=1$              $n=1$
- $g(n)=2$              $n=1$

## ❖ 从提交结果看：

- 输出 $f(n)+g(n)$ ——**超时**





# 算法分析

## ❖ 递推公式推导

$$\begin{aligned}\text{设 } x(n) &= f(n) + g(n) \\ &= g(n-1) + 2 * (f(n-1) + g(n-1)) \\ &= g(n-1) + 2 * x(n-1) \\ &= 2 * (f(n-2) + g(n-2)) + 2 * x(n-1) \\ &= 2 * x(n-2) + 2 * x(n-1)\end{aligned}$$

// 线性递归了

## ❖ 边界条件：

$$- x(1)=3, x(2)=8$$

## ❖ 从提交结果看：

- 输出 $x(n)$ ——**超时**



beef2.c

# 递推的求解方法

## ❖ 递推算法的求解方法

- 数组：从已知条件入手，填充数组。
- 递归：从问题入手，递归调用函数。
- 用递推式的通项表达式：
  - 迭加法
  - 待定系数法
  - **特征方程法**
  - **生成函数法**

# 特征方程法

- ❖ 如果 $a_1, \dots, a_k$ 是常数，且 $a_k \neq 0, n > k$ ，则递推关系
$$f(n) = a_1 f(n-1) + a_2 f(n-2) + \dots + a_k f(n-k)$$
称为 $k$ 阶常系数线性齐次递推关系。
- ❖ 它的特征方程是：
$$X^k - a_1 X^{k-1} - a_2 X^{k-2} - \dots - a_k = 0$$
- ❖ 只要求出特征方程的根，再由初始条件表达式中的待定系数，便可以得到原递推关系的解。
- ❖ 如果特征方程有 $k$ 个互不相同的解 $X_1, X_2, \dots, X_k$ ，则通解为：
$$f(n) = c_1 X_1^n + c_2 X_2^n + \dots + c_k X_k^n$$
 $c_1, c_2, \dots, c_k$ 待定。

# 算法分析

❖ 递推公式： $f(n)=2f(n-1)+2f(n-2)$

❖ 求解通项：

– 特征方程： $x^2-2x-2=0$

解得： $x_1 = 1 + \sqrt{3}$  ,  $x_2 = 1 - \sqrt{3}$

则： $f(n)=C_1X_1^n+C_2X_2^n$

又因为： $f(1)=3, f(2)=8$

$C_1X_1+C_2X_2=3, C_1X_1^2+C_2X_2^2=8$

解得： $C_1 = \frac{5+3\sqrt{3}}{6+2\sqrt{3}}$  ,  $C_2 = \frac{5-3\sqrt{3}}{6-2\sqrt{3}}$

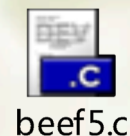
$f(n)=\frac{5+3\sqrt{3}}{6+2\sqrt{3}} * (1 + \sqrt{3})^n + \frac{5-3\sqrt{3}}{6-2\sqrt{3}} * (1 - \sqrt{3})^n$

从提交结果看：**WA**  
大数计算时，易产生  
**误差**，不建议使用。



# 算法分析

- ❖ 递归：超时
- ❖ 递推：超时（递归实现）
- ❖ 通项公式：误差错误（特征根法）
- ❖ ？？？
- ❖ 递推：
  - 数组迭代实现
  - 数组递归实现（动归）

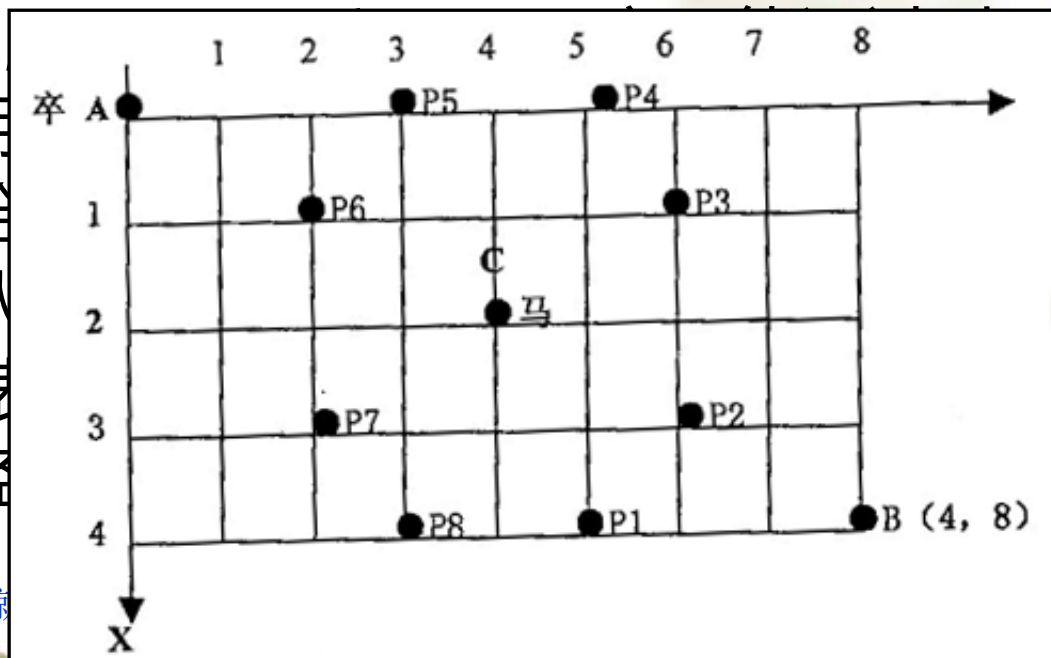


## 8

## 马拦过河卒

## 中

- ❖ A 点有一个过河卒，需要走到目标 B 点。卒行走规则：可以向下、或者向右。同时在棋盘上的任一点有一个对方的马（如图中的C点），该马所在的点和所有跳跃一步可达的点称为对方马的控制点。例如上图 C 点上的马可以控制 9 个点（图中的P1方马的控制点。棋盘用马的位置坐标是需要给。现在要求计算出卒从
- ❖ 输入：四个整数 分别表示
- ❖ 输出：一个整数（路径



示例输入

6 6 3 3

示例输出

6

努力就

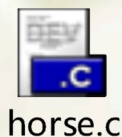
# 算法分析

- ❖ 过河卒是一道很老的题目，竞赛中会经常出现这一类型题的变形。一看到这种类型题，容易想到搜索，但盲目的搜索仅当 $n, m=15$ 就会超时。可试着用递推来进行求解。
- ❖ 根据卒行走的规则，卒要达到棋盘上的一个点，只能有两种可能：从左边过来（左点）或是从上面过来（上点），所以根据加法原理，过河卒到达某一点的路径数目，就等于其到达其相邻的上点和左点的路径数目之和。
- ❖ 因此可用逐列（或逐行）递推的方法求出从起点到终点的路径数目。障碍点（马的控制点）也完全适用，只要将到达该点的路径数目设置为0即可。



# 算法分析

- ❖ 用 $dp[i][j]$ 表示到达点 $(i,j)$ 的路径数目，设马点坐标 $dp[x][y]$ ，A点坐标 $dp[0][0]$ ，B点坐标 $dp[n][m]$ 。
- ❖ 递推公式：
  - $dp[i][j] = dp[i-1][j] + dp[i][j-1]$
- ❖ 边界条件：
  - $dp[0,0] = 1$



## 9

## 火车过站

难

- ❖ 火车从始发站（称为第1站）开出，在始发站上车的人数为 $a$ ，然后到达第2站，在第2站有人上、下车，但上、下车的人数相同，因此在第2站开出时（即在到达第3站之前）车上的人数保持为 $a$ 人。
- ❖ 从第3站起(包括第3站)上、下车的人数有一定规律：**上车的人数都是前两站上车人数之和，而下车人数等于上一站上车人数**，一直到终点站的前一站（第 $n-1$ 站），都满足此规律。(NOIP1998)

# 火车过站

- ❖ 现给出的条件是：共有 $n$ 个车站，始发站上车的人数为 $a$ ，最后一站下车的人数是 $m$ (全部下车)。试问 $x$ 站开出时车上的人数是多少？
- ❖ 输入：四个正整数： $a, n, m, x$
- ❖ 输出：一个整数，为 $x$ 站开出时车上的人数。

## 样例输入

```
【输入样例】 railway.in  
1 6 7 3
```

## 样例输出

```
【输出样例】 railway.out  
2
```

# 算法分析

## ❖ 先找规律

| 站号      | 1   | 2   | 3     | 4      | 5       | 6       |
|---------|-----|-----|-------|--------|---------|---------|
| – 开车时人数 | $a$ | $a$ | $2a$  | $2a+b$ | $3a+2b$ | $4a+4b$ |
| – 上车人数  | $a$ | $b$ | $a+b$ | $a+2b$ | $2a+3b$ | $3a+5b$ |
| – 下车人数  | $0$ | $b$ | $b$   | $a+b$  | $a+2b$  | $2a+3b$ |

## ❖ 结论

- 上车人数： $up(i)=up(i-2)+up(i-1)$  ——Fibonacci
- 下车人数： $down(i)=up(i-1)$
- 开车时人数： $r(i)=r(i-1)+up(i)-down(i)$

# 算法分析

## ❖ 递推公式推导过程

– Fibonacci (1,1,2,3,5,8,13 , ....) 用f(n)表示Fibonacci。

– 上车人数：a   b   a+b   a+2b   2a+3b   3a+5b

$$\text{up}(i) = \text{up}(i-2) + \text{up}(i-1) = \mathbf{f(i-2)*a + f(i-1)*b} \quad // i \geq 2$$

– 开车时人数：a   a   2a   2a+b   3a+2b   4a+4b

–  $\mathbf{r(i)} = r(i-1) + \text{up}(i) - \text{down}(i)$       // 通项公式：迭加法

$$= r(1) + \mathbf{\text{up}(2)} - \text{down}(2) + \mathbf{\text{up}(3)} - \text{down}(3) + \dots + \text{up}(i) - \text{down}(i)$$

$$= r(1) + \cancel{\mathbf{\text{down}(3)}} - \text{down}(2) + \cancel{\mathbf{\text{down}(4)}} - \cancel{\text{down}(3)} + \dots + \cancel{\text{up}(i)} - \cancel{\text{down}(i)}$$

$$= r(1) - \text{down}(2) + \text{up}(i)$$

$$= a - b + f(i-2)*a + f(i-1)*b = \mathbf{(f(i-2)+1)*a + (f(i-1) - 1)*b}$$

❶

# 算法分析

## ❖ 递推公式推导过程

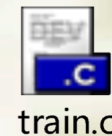
- 第 $n-1$ 站开车时人数为 $m$ ，容易计算出 $b$ ，即：
- $(f(n-3)+1)*a + (f(n-2)-1)*b = m$  ②
- $b=(m-(f(n-3)+1)*a)/(f(n-2)-1)$  代入 ①
- $r(i)=(f(i-2)+1)*a+(f(i-1)-1)*(m-(f(n-3)+1)*a)/(f(n-2)-1)$

## ❖ 递推公式

- 如上

## ❖ 边界条件

- $r(1)=a$     $r(2)=a$     $r(n-1)=m$





# 10 神、上帝以及老天爷

难

- ❖ HDU 2006'10 ACM contest的颁奖晚会隆重开始了！为了活跃气氛，组织者举行了一个别开生面、奖品丰厚的抽奖活动，这个活动的具体要求是这样的：
- ❖ 首先，所有参加晚会的人员都将一张写有自己名字的字条放入抽奖箱中；然后，每人从箱中取一个字条；最后，如果取得的字条上写的就是自己的名字，那么“恭喜你，中奖了！”
- ❖ 不过，这次抽奖活动最后竟然没有一个人中奖！
- ❖ 我的神、上帝以及老天爷呀，怎么会这样呢？现在问题来了，你能计算一下发生这种情况的概率吗？



# 神、上帝以及老天爷

- ❖ 输入：多组测试数据，每组一个整数 $n(1 < n \leq 20)$ ，表示参加抽奖的人数。
- ❖ 输出：输出发生这种情况的百分比，结果保留两位小数(四舍五入)。

**Sample Input**

```
1
2
```

**Sample Output**

```
50.00%
```

# 算法分析

- ❖ 这是组合数学中的典型错排问题。
- ❖ 错排问题来源于“装错信封问题”是由著名的数学家约翰·伯努利的儿子丹尼尔·伯努利提出来的，大意如下：一个人写了 $n$ 封不同的信及相应的 $n$ 个不同的信封，他把这 $n$ 封信都装错了信封，问都装错信封的装法有多少种？
- ❖ 用 $A$ 、 $B$ 、 $C$ ……表示写着 $n$ 位友人名字的信封， $a$ 、 $b$ 、 $c$ ……表示 $n$ 份相应的写好的信纸。把**错装的总数**为记作 **$f(n)$** 。

# 算法分析

- ❖ 假设把a错装进B里了，则包含该错误的一切错装法分两类：
  - (1) b装入A里，这时每种错装的其余部分都与A、B、a、b无关，后续错装应有 $f(n-2)$ 种。
  - (2) b装入A、B之外的一个信封，这时的装信工作实际是把(除a之外的)的 $(n-1)$ 份信纸b、c……装入(除B以外的)  $n - 1$ 个信封A、C……，显然这时装错的方法有 $f(n-1)$ 种。

# 算法分析

- ❖ 总之，在a装入B的错误下，共有错装法 $f(n-2)+f(n-1)$ 种。a装入C，装入D.....的n-2种错误下，同样都有 $f(n-2)+f(n-1)$ 种错装法。
- ❖ 因此： $f(n)=(n-1)*(f(n-1)+f(n-2))$
- ❖ 1个元素没有全错位排列，2个元素的全错位排列有1种，3个元素的全错位排列有2种，4个元素的全错位排列有9种，5个元素的全错位排列有44种.....
- ❖ **递推公式**： $f(n)=(n-1)*(f(n-1)+f(n-2))$
- ❖ **边界条件**： $f(1)=0$ ， $f(2)=1$



god.c

# 内容提纲

- ❖ 认识递推
  - 递推的概念
  - 生活中的递推示例
- ❖ 递推关系的建立与求解方法
  - 递推关系的建立
  - 递推的求解方法
- ❖ 算法示例
  - 模拟示例
  - **练习系统**
  - 蓝桥真题

*You are here!*  
你在这儿！

## 1

## Hanoi问题

易

- ❖ 给定 $n$ 只盘子，3根柱子，但是允许每次最多移动相邻的 $m$ 只盘子（当然移动盘子的数目可以小于 $m$ ），最少需要多少次？
- ❖ 例如 $n=5$ ， $m=2$ 时，可以分别将最小的2个盘子、中间的2个盘子以及最大的一个盘子分别看作一个整体，这样可以转变为 $n=3$ ， $m=1$ 的情况，共需要移动7次。
- ❖ 输入： $n$ 和 $m$
- ❖ 输出：表示需要移动的最少次数。

略



# 内容提纲

- ❖ 认识递推
  - 递推的概念
  - 生活中的递推示例
- ❖ 递推关系的建立与求解方法
  - 递推关系的建立
  - 递推的求解方法
- ❖ 算法示例
  - 模拟示例
  - 练习系统
  - **蓝桥真题**

*You are here!*  
你在这儿！



# 1 振兴中华(2013-A组)

易

- ❖ 小明参加了学校的趣味运动会，其中的一个项目是：跳格子。地上画着一些格子，每个格子里写一个字，如图所示。比赛时，先站在左上角的写着“从”字的格子里，可以横向或纵向跳到相邻的格子里，但不能跳到对角的格子或其它位置。一直要跳到“华”字结束。要求跳过的路线刚好构成“从我做起振兴中华”这句话。请你帮助小明算一算他一共有多少种可能的跳跃路线呢？

|   |   |   |   |   |
|---|---|---|---|---|
|   |   |   |   |   |
| 从 | 我 | 做 | 起 | 振 |
| 我 | 做 | 起 | 振 | 兴 |
| 做 | 起 | 振 | 兴 | 中 |
| 起 | 振 | 兴 | 中 | 华 |
|   |   |   |   |   |

# 算法分析

- ❖ 跟小蜜蜂的题目类似，就是递推题目， $f(i,j)$ 格子的走法应该为： $f(i,j)=f(i-1,j)+f(i,j-1)$ 。



## 2 第39级台阶(2012)

中

- ❖ 小明刚刚看完电影《第39级台阶》，离开电影院的时候，他数了数礼堂前的台阶数，恰好是39级！站在台阶前，他突然又想着一个问题：
- ❖ 如果我每一步只能迈上1个或2个台阶。先迈左脚，然后左右交替，最后一步是迈右脚，也就是说一共要走偶数步。那么，上完39级台阶，有多少种不同的上法呢？
- ❖ 请你利用计算机的优势，帮助小明寻找答案。

# 算法分析

- ❖ 踏到第 $i$ 层有2种方法，第 $i-1$ 层向上迈，或第 $i-2$ 层向上迈。不过此时只需加一个判断是左还是右脚踏的变量。我们需要的是最后一步迈右脚。
- ❖ 用 $n$ 表示要跨的步数
  - $n=2$ 时，不管 $flag$ 是0还是1，都各有一种跨法。如果是左脚，直接跨2级结束，如果是右脚，则左脚跨上1级。
- ❖ 用 $flag$ 表示左脚还是右脚
  - $flag=0$ ，表示这一步要跨左脚，即奇数步。
  - $flag=1$ ，表示这一步要跨右脚，即偶数步。
  - 当台阶只剩一个时，必须要跨右脚。



ladder5.c



ladder1.c



ladder2.c



ladder3.c



ladder4.c

# 作业

---

- ❖ 骨牌铺方格
- ❖ 母牛的故事
- ❖ 不容易系列之二
- ❖ 封闭曲线分割平面
- ❖ 平面分割空间
- ❖ 汉诺塔III
- ❖ 汉诺塔IV
- ❖ LELE的RPG难题
- ❖ 不容易系列之一
- ❖ 考新郎

## 1

## 骨牌铺方格

易

- ❖ 在 $2 \times n$ 的一个长方形方格中,用一个 $1 \times 2$ 的骨牌铺满方格,输入 $n$ ,输出铺放方案的总数.  
例如 $n=3$ 时,为 $2 \times 3$ 方格,骨牌的铺放方案有三种,如下图:
- ❖ 输入:多组测试数据,每行一个整数 $n$ ,表示长方形方格的规格是 $2 \times n$  ( $0 < n \leq 50$ ).
- ❖ 输出:输出铺放方案的总数.

Sample Input




1  
3  
2

Sample Output

1  
3  
2



# 算法分析

- ❖ 当 $n=1$ 时，有2个格子。只能**一**种铺法。 
- ❖ 当 $n=2$ 时，有4个格子，**两**种铺法，并列横排或并列竖排。 
- ❖ 当 $n=3$ 时，有6个格子，**三**种铺法。 
- ❖ 当 $n=4$ 时，有8个格子，**五**种铺法。
- ❖ 由此可见，还是Fibonacci。注意\_\_int64。



## 2

## 母牛的故事

易

- ❖ 有一头母牛，它每年年初生一头小母牛。每头小母牛从第四个年头开始，每年年初也生一头小母牛。请编程实现在第 $n$ 年的时候，共有多少头母牛？
- ❖ 输入：整数 $n$ 。
- ❖ 输出：第 $n$ 年时母牛的数量。

| 样例输入 |
|------|
| 2    |
| 4    |
| 5    |
| 0    |

| 样例输出 |
|------|
| 2    |
| 4    |
| 6    |

# 算法分析

❖ 推导过程：

| 年   | 母牛  | 一岁  | 两岁  | 三岁  | 总数            |
|-----|-----|-----|-----|-----|---------------|
| 1   | 1   | 0   | 0   | 0   | 1             |
| 2   | 1   | 1   | 0   | 0   | 2             |
| 3   | 1   | 1   | 1   | 0   | 3             |
| 4   | 1   | 1   | 1   | 1   | 4             |
| 5   | 2   | 2   | 1   | 1   | 6             |
| 6   | 3   | 3   | 2   | 1   | 9             |
| 7   | 4   | 4   | 3   | 2   | 13            |
| ... | ... | ... | ... | ... | ...           |
| i   |     |     |     |     | $f(i)+f(i-3)$ |

# 算法分析

## ❖ 递推公式：

- $f(n)=f(n-1)+f(n-3)$

## ❖ 边界条件：

- $f(n)=n$   $n \leq 4$

- $f(n)=f(n-1)+f(n-3)$   $n \geq 5$

# 3 不容易系列之二

易

- ❖ 徐老汉养了不少羊，决定把这些羊都赶到集市去卖。从出发地到交易地要经过 $n$ 个收费站，没钱。收费员就将他的羊拿走一半，看到老汉泪水涟涟，又还给老汉一只。后面每过一个收费站，都是拿走当时羊的一半，然后退还一只，等到老汉到达市场，就只剩下3只羊了。帮忙算一下老汉最初有多少只羊吗？
- ❖ 输入：整数 $n$ ，表示测试数据组数，整数 $a$ ，表示收费站的数量。
- ❖ 输出：最初的羊的数量。

Sample Input

2  
1  
2

Sample Output

4  
6

# 算法分析

- ❖ 1个收费站，最初羊为：4
- ❖ 2个收费站，最初羊为：6
- ❖ 3个收费站，最初羊为：10

...

- ❖ **递推公式：**

- $f(n) = (f(n-1) - 1) * 2$

- ❖ **边界条件**

- $f(1) = 4$

# 算法分析

## ❖ 待定系数：

$$- f(n) + A = 2(f(n-1) + A)$$

$$f(n) + A = 2f(n-1) + 2A - 2 + 2$$

$$f(n) + A = 2(f(n-1) - 1) + 2A + 2$$

$$\text{求得：} A = -2$$

$$f(n) - 2 = 2 (f(n-1) - 2)$$

由等比数列性质（ $a_n = a_1 q^{n-1}$ ）得出：

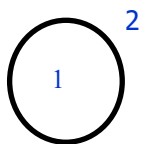
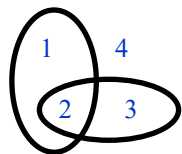
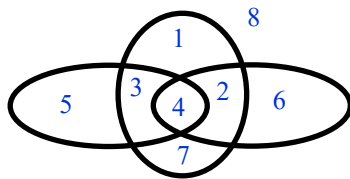
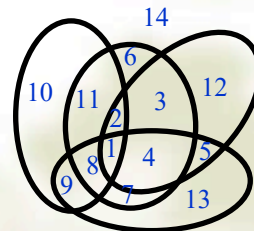
$$f(n) - 2 = 2^{n-1}(f(1) - 2)$$

代入 $f(1)=4$ ，所以： **$f(n) = 2^n + 2$**

# 4 封闭曲线分割平面

易

- ❖ 简述：设有 $n$ 条封闭曲线画在平面上，而任何两条封闭曲线恰好相交于两点，且任何三条封闭曲线不相交于同一点，问这些封闭曲线把平面分割成的区域个数？

 **$n=1$**  **$n=2$**  **$n=3$**  **$n =$   
 $4$**



# 算法分析

---

❖ 归纳法，会推导吗？

–  $f(n)=n^2-n+2$

❖ **递推公式：**

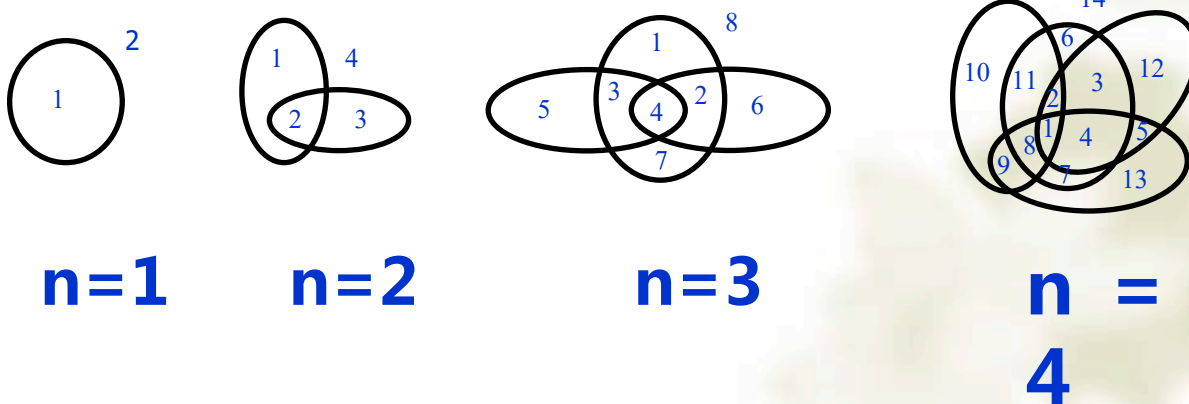
–  $f(n)=f(n-1)+2(n-1)$

❖ **边界条件：**

–  $f(1)=2$

# 算法分析

- ❖ 当 $n-1$ 个圆时，区域数为 $f(n-1)$ ，那么第 $n$ 个圆就必须与前 $n-1$ 个圆相交，则第 $n$ 个圆被分为 $2(n-1)$ 段线段，增加了 $2(n-1)$ 个区域。故， $f(n)=f(n-1)+2(n-1)$ 。



# 算法分析

❖ 迭加，化简后，

$$f(n) = f(n-1) + 2(n-1)$$

$$f(n-1) = f(n-2) + 2(n-2)$$

$$= \dots$$

$$f(2) = f(1) + 2$$

$$f(1) = 2$$

---

$$f(n) = 2 + 2 + 4 + \dots + 2(n-1)$$

$$= n^2 - n + 2$$

## 5

## 平面分割空间

难

- ❖ 今年是杭电建校五十周年，该送给母校一个怎样的礼物呢？送一个球形大蛋糕吧。
- ❖ 如果校长在蛋糕上切了 $N$ 刀（每一刀都是一个绝对的平面），最多可把这个球形蛋糕切成几块？

# 算法分析

❖ 归纳法，会推导吗？

- $(n^3+5n)/6+1$

❖ **递推公式：**

- $f(n)=f(n-1)+g(n-1)$

- $g(n)=n(n+1)/2+1$

❖ **边界条件：**

- $f(1)=2$

# 算法分析

- ❖ 由二维的分割问题可知，平面分割与线之间的交点有关，即交点决定射线和线段的条数，从而决定新增的区域数。
- ❖ 试想在三维中则是否与平面的交线有关呢？当有 $n-1$ 个平面时，分割的空间数为 $f(n-1)$ 。要有最多的空间数，则第 $n$ 个平面需与前 $n-1$ 个平面相交，且不能有共同的交线。即最多有 $n-1$ 条交线。
- ❖  $n-1$ 条交线能把 $n$ 个平面最多分割多少区域？当然是 $4-1$ 的计算方法啦。此平面将原有的空间一分为二，则最多增加 $g(n-1)$ 个空间， $g$ 函数就是 $4-1$ 例题的函数。

# 算法分析

- ❖  $n$ 平面与 $n-1$ 平面有 $n-1$ 条交线， $n-1$ 条交线则新增 $g(n-1)$ 区域（ $g(n-1)$ 解法见例4-1，直线分割平面）
- ❖ **递推公式：**
  - $f(n)=f(n-1)+g(n-1)$
  - $g(n)=n(n+1)/2+1$
- ❖ **边界条件：**
  - $f(1)=2$



## ❖ 迭加，化简后

$$f(n) = f(n-1) + g(n-1)$$

$$f(n-1) = f(n-2) + g(n-2)$$

$$= \dots$$

$$f(2) = f(1) + g(1)$$

$$f(1) = 2$$

---

$$\begin{aligned} f(n) &= 2 + g(1) + g(2) + g(3) \dots + g(n-1) \\ &= 2 + (1 \cdot 2/2 + 1) + (2 \cdot 3/2 + 1) + (3 \cdot 4/2 + 1) \dots + ((n-1) \cdot n/2 + 1) \\ &= 2 + (1 \cdot 2 + 2 \cdot 3 + 3 \cdot 4 \dots + (n-1) \cdot n)/2 + n - 1 \\ &= ((2-1) \cdot 2 + (3-1) \cdot 3 + (4-1) \cdot 4 + \dots + (n-1) \cdot n)/2 + n + 1 \\ &= (2 \cdot 2 + 3 \cdot 3 + 4 \cdot 4 + \dots + n \cdot n - 2 - 3 - 4 - \dots - n)/2 + n + 1 \\ &= (1^2 + 2^2 + 3^2 + 4^2 + \dots + n^2 - (1 + 2 + 3 + 4 + \dots + n))/2 + n + 1 \\ &= (n(n+1)(2n+1)/6 - n(n+1)/2)/2 + n + 1 \\ &= \mathbf{(n^3 + 5n)/6 + 1} \end{aligned}$$

## 6

## 汉诺塔III

易

- ❖ 汉诺塔是将最左边杆上的盘全部移到右边的杆上，条件是一次只能移一个盘，且不允许大盘放在小盘上面。
- ❖ 现在改变游戏的玩法，不允许直接从最左(右)边移到最右(左)边(每次移动一定是移到中间杆或从中间移出)，也不允许大盘放到下盘的上面。
- ❖ 现有N个圆盘，至少需要多少次移动才能把这些圆盘从最左边移到最右边？
- ❖ 输入：每次输入一个N值。
- ❖ 输出：移动最小的次数。

## Sample Input

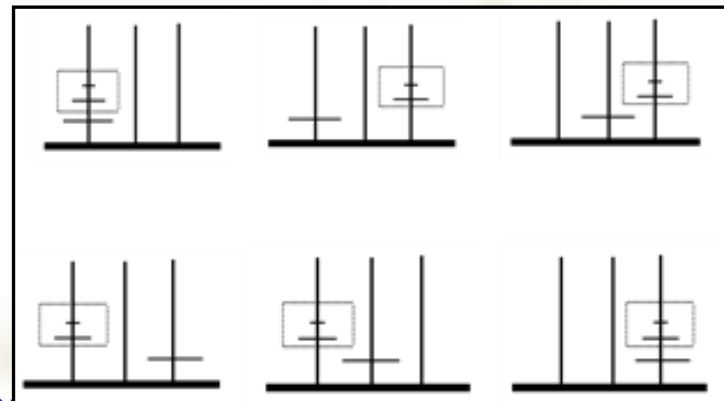
1  
3  
12

## Sample Output

2  
26  
531440

# 算法分析

- ❖  $n-1$ 个盘子移动到右边柱，需要步数 $f(n-1)$
- ❖ 第 $n$ 个盘子移动到中间柱，需要步数1
- ❖  $n-1$ 个盘子移动到左边柱，需要步数 $f(n-1)$
- ❖ 第 $n$ 个盘子移动到右边柱，需要步数1
- ❖  $n-1$ 个盘子移动到右边柱，需要步数 $f(n-1)$
- ❖ 故，总步数为 $3*f(n-1)+2$



# 算法分析

---

❖ 归纳法，会推导吗？

–  $3^n - 1$

❖ **递推公式：**

–  $f(n) = 3f(n-1) + 2$

❖ **边界条件：**

–  $f(1) = 2$

# 算法分析

- ❖ 由递推公式： $f(n)=3*f(n-1)+2$
- ❖ 令： $f(n)+A=3(f(n-1)+A)$
- ❖ 求得 $A=1$ ，即： $f(n)+1=3(f(n-1)+1)$
- ❖ 由等比数列性质（ $a_n=a_1q^{n-1}$ ）得出：
- ❖  $f(n)+1 = (f(1)+1) * 3^{n-1}$
- ❖ 代入 $f(1)=2$ ，所以： $f(n) = 3^n - 1$

## 7

## 汉诺塔IV

## 中

- ❖ 还记得汉诺塔III吗？规则是这样的：不允许直接从最左(右)边移到最右(左)边(每次移动一定是移到中间杆或从中间移出)，也不允许大盘放到小盘的上面。xhd想如果允许最大的盘子放到最上面会怎么样呢？（**只允许最大的放在最上面**）当然最后需要的结果是盘子从小到大排在最右边
- ❖ 输入一个正整数 $n$ ，表示有 $n$ 个盘子。
- ❖ 输出最少需要的摆放次数。

Sample Input

2  
1  
10

Sample Output

2  
19684

# 算法分析

- ❖ 因为最大的盘子移动规矩和其他盘子不同，所以要单独移动。设 $g(n)$ 为 $n$ 个盘子移动到相邻柱的步数， $f(n)$ 为 $n$ 个盘子最左柱移动到最右柱。
  - 1) 前 $n-1$ 个盘子 $A \rightarrow B$ ，需要移动步数 $g(n-1)$
  - 2) 第 $n$ 个盘子 $A \rightarrow B \rightarrow C$ ，需要移动步数2
  - 3) 前 $n-1$ 个盘子 $B \rightarrow C$ ，需要移动步数 $g(n-1)$
- ❖ 总的移动步数为： $f(n) = 2 * g(n-1) + 2$  ----- ①
- ❖ 如何计算相邻柱移动的步数 $g(n)$ ？
  - 1) 前 $n-1$ 个盘子 $A \rightarrow C$ ，需要移动步数 $f(n-1)$
  - 2) 第 $n$ 个盘子 $A \rightarrow B$ ，需要移动步数1
  - 3) 前 $n-1$ 个盘子 $C \rightarrow B$ ，需要移动 $g(n-1)$
  - 总的移动步数： $g(n) = f(n-1) + g(n-1) + 1$  ----- ②



# 算法分析

- ❖ 注意：本题中只有“最”大的盘子（即第 $n$ 个）可以放在小盘子上，其他的是不可以的，所以，❷式中 $n-1$ 个盘子的移动方法 $f(n-1)$ 和❶中的 $f(n)$ 不同。因为❷式中 $f(n-1)$ 不可以“以大欺小”，而❶式中 $f(n)$ 则可以“恃强凌弱”
- ❖ 故，将❶中的函数改为 $ans(n)$
- ❖ 而， $f(n-1)=3*f(n-2)+2=3^{(n-2)}-1$

# 算法分析

❖ 归纳法，会推导吗？

- $3^{(n-1)}+1$

❖ **递推公式：**

- $\text{ans}(n) = 2 * g(n-1) + 2$

- $g(n) = f(n-1) + g(n-1) + 1$

- $f(n) = 3f(n-1) + 2$

❖ **边界条件：**

- $\text{ans}(1) = 2, g(1) = 1, f(1) = 2$

# 算法分析

- ❖  $f(n)=3^{n-1}$  // 已经推导过了
- ❖ 代入到  $g(n)=f(n-1)+g(n-1)+1$  中，得：
  - $g(n)=3^{n-1}-1+g(n-1)+1$   
 $=g(n-1)+3^{n-1}$
- ❖ 可以使用**迭加法**求解 $g(n)$ ：
  - $g(n)=1+3+3^2+\dots+3^{n-1}=(3^n-1)/2$
- ❖ 代入 $ans(n)=2*g(n-1)+2$ ，得：
  - $=2*(3^{n-1}-1)/2+2=3^{n-1}+1$

# 8 LELE的RPG难题

难

- ❖ LELE最近研究起了著名的RPG难题：有排成一行的  $n$  个方格，用红(Red)、粉(Pink)、绿(Green)三色涂每个格子，每格涂一色，要求任何相邻的方格不能同色，且首尾两格也不同色。求全部的满足要求的涂法。
- ❖ 输入：输入数据包含多个测试实例，每个测试实例由一个整数  $N$  组成，( $0 < n \leq 50$ )。
- ❖ 对于每个测试实例，请输出全部的满足要求的涂法，每个实例的输出占一行。

Sample Input

1  
2

Sample Output

3  
6

# 算法分析

- ❖ 当 $n=1$ 时，一个格子 必然只有3种涂法，红、粉、绿。
- ❖ 当 $n=2$ 时，第1个格可以选红、粉、绿；但第2个格子，只能选剩余的两个，因而是 $2*3=6$ 。
- ❖ 当 $n=3$ 时，第1个格可以选红、粉、绿；但第2个格子，只能选剩余的两个；第3个格子，可以选剩余的那个颜色和第一个格子颜色，然头尾不能相同，那么尾必定只有1种，因而是 $3*2*1=6$ 。

# 算法分析

- ❖ 当 $n=4$ 时，第1个格可以选红、粉、绿；第2个格，只能选剩余的两个，之后就要分情况考虑了
- ❖ 当第3个格放的是与第一和第二都不相同的，那么第四格格只能放1种  $3*2*1*1=6$ 。
- ❖ 如果第3格放的跟第一种相同的，第四格就能放2种的了， $3*2*2=12$  总共有 $12+6=18$ 种。
- ❖ 如果有 $n$ 个方格，当对第 $n$ 个方格填色时，有两种情况：



# 算法分析

- ❖ 应该已经对前面 $n-1$ 个方格填好了色，有 $f(n-1)$ 种情况，此时第 $n-1$ 个跟第一个颜色一定不一样，所以第 $n$ 个只有一种选择。
- ❖ 对前面 $n-2$ 个方格填好色，有 $f(n-2)$ 种情况，第 $n-1$ 个空格颜色跟第一个颜色一样，只有一种可能，最后第 $n$ 个方格可以填两种颜色（因为 $n-1$ 和 $1$ 是同种颜色），所以是 $2*f(n-2)$ ；
- ❖ **递推公式**： $f(n)=f(n-1)+2*f(n-2)$        $n \geq 4$
- ❖ **边界条件**： $f(1)=3, f(2)=6, f(3)=6$



# 9 不容易系列之一

中

- ❖ HDU某同学给 $n$ 个网友每人写了一封信，要命的是，他竟然把所有的信都装错了信封！注意了，是全部装错哟！请问一共有多少种可能的错误方式呢？
- ❖ 输入：正整数 $n$ ，表示网友的人数。
- ❖ 输出：可能的错误方式的数量。

## Sample Input

2  
3

## Sample Output

1  
2

# 算法分析

---

- ❖ 错排问题
- ❖ **递推公式** :  $f(n)=(n-1)*(f(n-1)+f(n-2))$
- ❖ **边界条件** :  $f(1)=0$  ,  $f(2)=1$

## 10

## 考新郎

难

- ❖ 省城HZ举行了一场集体婚礼，司仪想出了有一个节目，叫“考新郎”，具体操作如下：
- ❖ 首先，每位新娘打扮得一模一样，并盖上红盖头随机坐成一排；然后，让各位新郎寻找自己的新娘。每人只准找一个，并且不允许多人找一个。
- ❖ 假设一共有 $n$ 对新婚夫妇，其中有 $m$ 个新郎找错了新娘，求发生这种情况一共有多少种可能。

Sample Input

```
2
2 2
3 2
```

Sample Output

```
1
3
```

# 算法分析

- ❖ 组合+错排
- ❖ 这道题就是N中有多少M个数的错排。
- ❖ 因此先找到N个新郎中M个错一共有几种，显然是 $C^M_N = N! / (M! * (N-M)!)$ 。即 $C^M_N = N! / M! / (N-M)!$ 。
- ❖ 然后在求出M个数的错排个数，递推关系： $f[n] = (n-1) * (f[n-1] + f[n-2])$



**下课了**