

第7讲 递归函数2








"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

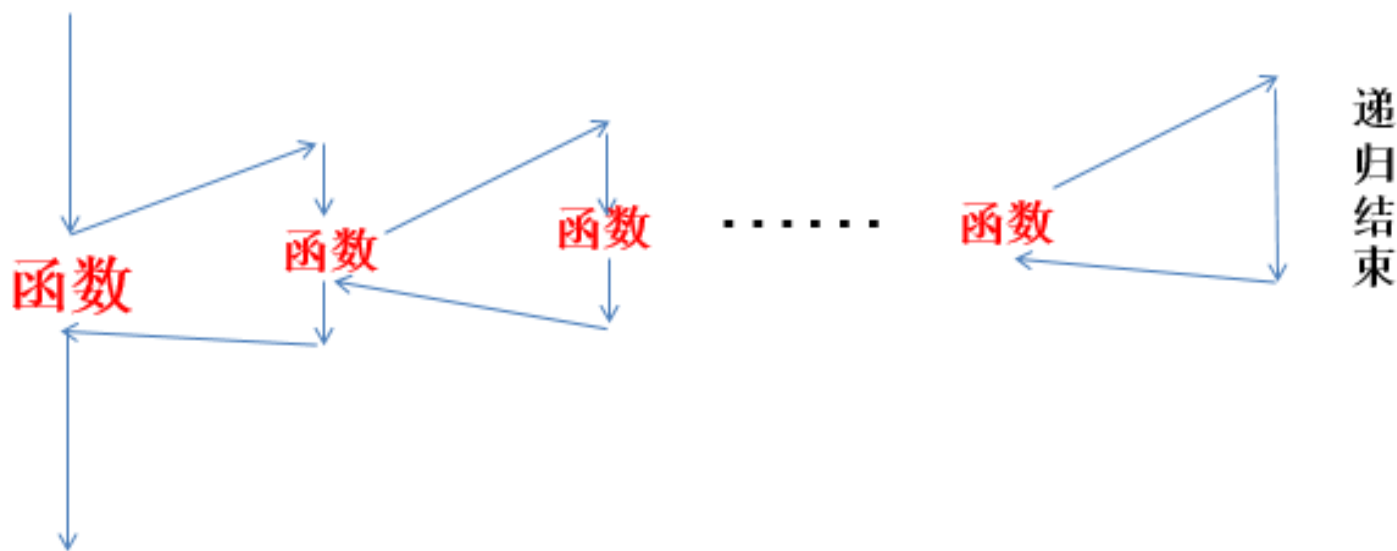
.....

"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

- 
- 儿子：“爸爸这个题怎么做？”
 - 爸爸：“问你妈妈去！”
 - 儿子：“妈妈这个题怎么做？”
 - 妈妈：“问你爸爸去！”
 - ...

在函数的定义中，内部又直接或间接的调用自己，这样的函数称为递归函数。

递归函数必须有终止条件，否则不停的进行下去，造成栈溢出



◆ 自定义函数

➤ 函数类型 函数名（参数） {
 • 函数体;
➤ }

函数类型:

Void: 无返回值。

Int: 有返回值

◆两种形式：

- 带有返回值的递归函数（调用结束返回需要的值）：

```
-int dfs(){  
    • dfs();  
-}
```

- 没有返回值的递归函数（完成某个功能）

```
-Void dfs(){  
    • dfs();  
-}
```


实现递归的关键：

- 1.确定递归公式（关系）
- 2.确定边界(终止)条件

当递归没有到达边界终止时，继续向前，直至边界才返回。

◆ 带有返回值的递归函数

➤ 递归函数结束后返回一个具体的数值。

➤ 【例1】猴子吃桃问题

➤ 小猴摘了很多桃子。

➤ 第一天吃了一半又多吃一个；

➤ 第二天又吃掉一半再多吃一个；

➤

➤ 如此下去，到第真十天恰好还剩一个桃子。

➤ 问第一天小猴摘了多少桃子？



- $f(i)$ 为第 i 天剩下的桃子数量， $f(i+1)$ 为第 $i+1$ 天剩下桃子的数量，则有关系：
- $f(i+1)=f(i)/2-1$
- 得到递推关系式： $f(i)=2(f(i+1)+1)$
- 边界条件为： $i=10$ 时， $f(10)=1$.

◆ 递归 = 递推 + 回归

◆ 【例2】 1755:斐波那契数列

- 递推关系: $f(i)=f(i-1)+f(i-2)$;
- 边界条件: $f(1)=1, f(2)=1$.
- 思考为什么需要两个边界条件?



◆ 【例3】 3089:爬楼梯



◆ 【例4】 1788:Pell数列

➤ 【例4】7592:最大公约数

➤ 输入a和b，输出a和b的最大公约数。

➤ 如：

➤ 输入：100 75

➤ 输出：25

◆欧几里德算法（ 又称辗转相除法 ）

- 用于计算两个正整数 a ， b 的最大公约数。
- 一般把 a 和 b 的最大公约数记为 $\gcd(a, b)$ 。
- 公式：
 - $\gcd(a, b) = \gcd(b, a \bmod b)$
 - $\gcd(a, 0) = a$
- 如： $\gcd(100, 75) = \gcd(75, 25) = \gcd(25, 0) = 25$;

- 递归的层数是有限的。
- 可以把递归改为递推



◆【例5】 666:放苹果

- 定义： $f(i,j)$ 为 i 个苹果放到 j 个盘子的放法，分为两种方案：
- 方案1：保证每个盘子至少有1个苹果的方案：先拿出 j 个苹果，每个盘子放一个，保证每个盘子里有一个苹果，然后把剩下的 $i-j$ 个苹果再随意放到 j 个盘子里了里，方案数是 $f(i-j,j)$;
- 方案2：保证至少有一个盘子是空的：任选其中一个盘子一个都不放（空盘子），然后报 i 个苹果放到其余的 $j-1$ 个盘子了，方案是 $f(i,j-1)$.
- 按分类的加法原理，所以总的方案是： $f(i,j)=f(i-j,j)+f(i,j-1)$
- 关键的边界条件：
- $f(i,j)=0$ ($i<0$)
- $f(i,j)=1$ ($j=1$)
- 注意： $i=0$ ， $j=1$ 时 $f(i,j)=1$;




无返回值的递归函数

void



◆ 【例1】利用递归实现输出1到10



```
➤ #include<iostream>
➤ #include<cstdio>
➤ using namespace std;
➤ void dfs(int i){
➤     if(i>10) return;
➤     cout<<i<<" ";
➤     dfs(i+1);
➤ }
➤ int main(){
➤     int n;
➤     dfs(1);
➤     return 0;
➤ }
```




◆如何实现输出10到1？

➤ (2) 阅读下列程序的输出结果:

➤ #include<iostream>

➤ #include<cstdio>

➤ using namespace std;

➤ void dfs(int i){

➤ if(i>10) return;

➤ cout<<i<<" ";

➤ dfs(i+1);

➤ cout<<i<<" ";

➤ }

➤ int main(){

➤ int n;

➤ dfs(1);

➤ return 0;

➤ }

➤ (3) 阅读下列程序的输出结果:

```
➤ #include<iostream>
➤ #include<cstdio>
➤ using namespace std;
➤ void dfs(int i){
➤     if(i>4) return;
➤     dfs(i+1);
➤     cout<<i<<" ";
➤     dfs(i+1);
➤ }
➤ int main(){
➤     int n;
➤     dfs(1);
➤     return 0;
➤ }
```

➤ (4) 阅读下列程序的输出结果:

➤ #include<iostream>

➤ #include<cstdio>

➤ using namespace std;

➤ void dfs(int i){

➤ if(i<=0) return;

➤ dfs(i-1);

➤ cout<<i<<" ";

➤ dfs(i-1);

➤ }

➤ int main(){

➤ int n;

➤ dfs(3);

➤ return 0;

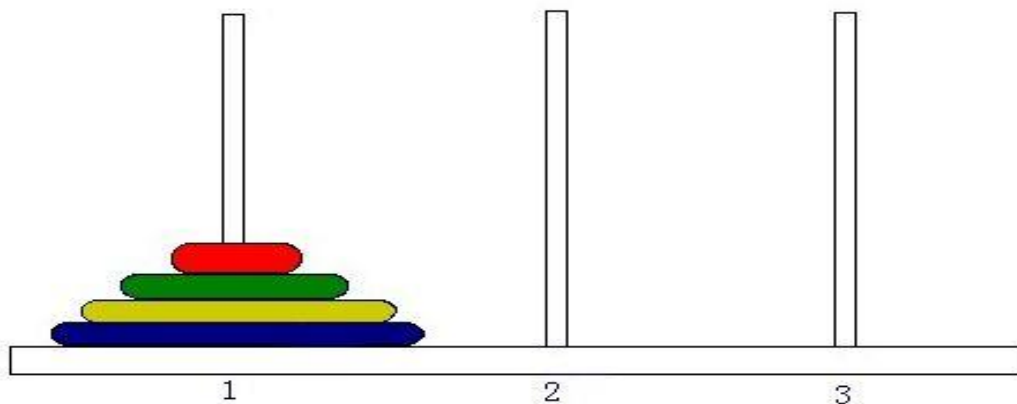
➤ }

◆【例2】输入n，输出n的二进制

- 输入n，输出n的二进制表示。
- 输入： 10
- 输出： 1010

◆ 【例3】 Hanoi (汉诺塔) 问题

- Hanoi塔由 n 个大小不同的圆盘和3根木柱1,2,3组成。开始时，这 n 个圆盘由大到小依次套在1柱上，如图所示。
- 现在要求用最少的移动次数把1柱上 n 个圆盘按下述规则移到3柱上：
- (1) 一次只能移一个圆盘；
- (2) 圆盘只能在3个柱上存放；
- (3) 在移动过程中，不允许大盘压小盘。
- 请编程描述移动的过程。
- www.543.cn/play/265599



➤ 输入:

➤ 3

➤ 输出:

➤ 1 : 1-->3

➤ 2 : 1-->2

➤ 1 : 3-->2

➤ 3 : 1-->3

➤ 1 : 2-->1

➤ 2 : 2-->3

➤ 1 : 1-->3

第一步：先借助**3**柱把**1**柱上面的 **$n-1$** 个盘子移动到**2**柱上。

第二步：然后再把**1**柱最下面的一个盘子移动到**3**柱上。

第三步：再借助**1**柱把**2**柱上的 **$n-1$** 个盘子移动到**3**上。

$\text{dfs}(n,a,b,c)$