 递归算法的执行过程分递推和回归两个阶段。在递推阶段，把较复杂的问题（规模为n）的求解推到比原问题简单一些的问题（规模小于n）的求解。例如上例中，求解fib(n)，把它推到求解fib(n-1)和fib(n-2)。也就是说，为计算fib(n)，必须先计算fib(n-1)和fib(n-2)，而计算fib(n-1)和fib(n-2)，又必须先计算fib(n-3)和fib(n-4)。依次类推，直至计算fib(1)和fib(0)，分别能立即得到结果1和0。在递推阶段，必须要有终止递归的情况。例如在函数fib中，当n为1和0的情况。 在回归阶段，当获得最简单情况的解后，逐级返回，依次得到稍复杂问题的解，例如得到fib(1)和fib(0)后，返回得到fib(2)的结果，……，在得到了fib(n-1)和fib(n-2)的结果后，返回得到fib(n)的结果。 在编写递归函数时要注意，函数中的局部变量和参数知识局限于当前调用层，当递推进入“ 简单问题 ”层时，原来层次上的参数和局部变量便被隐蔽起来。在一系列“ 简单问题 ”层，它们各有自己的参数和局部变量。 由于递归引起一系列的函数调用，并且可能会有一系列的重复计算，递归算法的执行效率相对较低。当某个递归算法能较方便地转换成递推算法时，通常按递推算法编写程序。例如上例计算斐波那契数列的第n项的函数fib(n)应采用递推算法，即从斐波那契数列的前两项出发，逐次由前两项计算出下一项，直至计算出要求的第n项。 【问题】 组合问题 问题描述：找出从自然数1、2、……、n中任取r个数的所有组合。例如n=5，r=3的所有组

合为：（1）5、4、3（2）5、4、2（3）5、4、1（4）5、3、2（5）5、3、1（6）5、2、1（7）4、3、2（8）4、3、1（9）4、2、1（10）3、2、1 分析所列的10个组合，可以采用这样的递归 思想 来考虑求组合函数的算法。设函数为void comb(int m,int k)为找出从自然数1、2、……、m中任取k个数的所有组合。当组合的第一个数字选定时，其后的数字是从余下的m-1个数中取k-1数的组合。这就将求m个数中取k个数的组合问题转化成求m-1个数中取k-1个数的组合问题。设函数引入工作数组a[ ]存放求出的组合的数字，约定函数将确定的k个数字组合的第一个数字放在a[k]中，当一个组合求出后，才将a[ ]中的一个组合输出。第一个数可以是m、m-1、……、k，函数将确定组合的第一个数字放入数组后，有两种可能的选择，因还未去顶组合的其余元素，继续递归去确定；或因已确定了组合的全部元素，输出这个组合。细节见以下程序中的函数comb。【程序】# include # define MAXN 100 int a[MAXN]. void comb(int m,int k) { int i,j. for (i=m.i>=k.i--) { a[k]=i. if (k>1) comb(i-1,k-1). else { for (j=a[0].j>0.j--) printf(“ M ”,a[j]). printf(“ ”). } } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com