电子科技大学

递归与分治递归

例 1

打印n个数的全排列

比如n=3,则这3个数1,2,3的全排列有：

123
132
213
231
312
321

算法思想

- 假设有n个数据a[n],要把这n个数据的全排列都打印出来，考虑递归方法PrintPermutation。
- 1. 第一个位置的元素可能是这n个元素的任意一个，可以采用swap，将第一个位置与其他位置的交换，就可以得到第一个位置元素不同的情况，而余下的递归调用打印全排列函数PrintPermutation完成。但要注意还应该交换回去，便于后面的正确处理。
- 2. 递归函数内部，又会当前的第一个位置（第2个位置）的元素与余下n-2个位置元素交换；里面又会当前的第一个位置（第3个位置）的元素与余下n-3个位置元素交换；……；最后第n个位置的元素，就表示前面的n-1个位置已经处理了，就可以打印了。所以递归打印条件是位置k=n，这个也是递归出口

```cpp
void PrintPermutation(int a[], int k, int n){
    if (k == n){
        for (int i = 1; i < n + 1; ++i){
            cout << a[i] << " ";
        }
        cout << endl;
        return;
    }
    else{
        for (int i = k; i < n+1; ++i){
            Swap(a[k], a[i]);
            PrintPermutation(a, k + 1, n);
            Swap(a[k], a[i]);
        }
    }
}
```

```cpp
int main()
{
    int *a, len;
    cout << "len:" << endl;
    cin >> len;
    a = new int[len+1];
    if (!a)return -1;
    for (int i = 1; i < len + 1; ++i)
a[i] = i;
   PrintPermutation(a, 1, len);
}
```

电子科技大学

性能分析

- $T(n)=nT(n-1)+cn$
- $=n*[(n-1)*T(n-2)+c(n-1)]+cn$
- $=n*(n-1)*T(n-2)+cn(n-1+1)$
- $=......$
- $=n!T(1)+c(n+n*(n-1)+n*(n-1)(n-2)+....+n!)$
- $=O(n!)$

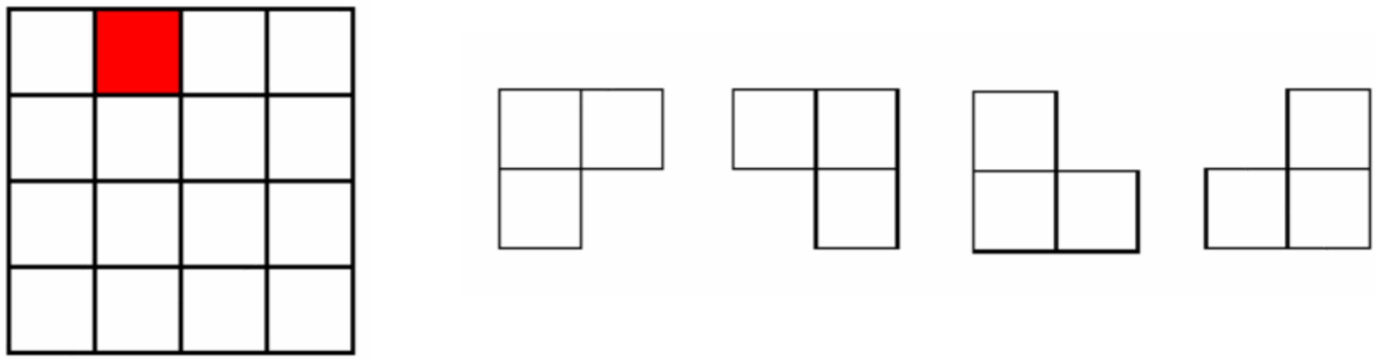
例 2

棋盘覆盖

电子科技大学

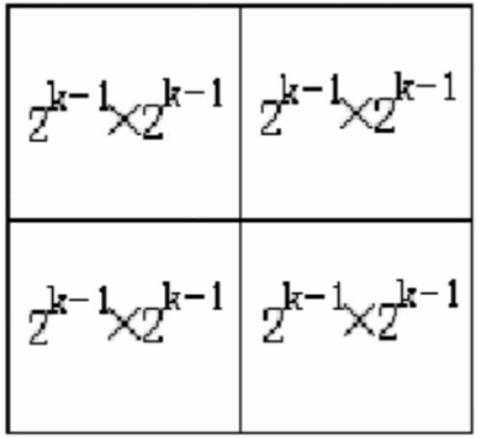
棋盘覆盖

在一个$2^k \times 2^k$个方格组成的棋盘中，恰有一个方格与其他方格不同，称该方格为一特殊方格，且称该棋盘为一特殊棋盘。在棋盘覆盖问题中，要用图示的4种不同形态的L型骨牌覆盖给定的特殊棋盘上除特殊方格以外的所有方格，且任何2个L型骨牌不得重叠覆盖。

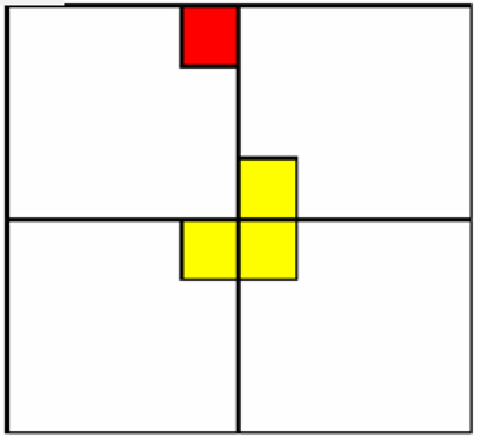


棋盘覆盖

当k>0时，将$2^k \times 2^k$棋盘分割为4个$2^{k-1} \times 2^{k-1}$ 子棋盘(a)所示。特殊方格必位于4个较小子棋盘之一中，其余3个子棋盘中无特殊方格。为了将这3个无特殊方格的子棋盘转化为特殊棋盘，可以用一个L型骨牌覆盖这3个较小棋盘的会合处，如 (b)所示，从而将原问题转化为4个较小规模的棋盘覆盖问题。递归地使用这种分割，直至棋盘简化为棋盘 4*4



(a)



(b)

性能分析

- $T(2^k*2^k)=4T(2^{k-1}*2^{k-1})+C$

令$m=2^k, n=m^2$

$T(n)=4T(m*m/4)+C$

$=4T(n/4)+C$

$=4[4T(n/4^2)+C]+C$

$=4^2T(n/4^2))+4C+C$

$=......$

$=4^tT(n/4^t))+(4^{t-1}+4^{t-2}+...+1)C$

$T(n/4^t)=T(1)$

$n/4^t=1$

$t=\log_4 n$

所以

$T(n)=4^tT(1)+[(4^t-1)/(4-1)]C$

$=O(n)$