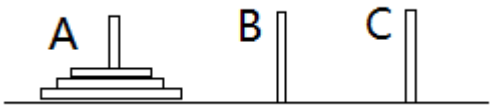
第三章练习

打*者可选作。

一、 讲义练习三：1*、2*、3(讲义有)、4*。

二、

1. 改进插入排序算法(第三章 ppt No.6)，在插入元素 a(i)时使用二分查找代替顺序查找，将这个算法记做 ModInsertSort，估计算法在最坏情况下的时间复杂度。
2. 设 A 是 n 个非 0 实数构成的数组，设计一个算法重新排列数组中的数，使得负数都排在正数前面，要求算法复杂度为 O(n)。
3. Hanoi 塔问题。图中有 A、B、C 三根柱子，在 A 柱上放着 n 个圆盘，其中小圆盘放在大圆盘的上边。从 A 柱将这些圆盘移到 C 柱上去，在移动和放置时允许使用 B 柱，但不能把大盘放到小盘的下面。设计算法解决此问题，分析算法复杂度。



4.给定含有 n 个不同数的数组 $L=\{x_1,x_2,\ldots,x_n\}$，如果 L 中存在 x_i，使得 $x_1<x_2<\ldots<x_{i-1}<x_i>x_{i+1}>\ldots>x_n$，则称 L 是单峰的，并称 x_i 是 L 的峰顶。假设 L 是单峰的，设计一个优于 O(n)的算法找到 L 的峰顶。

5.设 A 是 n 个不同的排好序的数组，给定数 L 和 U，L<U，设计一个优于 O(n)的算法，找到 A 中满足 L<x<U 的所有数 x。

6*.在 n(n>=3)枚硬币中有一枚重量不合格的硬币(过轻或过重)，如果

只有一架天平可以用来称重且称重的硬币数没有限制，设计一个算法，找出这枚不合格的硬币，使得称重的次数最少(优于 O(n))。

(提示：分成 n/3 或 n/4 份，至少两份数量相等)

7.设 $A=\{a_1,a_2,\ldots,a_n\}$,$B=\{b_1,b_2,\ldots,b_m\}$是整数集合，其中 m=O(logn)，设计一个优于 O(nm)的算法找出集合 C=A∩B。

8. 设 S 是 n 个不等的正整数的集合，n 为偶数，给出一个算法将 S 划分为子集 S1 和 S2，使得|S1|=|S2|且 $\left|\sum_{x\in S_1} x-\sum_{x\in S_2} x\right|$ 达到最大，即两个子集元素之和的差达到最大。(要求：T(n)=O(n))。

9.考虑第三章 PPT NO.17 Select(A,k)算法：

（1）如果初始元素分组 r=3，算法的时间复杂度如何？

(2) 如果初始元素分组 r=7，算法的时间复杂度如何？

10*．在 Internet 上的搜索引擎经常需要对信息进行比较，比如可以通过某个人对一些事物的排名来估计他对各种不同信息的兴趣。对于不同的排名结果可以用逆序来评价他们之间的差异。考虑 1,2,…,n 的排列 $i_1,i_2,\ldots,i_n$，如果其中存在 i_j，i_k，使得 j<k 但 $i_j>i_k$，那么就称(i_j,i_k)是这个排列的一个逆序。一个排列含有逆序的个数称为这个排列的逆序数。例如：排列 2 6 3 4 5 1 含有 8 个逆序：(2,1)，(6,3)，(6,4)，(6,5)，(6,1)，(3,1)，(4,1)，(5,1)，它的逆序数就是 8。一个由 1,2,…,n 组成的所有 n!个排列中，最小的逆序数是 0，对应的排列是 1 2 3 4…n，最大的逆序数是 n(n-1)/2，对应的排列是 n n-1 … 2 1。逆序数越大的排列与原始排列的差异度越大。

利用二分归并排序算法设计一个计数给定排列逆序数的分治算

法，并对算法的时间复杂度进行分析。

11.对玻璃瓶做强度试验，设地面高度为 0，从 0 向上有 n 个高度，记为 1,2,…,n，其中任何两个高度之间的距离都相等。如果一个玻璃瓶从高度 i 落到地上没有摔碎，但从高度 i+1 落到地上摔碎了，那么就将玻璃瓶的强度记为 i。

(1)假设每种玻璃瓶只有 1 个测试样品，设计算法来测试出每种玻璃瓶的强度。以测试次数作为算法的时间复杂度量度，估计算法的复杂度。

(2)假设每种玻璃瓶有足够多的相同的测试样品，设计算法使用最少的测试次数来完成测试。

(3)假设每种玻璃瓶只有 2 个相同的测试样品，设计次数尽可能少的算法完成测试。

12．使用主定理求解以下递归方程：

(1) $\begin{cases} T(n)=9T(n/3)+n \\ T(1)=1 \end{cases}$　　(2) $\begin{cases} T(n)=5T(n/2)+(n\log n)^2 \\ T(1)=1 \end{cases}$

(3) $\begin{cases} T(n)=2T(n/2)+n^2\log n \\ T(1)=1 \end{cases}$

13．使用递归树求解：$\begin{cases} T(n)=T(n/2)+T(n/4)+cn \\ T(1)=1 \end{cases}$

14．使用迭代递归法求解：

(1) $\begin{cases} T(n)=T(n-1)+\log 3^n \\ T\ (1)=1 \end{cases}$　　(2) $\begin{cases} \mathrm{T}(n)=T(n-1)+1/n \\ T(1)=1 \end{cases}$