保镖

考虑把不在 A 中的数字给插入 A 中，i 最优的插入位置是 A 中第一个大于 i 的位置之前，同时插入的数字之间的相对顺序肯定是递增的。

从前到后扫描 A，每一次把小于 A_i 的还没有插入的数字插到 A_i 之前。时间复杂度 $O(n)$。

矿石

把所有采矿点升序排序，令 A_i 表示第 i 个采矿点能采到的矿石种类数，B_i 表示第 i 个采矿点第一次能采到的矿石种类数。

考虑利用最小表示法计数，即对于每一个矿石集合 S，只在第一次能采到这个集合的采矿点计入答案。则在第 i 个采矿点计入的答案中，必须要要有至少一个 B_i 中的矿石，共有 $(2^{B_i}-1)\times 2^{A_i}$，求和即可。

A_i, B_i 都可以在 $O(n\log n)$ 的时间内计算得到，因此总的时间复杂度为 $O(n\log n)$。

括号序列

先考虑判断一个串是否存在一个合法的转化方案（下面简称合法）。可以用一个栈维护当前还没有被匹配的字符，从左往右扫描字符串，如果当前字符和栈顶元素相同，则弹栈，否则把当前字符压入栈中。这个串合法的充要条件是最后栈为空。

这样可以在 $O(n)$ 的时间内判断以 i 开头的所有字符串中有多少个是合法的，总的判断时间复杂度为 $O(n^2)$.

现在考虑把复杂度降下来。我们不从每一个位置 i 开始扫字符串，就只扫描整个字符串一遍。这样我们得到了 $n+1$ 个栈 s_0 至 s_n。一个结论是区间 $[l,r]$ 是合法的当前仅当 $s_{l-1}=s_r$。

利用这个结论，我们可以求出 s_0 至 s_n 的哈希值（因为单次的操作只有弹栈和插入一个元素，所以可以很方便地维护 Hash 值），然后排序求其中值相等的无序对数即可。时间复杂度为 $O(n\log n)$。

关于上面的结论，可以对 s_{l-1} 和 s_r 的最长公共前缀长度归纳。当最长公共前缀长度为 0 的时候显然成立。

现在考虑消去 s_{l-1} 和 s_r 的第一位元素。如果他们不同，假设分别是 a 和 b，则这个串一定形如 $S_1aS_2aS_3bS$，其中 S_1,S_2,S_3 均为合法串，$l-1$ 的位置在 S_1 中，r 的位置在 S 中。此时如果从 l 开始用栈扫描串的话，这个 b 一定消除不掉。

假设他们不同，如果第一位元素在原串中的位置相同，显然这一位可以忽略，如果位置不同，则串形如 $S_1aS_2aS_3aS$，同样 S_1,S_2,S_3 均为合法串，$l-1$ 的位置在 S_1 中，r 的位置在 S 中。因此从 l 开始扫的话两个 a 会互相匹配掉，因此只需要考虑第二个 a 后的部分，这是最长公共前缀长度减一的情况。

因此归纳成立。