Solution

跳跃(jump)

签，这个做不出来的该谢罪了。

对于一个 x，它跳到的位置就是左边第一个比他大的元素。使用一个栈即可简单求出。然后直接从左往右递推一下就行。

时间复杂度 $O(N)$，期望得分 100 分。

最近公共祖先(lca)

固定根节点时如何求出所有 lca 的和是简单的。

我们可以直接统计每一个点 x 作为 lca 的次数，$lca(i,j)=x$，当且仅当 i,j 来自 x 的两个不同子树，具体地说

$$cnt_x = siz_x^2 - \sum_{y\in son_x} siz_y^2$$

显然可以 $O(n)$ 求出。

对于不同的根节点统计起来也并不困难，具体地说，对于 x 来说，不同的 cnt_x 只有 deg_x 种。只取决于根节点在 x 的哪一棵子树内。

而对于每一种情况，上面给出的式子中不同的只有 siz_x 以及一个 siz_y。显然是可以直接枚举根节点所在的子树直接计算的。而不同的子树内的根节点在 dfs 序上是连续的，差分之后做加法最后前缀和即可。

时间复杂度 $O(N)$。

翻转 (flip)

将一次操作看成一个区间，那么区间之间只能相离或者包含，并且是前面的区间包含后面的区间。

考虑从前往后加入这些区间，首先考虑区间之间的相对关系而不考虑区间的大小。

插入第 i 个区间时，前 $i-1$ 个区间会产生 $2i-1$ 个可插入的位置，具体的说，一共 $2i-2$ 个端点，每两个相邻的端点之间都是可以插入的，同时算上头尾，一共 $2i-1$ 种插入的方式。

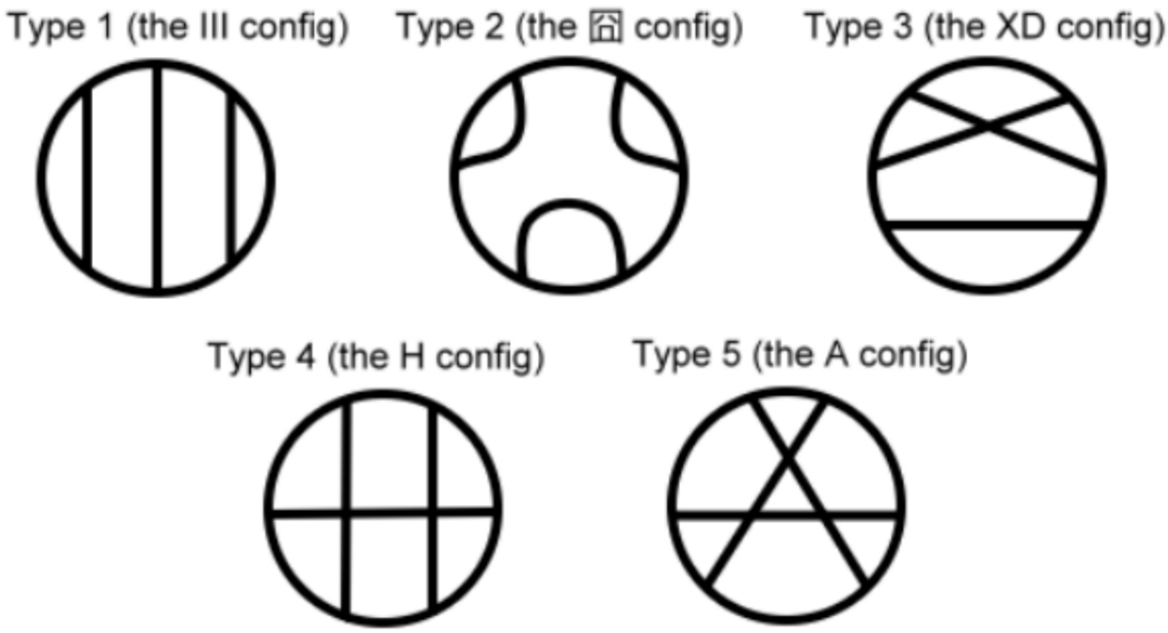
插入的总方案数为:$(2k-1)!! = (2k-1)\times(2k-3)\ldots\times 1$。

接着考虑安排这些区间的长度。考虑最短的，使得这些位置关系合法的 01 交替串。恰好使用了 k 个 1 和 $k-1$ 个 0。(插入到区间内会使得原来的 1 变成 101，不插入区间内则会增加一组 10，如果时最后一个右端点，那么最后那个 0 就是不需要的)。

最后我们把剩下的 $n-k$ 对 01 任意插入到形成的 $2k+1$ 段中。根据插板法得到方案数为 $\binom{n+k}{2k}$。

答案就是 $\binom{n+k}{2k}(2k-1)!!$

探险家(explorer)



这个题目已经老到我清晰的记得我当初做过的程度了。

考虑环上的三条弦相交只可能是这五种情况之一，容易发现 Type 2 和 Type 5 符合题意。

考虑容斥，即用 $\binom{n}{3}$ 来减掉不合法的方案数。对于 Type 1 ，枚举中间的弦，计算在它左边的弦的数量 l_i 和在它右边的弦的数量 r_i 的乘积，这部分的答案等于 $\sum_{i=1}^{n} l_i r_i$。

对于 Type 3 和 Type 4，它们的共同点是，恰有两条弦满足**「和一条弦不交，和另外一条弦相交」**。显然和一条弦相交的弦有 $n-1-(l_i+r_i)$ 条。注意到一种方案中有两条弦满足这个性质，于是这部分的答案等于 $\frac{1}{2}\sum_{i=1}^{n}(l_i+r_i)(n-1-(l_i+r_i))$。

接下来考虑怎么计算 l_i 和 r_i。不妨给环上的点逆时针标号 1到 n。手玩下可以知道，对于弦 $(x,y)(x<y)$，一条弦 $(a,b)(a<b)$ 在它左侧当且仅当满足下列条件之一：

- $a,b<x$
- $a,b>y$
- $a<x,b>y$

同理在它右侧的弦满足 $x<a<b<y$。

注意到两部分都仅仅是一个二维偏序问题，树状数组计算即可。

> 本质上，考虑在序列上的区间关系，确定 (x,y) 之后，所谓的左边，右边，一定指的是包含，被包含以及相离两种情况。
>
> 要么包含代表左边被包含以及相离代表右边，要么被包含以及相离代表左边，包含代表右边。
>
> 包含，被包含，相离三种相对位置关系都是比较好求得。并且情况非常简洁。这也是为什么本题采用了容斥的手段。
>
> 对于Type 2 和 Type5 的两种情况，枚举其中一个区间之后，对于另外两个区间关系有着限制。
>
> 例如，Type 2 中枚举一个区间之后，剩下两个区间要么是该区间内部两个无交的子区间，要么是三个互相无交的区间。
>
> 而在 Type5 之中，关系则更为复杂，枚举一个区间之后，另外两个区间均与该区间有交的同时，这两个区间也必须是相交关系。

手电筒(flash)

题意为给定一个正整数数列，通过划分为若干个长度不超过 L 的段，使每段权值和最小。一段权值为将最后一个数减一后，本段的最大值加一。同时需要支持 q 次互不影响的修改。

对于原问题显然有 $\mathcal{O}(nL)$ 的 DP。记 f_i 为最后一段结尾是 i 的最小权值和。时间复杂度 $\mathcal{O}(qnL)$。

- 发现每一次都重新 DP 非常不优。我们记录 f_i 表示 $[1,i]$ 分段的最小值。 g_i 表示 $[i,n]$ 分段的最小值。那么只需要每次对 $[x,x+L-1]$ 这一段跑类似 DP 即可。时间复杂度 $\mathcal{O}(nL+qL^2)$。
- 由于 f 不降，那么当 j 和 $j-1$ 都能贡献到 i ，且 $v_{j,i}=v_{j-1,i}$ 时，用 $j-1$ 一定不劣于用 j 贡献。其中 $v_{j,i}$ 表示将 $[j,i]$ 单独分为一段的权值。同理，当有一段能贡献且增量一致时，只需考虑第一个。那么维护这些点，查询其中最小值即可。修改增量可以把最后几段进行合并，用线段树维护双端队列完成。总复杂度 $\mathcal{O}(n\log n)$。

将两个优化结合起来，复杂度 $\mathcal{O}(n\log n+qL\log L)$.

std 使用的循环队列优化，时间复杂度 $\mathcal{O}((n+qL)\log L)$。