三角形

对于 30%的数据：

对每次询问暴力枚举选哪三条边。

复杂度 O(n^3q)。

对于 60%的数据：

可以发现，对于每次询问，选取长度相邻的三条边一定最优。

证明：设$\{A_i\}$有序，若A_x, A_y, A_z是一种合法的选取方案，则A_{z-2}, A_{z-1}, A_z一定合法且更优。

因此对于每次询问我们可以将序列排序，枚举选连续的三条边。

复杂度 O($n\log n\, q$)。

对于 80%的数据：

我们可以发现，对于有序序列$\{A_i\}$，不存在合法的选边方案当且仅当$A_{i+2} > A_{i+1} + A_i$。

通过打表可以发现，当$A_i \le 10^9$时，不合法的序列长度不会超过 50，即连续的 50 个数中一定存在合法的选边方案。

因此对于每次询问，我们只需要考虑前 50 大的数即可。

带修区间第 k 大可以通过树状数组套权值线段树等维护。

复杂度 O($50\, log^2\, n\, q$)。

对于 100%的数据：

显然区间前 50 大可以通过线段树来维护。

线段树上每个点维护对应区间前 50 大，合并时取左右儿子的前 50 大即可。

复杂度 $O(50\log n\, q)$。

动态树

注：本题解中使用的求和符号Σ均为为异或和

对于 20%的数据：

对于每一时刻的每一个点进行一遍 dfs，统计子树异或和。

复杂度 $O(n^3)$。

对于 40%的数据：

设$p_{i,j}$代表第 i 个时刻 j 点的权值。

显然$p_{i,j}$=$p_{i-1,j}$^$\sum_{k}^{k\in \mathrm{son(j)}} p_{i,k}$。

复杂度 $\mathrm{O}(n^2)$。

对于 60%的数据：

对于同一深度的点，同一时刻对答案的贡献次数（即答案异或上该点权值的次数）相同。

设 $f_{i,j}$代表时刻 i 深度为 j 的点对答案的贡献次数，ans_i代表时刻 i 的答案，h_i表示所有深度为 i 的点的异或和。

则有$f_{i,j} = f_{i-1,j} + f_{i,j-1}$，$\mathrm{ans}_i$=$\sum_{j=0}^{dep} (f_{i,j}\&1) * h_j$。

复杂度 $\mathrm{O}(dep^2)$。

对于 80%的数据：

我们发现，$f_{i,j}$的递推式和组合数很像，通过打表可以发现$f_{i,j} = C_{i+j-1}^{\mathrm{i}-1}$~~(我只能想到这么证明了。。。。)~~。

由于我们只关心$f_{i,j}$的奇偶，这里直接给出结论：C_{i+j}^{i}为奇数当且仅当 i&j=0~~（具体证明较长，这里实在懒得写了。。。。）~~。

因此可以得出，时刻 i 时，深度为 j 的点对答案有贡献（贡献次数为奇数）当且仅当(i-1)&j=0。

即ans_i=$\sum_{j}^{j\&(i-1)=0} h_j$，j 可以通过枚举 i-1 补集的子集来枚举。

复杂度 $\mathrm{O}(3^{\log_2 n})$。

对于 100%的数据：

我们发现，80 分算法的瓶颈在于枚举子集，考虑优化。

设 $\mathrm{dp}_{i,j} = \sum_{k}^{k\text{的前 j 位是 i 前 j 位的子集，其余与 i 相同}} h_k$，S= $\log_2 dep$ 。~~（我知道我这个定义又丑又笨）~~

显然$\mathrm{dp}_{i,0}$=h_i 。考虑 i 的第 j 为为 0 或 1，则有$\mathrm{dp}_{i,j} = \mathrm{dp}_{i,j-1}$+[i&(1<<j)]* $\mathrm{dp}_{i\text{^}(1\ll j),j-1}$，

$\mathrm{ans}_i=\mathrm{dp}_{(1\ll s)-i,s}$。

复杂度 $\mathrm{O}(n\log n)$

或与异或

对于 30%的数据

暴力枚举选哪几个数。

复杂度 $\mathrm{O}(2^n n)$

对于 60%的数据

考虑 dp 转移。

设 $f_{i,j,k}$代表前 i 个数，异或起来为 j，或起来为 k 的方案数。

则有$f_{i-1,j\hat{}\boldsymbol{a_i},k|\boldsymbol{a_i}}= f_{i-1,j,k}+f_{i-1,j\hat{}\boldsymbol{a_i},k|\boldsymbol{a_i}}$，ans=$\sum_{i=1}^{a} f_{n,i,i}$。

复杂度 $\mathrm{O}(a^2n)$

对于 100%的数据

考虑优化，我们发现，当某一位异或起来为 1 时，或起来一定也为 1。

因此我们可以稍稍修改一下 f 数组的定义，

设 $f_{i,j,k}$为前 i 个数，异或起来为 j，或起来为 j|k 的方案数，并且满足 j&k=0，即 k 是或中扣掉了异或的其他部分。

对于每一个 j，转移时我们枚举 j 补集的子集 k。

转移$f_{i,j\hat{}\boldsymbol{a_i},k|j|\boldsymbol{a_i}\hat{}\boldsymbol{j}\hat{}\boldsymbol{a_i}}= f_{i-1,j,k}+f_{i-1,j\hat{}\boldsymbol{a_i},k|j|\boldsymbol{a_i}\hat{}\boldsymbol{j}\hat{}\boldsymbol{a_i}}$，ans=$\sum_{i=1}^{a} f_{n,i,0}$。

复杂度 $\mathrm{O}(3^{\log a}n)$