# 矩阵

**算法一**

由于题面已经给了一个矩阵乘法的公式，直接把 $C$ 矩阵计算出来，记一个前缀和即可。

时间复杂度 $O(n^3)$，可以得到 30 分。

**算法二**

题目中给了 $m$ 特别小的数据点。虽然不知道如何处理，但是显然可以只处理出询问的位置，没有询问的位置不处理。期望得分 30 分，但由于出题人很良心，实际得分 50 分。

**算法三**

我们考虑对于每个询问要求的是什么东西(假设 $a \le c, b \le d$)：

$$\sum_{i=a}^{c}\sum_{j=b}^{d}\sum_{k=1}^{n} A_{i,k} * B_{k,j}$$

里面那个求和显然可以提出来，就变成了：

$$\sum_{k=1}^{n}\sum_{i=a}^{c}\sum_{j=b}^{d} A_{i,k} * B_{k,j}$$

由于乘法分配律，可以化为：

$$\sum_{k=1}^{n}\sum_{i=a}^{c} A_{i,k} * \sum_{j=b}^{d} B_{k,j}$$

于是这个式子只需要对 $A$ 记一个行的前缀和，对 $B$ 记一个列的前缀和即可。

时间复杂度 $O(nm)$，可以得到 100 分。

# 序列

**算法一**

记录前缀和，暴力求出每个区间的和，然后 sort 一遍，第 $k$ 个即为答案。

时间复杂度 $O(n^2 \log n)$，可以得到 30 分。

**算法二**

注意到数据中有 $k = 1$ 的点，直接求最大子段和即可。

**算法三**

由于序列的所有连续子段和数量高达$10^{10}$级别，我们考虑如何在不求出所有子段和的前提下快速求出第 $k$ 大的值。

显然为了快速得到答案，我们首先想到的是二分答案。并且很快可以发现，答案满足单调性，因为当答案变大时，那么大于它的子段和数量将会变少。因此我们需要一个数据结构来维护我们快速查询对于当前值有多少个子段和比它大。

首先我们考虑一个连续区间$[l,r]$的和我们可以表示成 $S[r] - S[l-1]$($S$ 表示前缀和)。满足条件的区间会满足 $S[r] - S[l-1] \ge mid$，转化一下变成 $S[r] - mid \ge S[l-1]$。也就是说我可以从左往右扫一遍这个序列，然后对于每一个前缀和 $S[i]$我都去查询比 $S[i] - mid$ 小的

数有多少个。这个显然可以用树状数组维护。

所以我们的做法就是先把这个序列去重，然后对于每次 check 都扫一遍整个序列，每次查询并且插入一个新的前缀和元素。有一些细节需要注意，比如说初始我们需要插入一个 0，因为前缀和可以减去 0 表示区间从 1 开始，诸如此类的细节，不再赘述。

# 二分图

## 算法一

可以考虑对于每个时刻构一张图，然后跑一遍二分图染色 $check$ 一下即可。

时间复杂度 $O(T(n+m))$，可以得到 30 分。

## 算法二

这道好像没有什么部分分 TAT……于是接下来的就是 AC 算法了……

首先，我们考虑怎么判断一张图是否是二分图。于是，现在我们有一个不是那么明显的结论：一张图是二分图的充要条件是这张图没有点数为奇数的环。

然后，让我们感性地来想一想这个结论。一张图如果不是二分图，那么肯定是某些点的颜色冲了。而要让一个点的颜色冲突，必定是这个点往外走了一圈之后又回到了这个点。而只有奇数个点的环才可以是的限制这个点的颜色和原来的颜色不一样。所以二分图必定没有奇数个点的环。

然后，由于有多个询问，所以我们可以考虑对时间分治，写一个整体二分。发现 CDQ 分治是一个类似于线段树的东西，于是我们可以用在线段树上找一个区间的方法把一个区间拆成 $\log n$ 个区间，保存下来，然后就是判断有无奇环了。

判断有无奇环可以考虑有并查集判断。若$dis_{x,y}$表示 $x$ 节点到 $y$ 节点的距离，那么有：

$$dis_{x,y} = dis_{x,root} + dis_{y,root} - 2dis_{lca(x,y),root}$$

由于只需要判奇偶性，所以记录异或值即可。若$dis_{x,y}$表示路径上的权值异或和，有：

$$dis_{x,y} = dis_{x,root} \oplus dis_{\mathrm{y,root}}$$

其中⊕这个符号是异或的意思。

于是我们就可以使用带权并查集来维护每个点到根的异或和，然后查询两点间的异或和就是两个点到根的异或和异或起来。当我们把两个联通块合并时，假设把联通块 $x$ 接到联通块 $y$ 下面，那么 $x$ 这棵树中的节点到根的距离显然都需要异或 1 才能保证距离是对的，就在 $x$ 这个节点打上标记即可。由于 CDQ 分治需要用到回溯，那么并查集也需要回溯。于是并查集不能写路径压缩，要写按秩合并。也就是每次把 $size$ 小的子树接到 $size$ 大的子树下面，这样树高是不超过 $\log n$ 的。

于是接下来就是 CDQ 分治的板子辣！

空间复杂度 $O(n\log n)$，时间复杂度 $O(n\log^2 n)$。

## 算法三

我们还有另外一种 CDQ 分治的做法。由于每一条边都有一个出现时间和一个消失时间，每次递归之前就可以先把包含了左半区间的边给加进去，然后递归左边；然后再把包含了右半区间的边给加进去，递归右边。最后记得撤销操作。这样写的话空间复杂度就变为 $O(n)$ 了。