T1

发现是非空子序列，就相当于随便选啦

这不随便做?

考虑一堆字符，我们能够用它们拼出一个回文串的充要条件就是至多有一种字符出现偶数次

然后我们统计一下字符串里面每种字符的出现次数就做完啦

具体一点就是我们枚举每个字符选取奇数个或者选取偶数个的方案数(因为我们只关心奇偶情况)

注意这里是预处理!

要注意不选也算是偶数个

然后暴力枚举哪一种字符出现了奇数次

于是你就有了简单的组合数代码

但是写起来码量可能有点大(虽然全是板子就是了)

然后我们就想到了一个奇妙的结论:

在一堆不同东西里面选出任意个，选出奇数个与选出偶数个的方案数是一定相等的

T2

我们考虑枚举 $ans = i$ 然后计算这样的方案数，但是直接计算相等是不好做的，于是我们考虑计算 $ans \le i$ 的方案数。

假如说我们现在枚举到了 $ans \le i$ 了，我们考虑如何计算：我们可以把 a 序列刻画到一个平面直角坐标系中，其中 $a_i = 1$ 看作向上走，$a_i = -1$ 看作向右走，那么 $ans \le i$ 就等价于从 $(0, 0)$ 走到 (n, m) 且不经过 $y = x + i + 1$ 的方案数，我们反射容斥一下就做完了，不会反射容斥可以自己下去学习一下。

现在我们已经有了 $ans \le i$ 的方案数，想要计算 $ans = i$ 的方案数只需要拿 $ans \le i$ 的方案数减去 $ans \le i - 1$ 的方案数即可。

T3

枚举两端的两个点a, c，发现可以区间DP在n^2时间内解决。

如果固定a, b，合法的c构成了一段后缀。

发现可选点对太多了，尝试去除一些无用的点对。

当且仅当满足以下条件时，(a, b, c)严格优于(a', b', c')，(a', b', c')是无用的。

- $v_a + v_b + v_c \ge v_{a'} + v_{b'} + v_{c'}$
- $a' \le a < c \le c'$

这个条件太强了，不能有效降低复杂度。

对于(a, b)，如果存在a'满足$a < a' < b$，且$v_{a'} \ge v_a$**或**$v_{a'} \ge v_b$，可用a'替换价值较小的端点。

由此，对于每个i，分别求出L_i和R_i表示在i之前最后一个不小于v_i的位置和在i之后第一个不小于v_i的位置。可以使用单调栈。

有用的区间为所有$[L_i, i]$和$[i, R_i]$，数量只有$2n$个。

每个点记录对应的有用右端点，从后往前加入区间，将询问挂到左端点上，用线段树维护并查询答案。

T4

算法 1

对于 $n \le 100$，按照题意暴力模拟算出 f, g 即可。

然后区间个数的话外层 dp 一下就行了。

时间复杂度大概是 $\mathrm{O}(n^4)$。

算法 2

对于 $p = n^2$ 的情况，发现选择一个 l, r，直接对整个区间的平均值 dp 就是对的。

算法 3

下面设 $k = k_1 + k_2 \le 400$。

有一个观察：有用的子区间长度都为 $2/3$。~~这么简单的事情大家肯定都发现了呀。~~

证明非常简单，发现对于 f 函数，一定取区间最大值，若子区间长度 ≥ 4，一定可以把这个子区间从中间某一点断开，使得分裂至少有一个子区间不比原先平均值小（因为两边的平均值都比整个区间小的话，那么整个区间的平均值就一定会小于它自己，这显然是矛盾的）。又由于这个区间平均值已经是最大的，所以发现分裂出来的两个子区间一定也是相等的且都等于最大值。那么为了选出更多的区间，选择缩小区间的长度肯定会更优。至于为什么会有长度为 3 的，那就是因为长度为 3 的话，可能会分裂为两个长度为 $2 + 1/1 + 2$ 的，而长度为 1 的可能无法贡献，故需要考虑长度为 3。

接下来考虑 g，一定取第 k 小。事情是类似的。假设 g 最小值，虽然其是最小，但是由于我们要求的最终结果是最大值，所以每次分裂后向比较大的那一方去取就可以了，所以一部分情况是可以把 g 也缩小为长度为 2 的区间。长度为 3 的话，可能会出现其一个子区间长度为 2，但是值比 3 的小，但是我们发现保证了 $3 \le p$ ，发现我们卡着长度为 3 的取，它只有三个子区间，于是情况相同。

所以现在 f, g 本质相同。由于 $k \le 400$，直接 dp 就好了。

时间复杂度 $\mathrm{O}(nk)$。

算法 4

只有一次询问。如果沿用上面的做法，我们的做法就 T 飞了。不妨从 dp 本身的性质来考虑。

设 dp_i 表示从最终长度为 n 的序列中选了至多 i 个不交的相邻数对的最优值。

如果我们不 dp，直接贪心呢？那么显然是每次想选造成的增量最大的那一个。比较假的贪心是直接选择不交的最大区间。或许可以反悔贪心，但是仍然不好维护。

向着这个方向思考~~，再打一波表~~，你就会发现，我们每次选择造成的增量一定是不增的，否则一定可以与之前的某次选择进行交换，使得这之后的不变劣，这之前的更优。即这个 dp 函数是 凸 的。

那么又因为 $q = 1$，就可以使用 `WQS二分`。

于是复杂度为 $\mathrm{O}(nq \log n)$。

算法 5（正解）

既然 dp 是凸的，并且最后多次询问，那么我们直接做 `闵可夫斯基和` 就好了。

这个东西具体做法大概就是，先对序列分治，然后对每个区间都求出一个 dp 数组，同样是凸的。每次合并的时候去做一个背包，直接做的话复杂度 $\mathrm{O}(n^2)$。

考虑 $h_k = \max\limits_{i+j=k} f_i + g_j$。考查 $i \to i+1$ 的过程，假设这时 $h_i = f_j + g_k$，由于差分不降，那么只要贪心的选择到 f_{j+1}/g_{k+1} 就一定是对的。

实现维护 $f = \Delta dp$，和两个指针，每次贪心的往后选较大者，塞到 Δh 里面即可，最后一遍前缀和就可以求出正确的 h。

时间复杂度 $\mathrm{O}(n \log n)$。