Solution

by tsx

score

- 均匀随机两个 $1 \sim n$ 的排列 a_i, b_i，已知 $a_1 = a, b_1 = b$，请求出序列 $c_i = a_i + b_i$ 从小到大排序后，$a_1 + b_1$ 在该序列中的期望排名，也即序列 c_i 中比 c_1 小的数的个数加一。
- 对 998244353 取模。
- T 组数据，$1 \leq T \leq 1.5 \times 10^5$，$1 \leq a, b \leq n \leq 998244352$

score 题解

- 比 $a_1 + b_1$ 大的 c_i 数量的期望，可以转化为每一个 c_i 比 $a_1 + b_1$ 小的概率之和。
- 枚举 $x \neq a_1$，考虑对于 $a_u = x$ 的 u，满足 $a_u + b_u < a_1 + b_1$ 的概率。
- 此时注意到对于 b_u 仅有限制 $1 \leq b_u \leq n$, $b_u < a_1 + b_1 - x$ 以及 $b_u \neq b_1$，所以容易计算 b_u 能取到合法值的概率。
- 考虑加速这个过程，注意到当 x 每次增加 1 时，b_u 合法解的个数要么不变，要么减少 1，且可以拆成两个等差数列以及一个常数列。
- 复杂度瓶颈在于求逆元，实际上若使用离线求逆元的技巧，可以做到单次询问 $O(1)$。
- 时间复杂度 $O(T \log n)$ 或 $O(T + \log n)$。

flag

- 有 n 个地点，用 $1 \sim n$ 标号，有 $n-1$ 条无向道路将这些地点连通，也就是说它们构成了一棵树。两个地点之间的距离定义为它们最短路径经过的道路数。
- 小 T 最开始在 1 号地点。每次，小 T 会找到所有距离他不超过 k 且没有被插上旗的地点，从中选择一个离他最近的地点（如果有多个距离他最近的地点他可以任意选择其中一个），然后走到那个地点插上旗子。
- 小 T 需要将这 n 个地点都插上旗子，且他最后一个插旗的位置到 1 号地点的距离不能超过 k。 现在小 T 还没有确定参数 k 的取值，请你求出最小的正整数 k，使得存在一个满足上述条件的插旗方案。
- T 组数据，$1 \le T \le 10^4$，$\sum n \le 2 \times 10^5$

flag 题解

- 注意到这里的走法类似一个dfs的过程，那么我们自然希望每棵子树在回溯时跳的步长尽量小。
- 设 f_u 表示当前在结点 u，u 子树中最后遍历到的结点深度最浅是多少。
- 转移考虑 u 的所有子节点 v_i，那么肯定会将其中 f_{v_i} 最小的 v_i 放在最后遍历。过程中统计跳的最远的距离即可。
- 注意根节点的时候情况发生了变化，你只需要选择一个距离不超过 k 的 v_i 放到最后遍历，注意特判。
- 时间复杂度 $O(\sum n)$。

bracketplus

- 给定序列 a，小 T 想知道有多少 a 的非空子区间使得它是符合规范的超级括号序列。
- 符合规范的超级括号序列是指存在一种将每个数看成对应种类的左括号或右括号的方法，使得它是一个合法的括号序列。
- $1 \le n \le 3 \times 10^5$

bracketplus 题解

- 考虑使用栈来判定某个序列是否合法：
  - 开始时栈为空。
  - 从左到右加入序列中的元素，每次加入一个元素时判断它与栈顶是否相同，若相同直接匹配并将栈顶弹出，否则将该元素入栈。
  - 最后栈为空则序列合法。
- 接着，我们需要观察到一个性质：开始的栈中即使有元素，只要开始的栈和结束的栈完全相同，那么它依旧可以用来判定合法。
- 那么对整个序列跑一遍这个过程，则 $[l, r]$ 是合法子区间当且仅当 $l-1$ 时刻的栈与 r 时刻的栈相同。使用常规的字符串哈希可以做到末尾快速插入删除元素。
- 时间复杂度 $O(n)$。

string

- 定义长为 n 的 01 串 b 能被长为 n 的 01 串 a 生成当且仅当存在一个 $1\sim n$ 的排列 p_i，满足对于每个 $1\le i\le n$，有 $-1\le p_i-i\le k$ 且 $b_{p_i}=a_i$。其中 k 为给定的正整数。
- 定义一个 01 串的美观度为它能生成不同的 01 串的数量。
- 给定一个 01 串 A，你需要求出它的所有非空子区间的美观度之和。答案对 998244353 取模。
- 由于串 A 太长了，我们会给定一个 01 串 S 以及正整数 t，串 A 恰好为 t 个 S 依次拼接而成。
- $1\le |S|\le 2\times 10^6$，$1\le t\le 10^9$，$1\le k\le |S|-1$

string 题解

- 首先，序列 $|p_i - i|$ 是由若干 $c, -1, -1, \ldots, -1$ 以及 0 组合而成，其中前者 -1 的个数恰好为 c。
- 注意到 a 和 b 最好的对应方式为对 a 中从左往右出现的第 i 个 0 对应到 b 中的第 i 个 0，字符 1 同理，若这种对应方式不行则不存在其它合法的对应方式了。
- 那么相当于是某些 0 或 1 会向后挪 c 格，且挪的过程中不能碰到与自己相同的字符。
- 此时不妨将连续的 0 或 1 放在一起考虑，每一段只有最后一个数可以向后挪。
- 先来考虑整个串的答案怎么求。设 $f_{i,0/1}$ 表示考虑前 i 段，是否有数挪到了最后，容易转移。
- 那么此时对于每一个左端点分别算答案即可做到 $O((t|S|)^2)$。

string 题解

- 接着考虑固定右端点，对于所有左端点，它们的 $f_{i,0/1}$ 在之后的转移实际上是完全一致的，所以其实它们都可以合并成为一个。
- 此时可以做到 $O(t|S|)$，但是 $t \le 10^9$，怎么加速呢？
- 这里的转移都是线性的，考虑使用矩阵，记录四维向量 $(ans, f_0, f_1, 1)$，其中 ans 表示目前的答案，$f_{0/1}$ 表示当前的 $f_{i,0/1}$，1 用来辅助转移。
- 对于每个段我们都可以写出 4×4 的矩阵，于是将串中的矩阵乘起来，再做一次矩阵快速幂即可。
- 注意特判全0/全1以及首尾字符相同的情况。
- 时间复杂度 $O(|S| + \log t)$。