CodeM 美团点评编程大赛资格赛题目讲评

quailty

Beijing Normal University

2017 年 6 月 14 日

音乐研究

- 给定一个长为 n 的序列 $a_1, a_2, \cdots, a_n$ 和一个长为 m 的序列 $b_1, b_2, \cdots, b_m$，你需要找一个整数 $i (0 \le i \le m - n)$ 使得 $\sum_{j=1}^{n} (a_j - b_{i+j})^2$ 最小，并求出这个最小值。

音乐研究

- 给定一个长为 n 的序列 $a_1, a_2, \cdots, a_n$ 和一个长为 m 的序列 $b_1, b_2, \cdots, b_m$，你需要找一个整数 $i (0 \leq i \leq m - n)$ 使得 $\sum_{j=1}^{n} (a_j - b_{i+j})^2$ 最小，并求出这个最小值。
- $1 \leq n \leq m \leq 1000$。

音乐研究

- 对于固定的 i，计算 $\sum_{j=1}^{n}(a_j - b_{i+j})^2$ 是 $O(n)$ 的。

音乐研究

- 对于固定的 i，计算 $\sum_{j=1}^{n}(a_j - b_{i+j})^2$ 是 $O(n)$ 的。
- 只需要枚举 i，求出 $m - n + 1$ 个 sum 的最小值即可。

音乐研究

- 对于固定的 i，计算 $\sum_{j=1}^{n}(a_j - b_{i+j})^2$ 是 $O(n)$ 的。
- 只需要枚举 i，求出 $m - n + 1$ 个 sum 的最小值即可。
- 复杂度 $O(mn)$。

音乐研究 EXT

- $1 \leq n \leq m \leq 100000$?

音乐研究 EXT

- $1 \le n \le m \le 100000$?
- $\sum_{j=1}^{n}(a_j - b_{i+j})^2 = \sum_{j=1}^{n} a_j^2 + \sum_{j=1}^{n} b_{i+j}^2 - 2\sum_{j=1}^{n} a_j b_{i+j}$。

音乐研究 EXT

- $1 \leq n \leq m \leq 100000$?
- $\sum_{j=1}^{n}(a_j - b_{i+j})^2 = \sum_{j=1}^{n} a_j^2 + \sum_{j=1}^{n} b_{i+j}^2 - 2\sum_{j=1}^{n} a_j b_{i+j}$。
- 前两个求和式是简单的，难点在于第三个求和式。

音乐研究 EXT

- $1 \leq n \leq m \leq 100000$?
- $\sum_{j=1}^{n}(a_j - b_{i+j})^2 = \sum_{j=1}^{n} a_j^2 + \sum_{j=1}^{n} b_{i+j}^2 - 2\sum_{j=1}^{n} a_j b_{i+j}$。
- 前两个求和式是简单的，难点在于第三个求和式。
- 构造多项式 $f = \sum_{j=1}^{n} a_j x^{-j}$，$g = \sum_{j=1}^{m} b_j x^j$。

音乐研究 EXT

- $1 \leq n \leq m \leq 100000$?
- $\sum_{j=1}^{n}(a_j - b_{i+j})^2 = \sum_{j=1}^{n} a_j^2 + \sum_{j=1}^{n} b_{i+j}^2 - 2\sum_{j=1}^{n} a_j b_{i+j}$。
- 前两个求和式是简单的，难点在于第三个求和式。
- 构造多项式 $f = \sum_{j=1}^{n} a_j x^{-j}$，$g = \sum_{j=1}^{m} b_j x^j$。
- $\sum_{j=1}^{n} a_j b_{i+j}$ 就是 $f \cdot g$ 的 x^i 项的系数。

音乐研究 EXT

- $1 \le n \le m \le 100000$?
- $\sum_{j=1}^{n}(a_j - b_{i+j})^2 = \sum_{j=1}^{n} a_j^2 + \sum_{j=1}^{n} b_{i+j}^2 - 2\sum_{j=1}^{n} a_j b_{i+j}$。
- 前两个求和式是简单的，难点在于第三个求和式。
- 构造多项式 $f = \sum_{j=1}^{n} a_j x^{-j}$，$g = \sum_{j=1}^{m} b_j x^j$。
- $\sum_{j=1}^{n} a_j b_{i+j}$ 就是 $f \cdot g$ 的 x^i 项的系数。
- Karatsuba/FFT 优化多项式乘法。

音乐研究 EXT

- $1 \le n \le m \le 100000$？
- $\sum_{j=1}^{n}(a_j - b_{i+j})^2 = \sum_{j=1}^{n} a_j^2 + \sum_{j=1}^{n} b_{i+j}^2 - 2\sum_{j=1}^{n} a_j b_{i+j}$。
- 前两个求和式是简单的，难点在于第三个求和式。
- 构造多项式 $f = \sum_{j=1}^{n} a_j x^{-j}$，$g = \sum_{j=1}^{m} b_j x^j$。
- $\sum_{j=1}^{n} a_j b_{i+j}$ 就是 $f \cdot g$ 的 x^i 项的系数。
- Karatsuba/FFT 优化多项式乘法。
- 复杂度 $O(m^{\log_2 3}) = O(m^{1.59})$ 或者 $O(m \log m)$。

锦标赛

- 给出 n 个选手的积分 $a_1, a_2, \cdots, a_n$，每一轮当前未被淘汰的选手两两捉对厮杀，积分高的一定胜出，积分相同随机胜负，胜者晋级下一轮，直到最后只剩一个选手，问 1 号选手最多能获胜多少场。

锦标赛

- 给出 n 个选手的积分 $a_1, a_2, \cdots, a_n$，每一轮当前未被淘汰的选手两两捉对厮杀，积分高的一定胜出，积分相同随机胜负，胜者晋级下一轮，直到最后只剩一个选手，问 1 号选手最多能获胜多少场。
- $n = 2^p, p = 0, 1, 2, \cdots, 20$。

锦标赛

- 假设 1 号选手的积分在所有选手中是第 k 小的。

锦标赛

- 假设 1 号选手的积分在所有选手中是第 k 小的。
- $ans = \lfloor \log_2 k \rfloor$。

锦标赛

- 假设 1 号选手的积分在所有选手中是第 k 小的。
- $ans = \lfloor \log_2 k \rfloor$。
- 证明？

优惠券

- 按照时间顺序给出 m 条记录，每条记录可能是购买了 x 号优惠券，可能是使用了 x 号优惠券，也可能是未知，你需要求出最大的 s 使得第 1 到第 $s-1$ 条记录是正确的，如果所有记录都正确那么输出 -1。

优惠券

- 按照时间顺序给出 m 条记录，每条记录可能是购买了 x 号优惠券，可能是使用了 x 号优惠券，也可能是未知，你需要求出最大的 s 使得第 1 到第 $s-1$ 条记录是正确的，如果所有记录都正确那么输出 -1。
- $0 \leq m \leq 500000$，$1 \leq x \leq 100000$。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。
- 如果已知记录出现错误，可以考虑利用未知记录来修正错误。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。
- 如果已知记录出现错误，可以考虑利用未知记录来修正错误。
- 如果在使用 x 号优惠券时发生错误，也就是此时没有 x 号优惠券。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。
- 如果已知记录出现错误，可以考虑利用未知记录来修正错误。
- 如果在使用 x 号优惠券时发生错误，也就是此时没有 x 号优惠券。
- 需要在上一次使用 x 号优惠券之后找一条最早的未知记录将其改为“购买 x 号优惠券”。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。
- 如果已知记录出现错误，可以考虑利用未知记录来修正错误。
- 如果在使用 x 号优惠券时发生错误，也就是此时没有 x 号优惠券。
- 需要在上一次使用 x 号优惠券之后找一条最早的未知记录将其改为“购买 x 号优惠券”。
- 类似处理在购买 x 号优惠券时发生错误的情况。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。
- 如果已知记录出现错误，可以考虑利用未知记录来修正错误。
- 如果在使用 x 号优惠券时发生错误，也就是此时没有 x 号优惠券。
- 需要在上一次使用 x 号优惠券之后找一条最早的未知记录将其改为“购买 x 号优惠券”。
- 类似处理在购买 x 号优惠券时发生错误的情况。
- std::set 维护未知记录的位置。

优惠券

- 如果已知记录都是正确的，那么加入未知记录不会导致错误。
- 如果已知记录出现错误，可以考虑利用未知记录来修正错误。
- 如果在使用 x 号优惠券时发生错误，也就是此时没有 x 号优惠券。
- 需要在上一次使用 x 号优惠券之后找一条最早的未知记录将其改为“购买 x 号优惠券”。
- 类似处理在购买 x 号优惠券时发生错误的情况。
- std::set 维护未知记录的位置。
- 复杂度 $O(m \log m)$。

送外卖

- 给出 n 个点，从 0 到 $n-1$ 标号，从 i 号点出发按照方案 a 可以走到 $i+a_i$ 号点，按照方案 b 可以走到 $i+b_i$ 号点，不允许走出这 n 个点，你需要判断是否存在从 0 号点出发到 $n-1$ 号点的方案，如果不存在输出 "No solution!"，否则输出字典序最小的方案，如果字典序最小的方案长度无限输出 "Infinity!"。

送外卖

- 给出 n 个点，从 0 到 $n-1$ 标号，从 i 号点出发按照方案 a 可以走到 $i+a_i$ 号点，按照方案 b 可以走到 $i+b_i$ 号点，不允许走出这 n 个点，你需要判断是否存在从 0 号点出发到 $n-1$ 号点的方案，如果不存在输出 “No solution!”，否则输出字典序最小的方案，如果字典序最小的方案长度无限输出 “Infinity!”。
- $1 \leq n \leq 100000$，$-n \leq a_i, b_i \leq n$。

送外卖

- “为什么长度会无限啊？”

送外卖

- “为什么长度会无限啊？”
- 从 $n-1$ 号点出发倒着 DFS 一次，求出所有能到达 $n-1$ 号点的点。

送外卖

- “为什么长度会无限啊？”
- 从 $n-1$ 号点出发倒着 DFS 一次，求出所有能到达 $n-1$ 号点的点。
- 如果从 0 号点出发不能到达 $n-1$ 那么 “No solution!”。

送外卖

- “为什么长度会无限啊？”
- 从 $n-1$ 号点出发倒着 DFS 一次，求出所有能到达 $n-1$ 号点的点。
- 如果从 0 号点出发不能到达 $n-1$ 那么 “No solution!”。
- 否则从 0 号点出发 DFS，每次沿着字典序小的能走到能到达 $n-1$ 号点的点的方案走。

送外卖

- “为什么长度会无限啊？”
- 从 $n-1$ 号点出发倒着 DFS 一次，求出所有能到达 $n-1$ 号点的点。
- 如果从 0 号点出发不能到达 $n-1$ 那么 “No solution!”。
- 否则从 0 号点出发 DFS，每次沿着字典序小的能走到能到达 $n-1$ 号点的点的方案走。
- 如果在到达 $n-1$ 号点之前走出了一个环，虽然存在到达 $n-1$ 号点的方案，但是沿着这个环一直绕下去才能保持字典序最小，长度会无限长，此时 “Infinity!”。

送外卖

- “为什么长度会无限啊？”
- 从 $n-1$ 号点出发倒着 DFS 一次，求出所有能到达 $n-1$ 号点的点。
- 如果从 0 号点出发不能到达 $n-1$ 那么 “No solution!”。
- 否则从 0 号点出发 DFS，每次沿着字典序小的能走到能到达 $n-1$ 号点的点的方案走。
- 如果在到达 $n-1$ 号点之前走出了一个环，虽然存在到达 $n-1$ 号点的方案，但是沿着这个环一直绕下去才能保持字典序最小，长度会无限长，此时 “Infinity!”。
- 否则输出路径。

送外卖

- “为什么长度会无限啊？”
- 从 $n-1$ 号点出发倒着 DFS 一次，求出所有能到达 $n-1$ 号点的点。
- 如果从 0 号点出发不能到达 $n-1$ 那么“No solution!”。
- 否则从 0 号点出发 DFS，每次沿着字典序小的能走到能到达 $n-1$ 号点的点的方案走。
- 如果在到达 $n-1$ 号点之前走出了一个环，虽然存在到达 $n-1$ 号点的方案，但是沿着这个环一直绕下去才能保持字典序最小，长度会无限长，此时“Infinity!”。
- 否则输出路径。
- 复杂度 $O(n)$。

数码

- 给定 l 和 r，对所有满足 $l \le x \le r$ 的 x，把 x 的所有约数全部写下来。对于每个写下来的数，只保留最高位的那个数码。求 $1, 2, \cdots, 9$ 每个数码出现的次数。

数码

- 给定 l 和 r，对所有满足 $l \leq x \leq r$ 的 x，把 x 的所有约数全部写下来。对于每个写下来的数，只保留最高位的那个数码。求 $1, 2, \cdots, 9$ 每个数码出现的次数。
- $1 \leq l \leq r \leq 1000000000$。

数码

- $[l, r] = [1, r] - [1, l-1]$，只需要考虑 $[1, n]$ 内的子问题。

数码

- $[l, r] = [1, r] - [1, l-1]$，只需要考虑 $[1, n]$ 内的子问题。
- 对于一个 d，它是 $d, 2d, 3d, \ldots$ 的约数，也就是 $[1, n]$ 内 $\lfloor \frac{n}{d} \rfloor$ 个数的约数。

数码

- $[l, r] = [1, r] - [1, l-1]$，只需要考虑 $[1, n]$ 内的子问题。
- 对于一个 d，它是 $d, 2d, 3d, \ldots$ 的约数，也就是 $[1, n]$ 内 $\lfloor \frac{n}{d} \rfloor$ 个数的约数。
- 考虑枚举 d 的最高位的数码 p 以及长度 q，那么有 $d \in [p \cdot 10^{q-1}, (p+1) \cdot 10^{q-1})$，由于 d 最大是 1000000000，q 枚举到 10 就够了。

数码

- $[l,r]=[1,r]-[1,l-1]$，只需要考虑 $[1,n]$ 内的子问题。
- 对于一个 d，它是 $d,2d,3d,\dots$ 的约数，也就是 $[1,n]$ 内 $\lfloor\frac{n}{d}\rfloor$ 个数的约数。
- 考虑枚举 d 的最高位的数码 p 以及长度 q，那么有 $d\in[p\cdot 10^{q-1},(p+1)\cdot 10^{q-1})$，由于 d 最大是 1000000000，q 枚举到 10 就够了。
- 那么 $res_p=\sum_{q=1}^{10}\sum_{d=p\cdot 10^{q-1}}^{(p+1)\cdot 10^{q-1}-1}\lfloor\frac{n}{d}\rfloor$。

数码

- $[l, r] = [1, r] - [1, l-1]$，只需要考虑 $[1, n]$ 内的子问题。
- 对于一个 d，它是 $d, 2d, 3d, \ldots$ 的约数，也就是 $[1, n]$ 内 $\lfloor \frac{n}{d} \rfloor$ 个数的约数。
- 考虑枚举 d 的最高位的数码 p 以及长度 q，那么有 $d \in [p \cdot 10^{q-1}, (p+1) \cdot 10^{q-1})$，由于 d 最大是 1000000000，q 枚举到 10 就够了。
- 那么 $res_p = \sum_{q=1}^{10} \sum_{d=p \cdot 10^{q-1}}^{(p+1) \cdot 10^{q-1} - 1} \lfloor \frac{n}{d} \rfloor$。
- 记 $S(n, m) = \sum_{d=1}^{m} \lfloor \frac{n}{d} \rfloor$。

数码

- $[l, r] = [1, r] - [1, l-1]$，只需要考虑 $[1, n]$ 内的子问题。
- 对于一个 d，它是 $d, 2d, 3d, \dots$ 的约数，也就是 $[1, n]$ 内 $\lfloor \frac{n}{d} \rfloor$ 个数的约数。
- 考虑枚举 d 的最高位的数码 p 以及长度 q，那么有 $d \in [p \cdot 10^{q-1}, (p+1) \cdot 10^{q-1})$，由于 d 最大是 1000000000，q 枚举到 10 就够了。
- 那么 $res_p = \sum_{q=1}^{10} \sum_{d=p\cdot 10^{q-1}}^{(p+1)\cdot 10^{q-1}-1} \lfloor \frac{n}{d} \rfloor$。
- 记 $S(n, m) = \sum_{d=1}^{m} \lfloor \frac{n}{d} \rfloor$。
- $res_p = \sum_{q=1}^{10} (S(n, (p+1) \cdot 10^{q-1} - 1) - S(n, p \cdot 10^{q-1} - 1))$。

数码

- $[l,r]=[1,r]-[1,l-1]$，只需要考虑 $[1,n]$ 内的子问题。
- 对于一个 d，它是 $d,2d,3d,\ldots$ 的约数，也就是 $[1,n]$ 内 $\lfloor \frac{n}{d} \rfloor$ 个数的约数。
- 考虑枚举 d 的最高位的数码 p 以及长度 q，那么有 $d \in [p \cdot 10^{q-1},(p+1)\cdot 10^{q-1})$，由于 d 最大是 1000000000，q 枚举到 10 就够了。
- 那么 $res_p=\sum_{q=1}^{10}\sum_{d=p\cdot 10^{q-1}}^{(p+1)\cdot 10^{q-1}-1}\lfloor \frac{n}{d} \rfloor$。
- 记 $S(n,m)=\sum_{d=1}^{m}\lfloor \frac{n}{d} \rfloor$。
- $res_p=\sum_{q=1}^{10}(S(n,(p+1)\cdot 10^{q-1}-1)-S(n,p\cdot 10^{q-1}-1))$。
- "为什么 $O(n)$ 也超时了啊？"

数码

- 问题在于如何快速计算 $S(n, m)$。

数码

- 问题在于如何快速计算 $S(n, m)$。
- 如果 $m \leq \sqrt{n}$，直接 $O(m)$ 计算。

数码

- 问题在于如何快速计算 $S(n, m)$。
- 如果 $m \leq \sqrt{n}$，直接 $O(m)$ 计算。
- 否则 $S(n, m) = \sum_{d=1}^{\lfloor\sqrt{n}\rfloor} \lfloor \frac{n}{d} \rfloor + \sum_{d=\lfloor\sqrt{n}\rfloor+1}^{m} \lfloor \frac{n}{d} \rfloor$

数码

- 问题在于如何快速计算 $S(n,m)$。
- 如果 $m \le \sqrt{n}$，直接 $O(m)$ 计算。
- 否则 $S(n,m) = \sum_{d=1}^{\lfloor\sqrt{n}\rfloor} \lfloor \frac{n}{d} \rfloor + \sum_{d=\lfloor\sqrt{n}\rfloor+1}^{m} \lfloor \frac{n}{d} \rfloor$
- 前一个求和式直接 $O(\sqrt{n})$ 计算。

数码

- 问题在于如何快速计算 $S(n,m)$。
- 如果 $m \le \sqrt{n}$，直接 $O(m)$ 计算。
- 否则 $S(n,m) = \sum_{d=1}^{\lfloor\sqrt{n}\rfloor} \lfloor \frac{n}{d} \rfloor + \sum_{d=\lfloor\sqrt{n}\rfloor+1}^{m} \lfloor \frac{n}{d} \rfloor$
- 前一个求和式直接 $O(\sqrt{n})$ 计算。
- 对于后一个求和式，注意到 $\lfloor \frac{n}{d} \rfloor < \lfloor \sqrt{n} \rfloor$，考虑枚举 $\lfloor \frac{n}{d} \rfloor$ 的值，记为 $k(>0)$。

数码

- 问题在于如何快速计算 $S(n, m)$。
- 如果 $m \le \sqrt{n}$，直接 $O(m)$ 计算。
- 否则 $S(n, m) = \sum_{d=1}^{\lfloor \sqrt{n} \rfloor} \lfloor \frac{n}{d} \rfloor + \sum_{d=\lfloor \sqrt{n} \rfloor + 1}^{m} \lfloor \frac{n}{d} \rfloor$
- 前一个求和式直接 $O(\sqrt{n})$ 计算。
- 对于后一个求和式，注意到 $\lfloor \frac{n}{d} \rfloor < \lfloor \sqrt{n} \rfloor$，考虑枚举 $\lfloor \frac{n}{d} \rfloor$ 的值，记为 $k(> 0)$。
- 可以求出 $d \in (\lfloor \frac{n}{k+1} \rfloor, \lfloor \frac{n}{k} \rfloor]$，将这个区间和 $[1, m]$ 求交之后可以知道有多少个 $d \in [1, m]$ 满足 $\lfloor \frac{n}{d} \rfloor = k$，记为 cnt_k。

数码

- 问题在于如何快速计算 $S(n,m)$。
- 如果 $m \le \sqrt{n}$，直接 $O(m)$ 计算。
- 否则 $S(n,m) = \sum_{d=1}^{\lfloor\sqrt{n}\rfloor} \lfloor \frac{n}{d} \rfloor + \sum_{d=\lfloor\sqrt{n}\rfloor+1}^{m} \lfloor \frac{n}{d} \rfloor$
- 前一个求和式直接 $O(\sqrt{n})$ 计算。
- 对于后一个求和式，注意到 $\lfloor \frac{n}{d} \rfloor < \lfloor \sqrt{n} \rfloor$，考虑枚举 $\lfloor \frac{n}{d} \rfloor$ 的值，记为 $k(>0)$。
- 可以求出 $d \in (\lfloor \frac{n}{k+1} \rfloor, \lfloor \frac{n}{k} \rfloor]$，将这个区间和 $[1,m]$ 求交之后可以知道有多少个 $d \in [1,m]$ 满足 $\lfloor \frac{n}{d} \rfloor = k$，记为 cnt_k。
- 那么 $\sum_{d=\lfloor\sqrt{n}\rfloor+1}^{m} \lfloor \frac{n}{d} \rfloor = \sum_{k=1}^{\lfloor\sqrt{n}\rfloor-1} k \cdot cnt_k$，由于 cnt_k 可以 $O(1)$ 计算，这部分复杂度也是 $O(\sqrt{n})$。

围棋

- 给出围棋的规则，输入一些操作，从空棋盘开始模拟这些操作。对于每一步，若结果不正确，则输出对应的 miss 并且忽略这个操作，并在最后输出棋盘的局面。

围棋

- 给出围棋的规则，输入一些操作，从空棋盘开始模拟这些操作。对于每一步，若结果不正确，则输出对应的 miss 并且忽略这个操作，并在最后输出棋盘的局面。
- 测试数据组数 ≤ 100，每组测试数据操作数 $n \leq 2000$。

围棋

- "根据题意模拟就好了。"

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是“miss 1”。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是“miss 1”。
- 否则尝试把这个棋子下下去。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是“miss 1”。
- 否则尝试把这个棋子下下去。
- 不妨设这一步是黑棋下，从这个黑棋出发 DFS 找出所在连通块并判断是否有气。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是“miss 1”。
- 否则尝试把这个棋子下下去。
- 不妨设这一步是黑棋下，从这个黑棋出发 DFS 找出所在连通块并判断是否有气。
- 然后从与这个黑棋相邻的白棋出发 DFS 找出所在连通块并判断是否有气，如果没有气那么整个连通块被提掉。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是 “miss 1”。
- 否则尝试把这个棋子下下去。
- 不妨设这一步是黑棋下，从这个黑棋出发 DFS 找出所在连通块并判断是否有气。
- 然后从与这个黑棋相邻的白棋出发 DFS 找出所在连通块并判断是否有气，如果没有气那么整个连通块被提掉。
- 如果黑棋下下去之后没有气，又不能提白子，就是 “miss 2”。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是“miss 1”。
- 否则尝试把这个棋子下下去。
- 不妨设这一步是黑棋下，从这个黑棋出发 DFS 找出所在连通块并判断是否有气。
- 然后从与这个黑棋相邻的白棋出发 DFS 找出所在连通块并判断是否有气，如果没有气那么整个连通块被提掉。
- 如果黑棋下下去之后没有气，又不能提白子，就是“miss 2”。
- 否则利用 hash 判断当前局面是否出现过，如果出现过就是“miss 3”。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是 “miss 1”。
- 否则尝试把这个棋子下下去。
- 不妨设这一步是黑棋下，从这个黑棋出发 DFS 找出所在连通块并判断是否有气。
- 然后从与这个黑棋相邻的白棋出发 DFS 找出所在连通块并判断是否有气，如果没有气那么整个连通块被提掉。
- 如果黑棋下下去之后没有气，又不能提白子，就是 “miss 2”。
- 否则利用 hash 判断当前局面是否出现过，如果出现过就是 “miss 3”。
- 如果出错，把棋盘还原到黑棋下下去之前的状态。

围棋

- “根据题意模拟就好了。”
- 如果当前下的位置已经有棋了，就是“miss 1”。
- 否则尝试把这个棋子下下去。
- 不妨设这一步是黑棋下，从这个黑棋出发 DFS 找出所在连通块并判断是否有气。
- 然后从与这个黑棋相邻的白棋出发 DFS 找出所在连通块并判断是否有气，如果没有气那么整个连通块被提掉。
- 如果黑棋下下去之后没有气，又不能提白子，就是“miss 2”。
- 否则利用 hash 判断当前局面是否出现过，如果出现过就是“miss 3”。
- 如果出错，把棋盘还原到黑棋下下去之前的状态。
- 复杂度 $O(tns)$，t 表示测试数据的组数，s 表示棋盘大小。

Thank you!