扫描线的应用

Claris

Hangzhou Dianzi University

2017 年 7 月 25 日

Overview

- 扫描线是一种“关键点”的思想，只关心在关键点处的变化，跳过不变化的部分，以达到优化效果。

Overview

- 扫描线是一种“关键点”的思想，只关心在关键点处的变化，跳过不变化的部分，以达到优化效果。
- 计算几何中的扫描线。

Overview

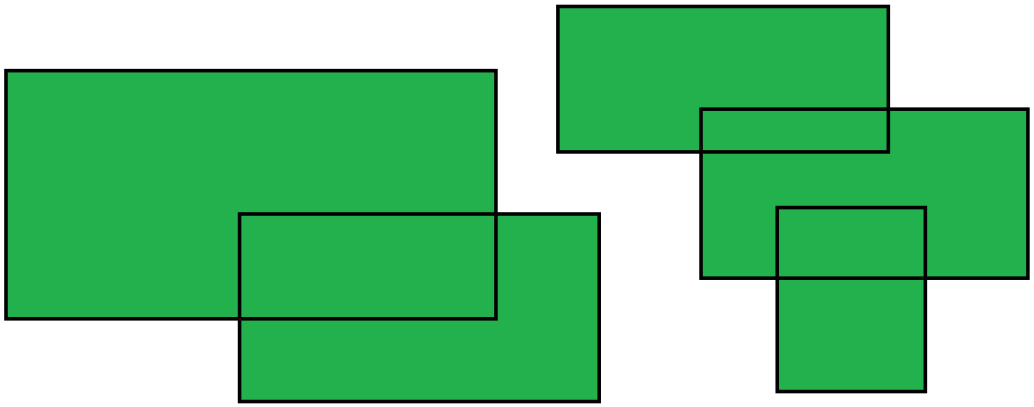
- 扫描线是一种“关键点”的思想，只关心在关键点处的变化，跳过不变化的部分，以达到优化效果。
- 计算几何中的扫描线。
- 扫描线解决维护统计问题。

矩形面积并

给定平面上 n 个平行于坐标轴的矩形，求被这些矩形覆盖部分的面积。

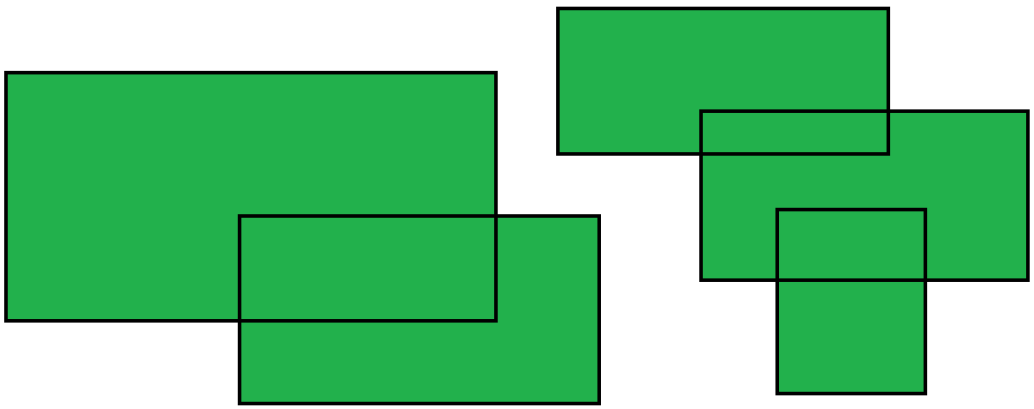
矩形面积并

给定平面上 n 个平行于坐标轴的矩形，求被这些矩形覆盖部分的面积。



矩形面积并

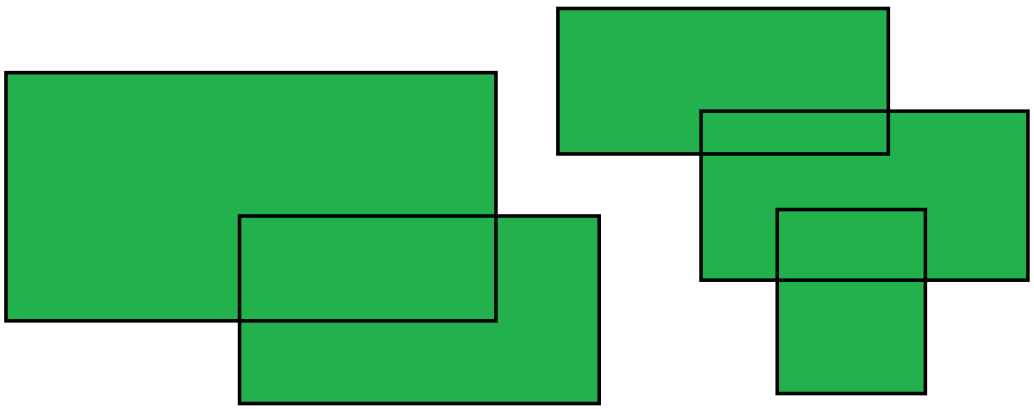
给定平面上 n 个平行于坐标轴的矩形，求被这些矩形覆盖部分的面积。



- $1 \leq n \leq 100000$。

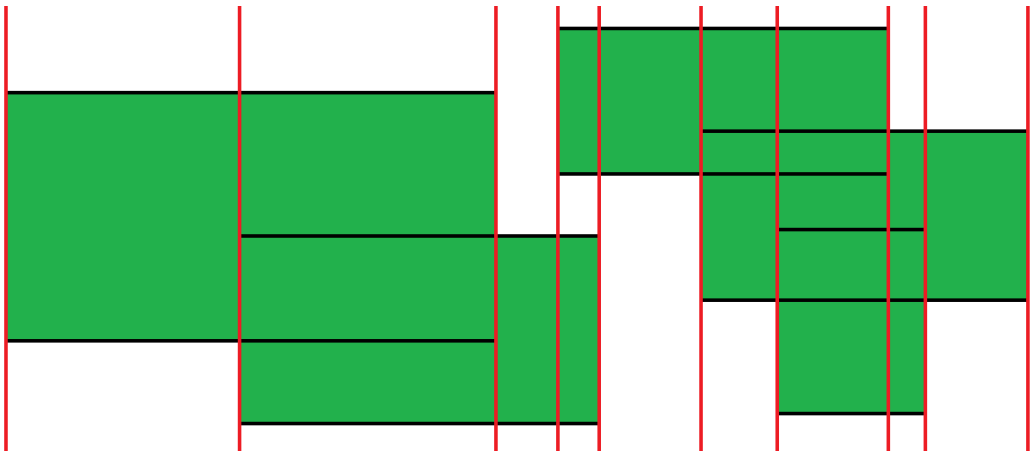
矩形面积并

给定平面上 n 个平行于坐标轴的矩形，求被这些矩形覆盖部分的面积。

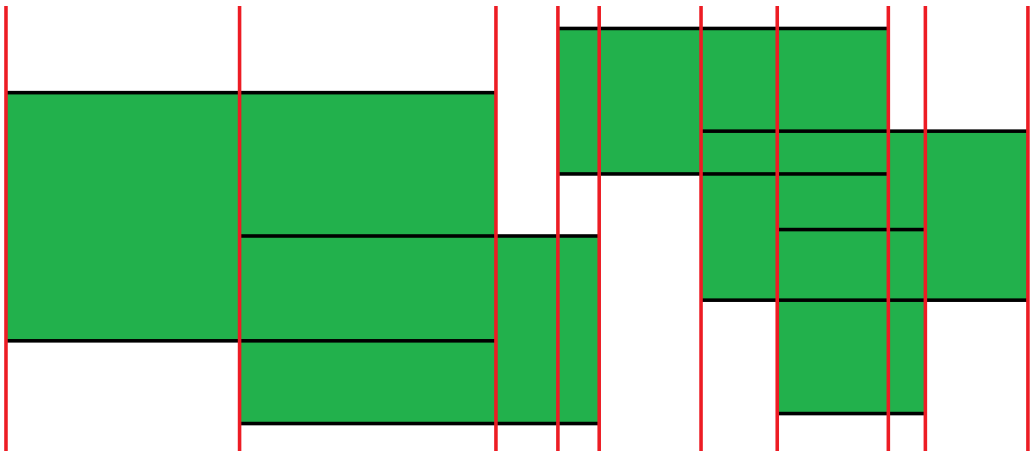


- $1 \leq n \leq 100000$。
- $1 \leq x_i, y_i \leq 10^9$。

Solution

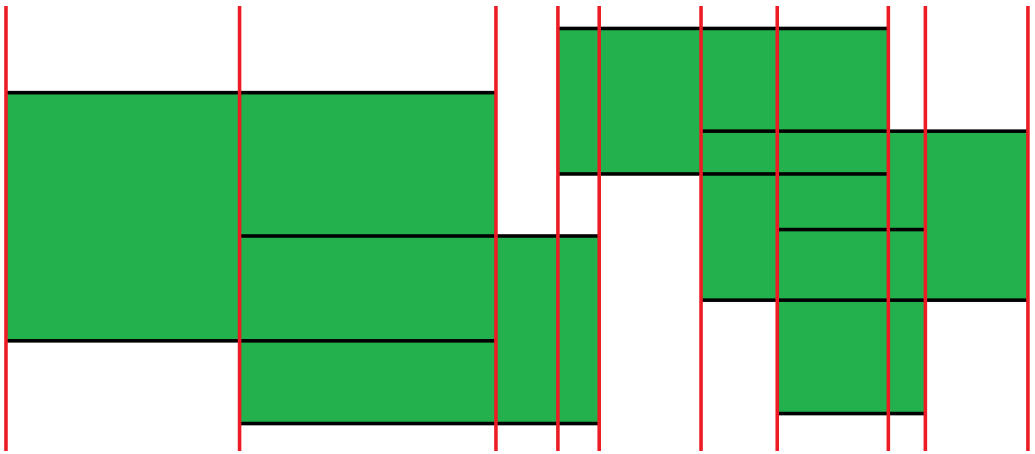


Solution



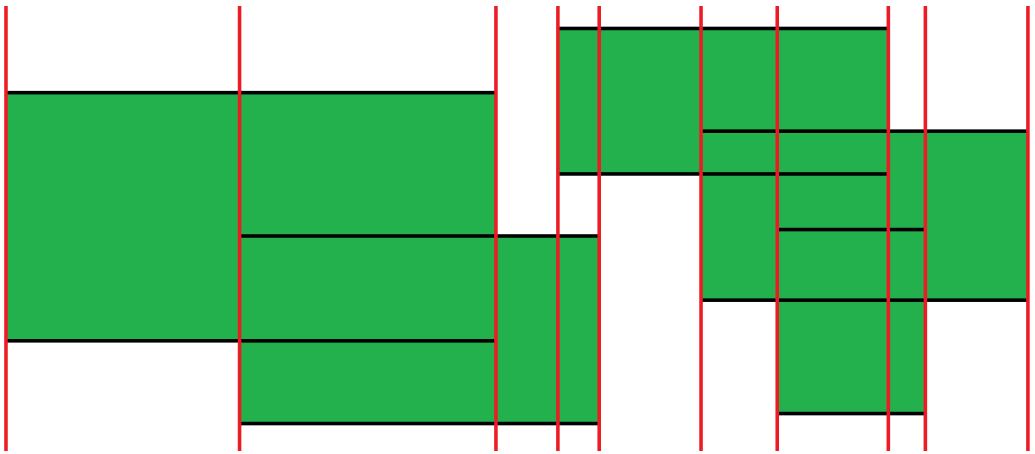
- 考虑将一根竖直线从左往右扫描，那么答案就是每个时刻线上被覆盖的长度之和。

Solution



- 考虑将一根竖直线从左往右扫描，那么答案就是每个时刻线上被覆盖的长度之和。
- 相邻关键点之间扫描线上被覆盖的长度不会变化，直接乘以横坐标差值即可。

Solution



- 考虑将一根竖直线从左往右扫描，那么答案就是每个时刻线上被覆盖的长度之和。
- 相邻关键点之间扫描线上被覆盖的长度不会变化，直接乘以横坐标差值即可。
- 关键点：(1) 在 $x = xl_i$ 处加入矩形 i。(2) 在 $x = xr_i$ 处删除矩形 i。

Solution

- 纵方向上则是一维覆盖问题，线段树维护。

Solution

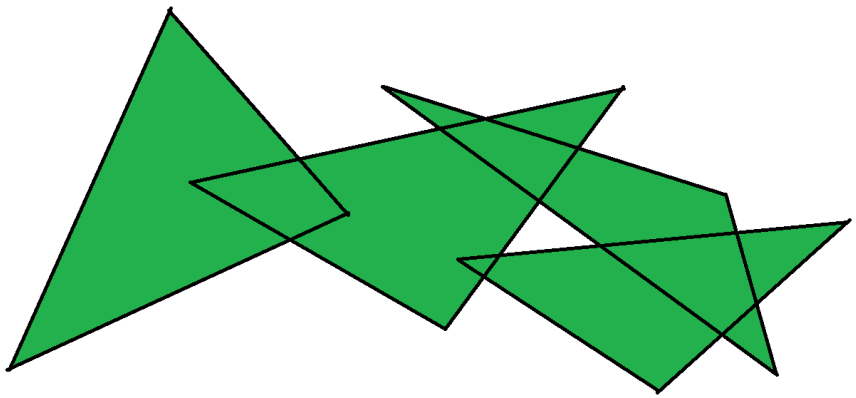
- 纵方向上则是一维覆盖问题，线段树维护。
- 时间复杂度 $O(n \log n)$。

三角形面积并

给定平面上 n 个三角形，求被这些三角形覆盖部分的面积。

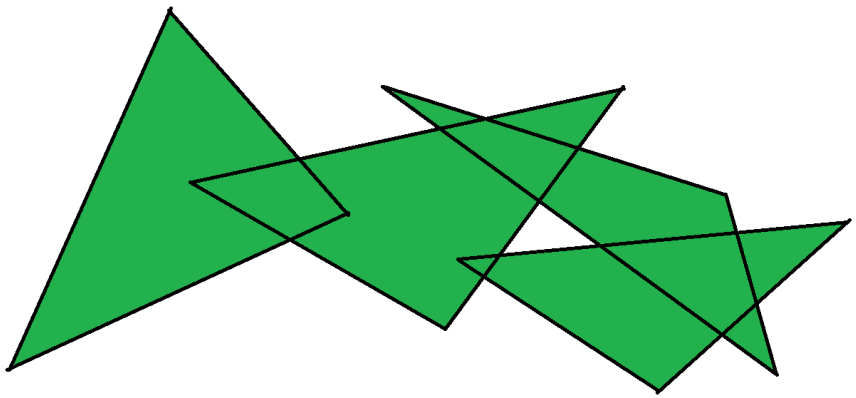
三角形面积并

给定平面上 n 个三角形，求被这些三角形覆盖部分的面积。



三角形面积并

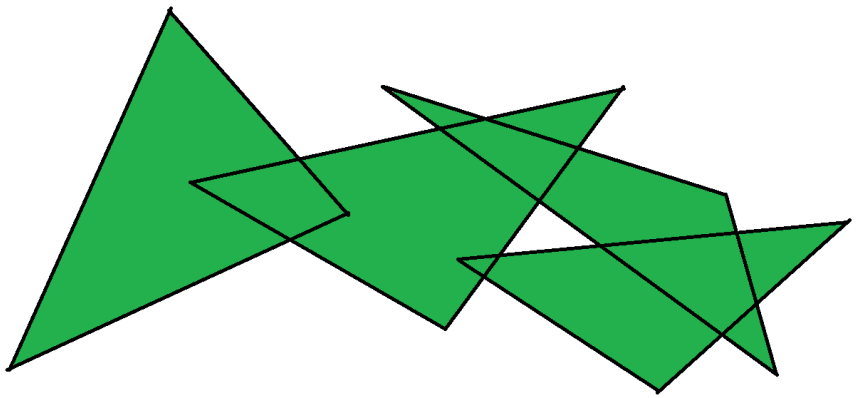
给定平面上 n 个三角形，求被这些三角形覆盖部分的面积。



- $1 \leq n \leq 100$。

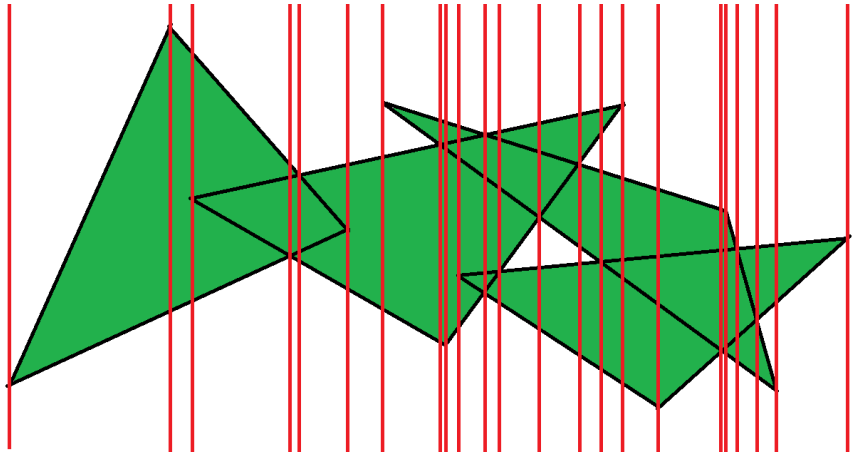
三角形面积并

给定平面上 n 个三角形，求被这些三角形覆盖部分的面积。

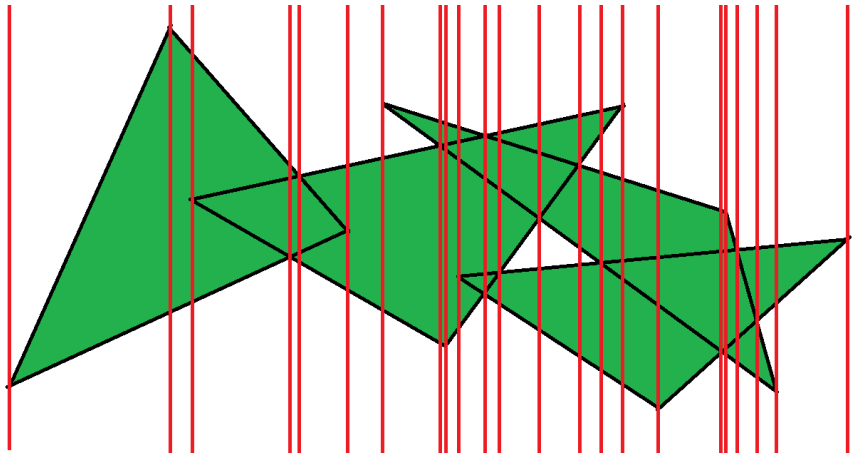


- $1 \leq n \leq 100$。
- $1 \leq x_i, y_i \leq 10^9$。

Solution

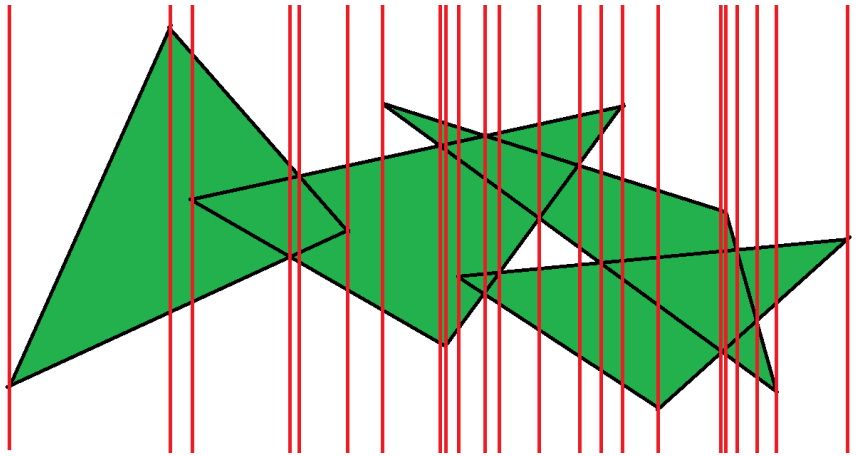


Solution



- 关键点：三角形的顶点或交点。

Solution



- 关键点：三角形的顶点或交点。
- 相邻关键点之间扫描线上被覆盖的长度是线性变化的，用中位线去截该图形，求出被覆盖的长度，然后乘以横坐标差值即可。

Solution

- 对于中位线上被覆盖的长度，则是线段覆盖问题，区间求并即可。

Solution

- 对于中位线上被覆盖的长度，则是线段覆盖问题，区间求并即可。
- 时间复杂度 $O(n^3 \log n)$。

Beacons

平面上有 n 个烽火台 (视作点)，m 个山脉 (视作圆)，任意两座山脉不会相碰，烽火台不会位于山脉中。

Beacons

平面上有 n 个烽火台 (视作点)，m 个山脉 (视作圆)，任意两座山脉不会相碰，烽火台不会位于山脉中。

两个烽火台能彼此望见，当且仅当它们的连线不经过任何山脉。数据保证不会有相切的情况。

Beacons

平面上有 n 个烽火台 (视作点)，m 个山脉 (视作圆)，任意两座山脉不会相碰，烽火台不会位于山脉中。

两个烽火台能彼此望见，当且仅当它们的连线不经过任何山脉。数据保证不会有相切的情况。

请判断任意两个烽火台能否彼此望见。

Beacons

平面上有 n 个烽火台 (视作点)，m 个山脉 (视作圆)，任意两座山脉不会相碰，烽火台不会位于山脉中。

两个烽火台能彼此望见，当且仅当它们的连线不经过任何山脉。数据保证不会有相切的情况。

请判断任意两个烽火台能否彼此望见。

- $1 \leq n \leq 1000, 0 \leq m \leq 1000$。

Beacons

平面上有 n 个烽火台 (视作点)，m 个山脉 (视作圆)，任意两座山脉不会相碰，烽火台不会位于山脉中。

两个烽火台能彼此望见，当且仅当它们的连线不经过任何山脉。数据保证不会有相切的情况。

请判断任意两个烽火台能否彼此望见。

- $1 \leq n \leq 1000, 0 \leq m \leq 1000$。
- $0 \leq x_i, y_i \leq 10000, 1 \leq r_i \leq 5000$。

Beacons

平面上有 n 个烽火台 (视作点)，m 个山脉 (视作圆)，任意两座山脉不会相碰，烽火台不会位于山脉中。

两个烽火台能彼此望见，当且仅当它们的连线不经过任何山脉。数据保证不会有相切的情况。

请判断任意两个烽火台能否彼此望见。

- $1 \leq n \leq 1000, 0 \leq m \leq 1000$。
- $0 \leq x_i, y_i \leq 10000, 1 \leq r_i \leq 5000$。
- Source： NCPC 2009

Solution

- 枚举每个烽火台作为原点，判断其与剩余每个烽火台是否可以望见。

Solution

- 枚举每个烽火台作为原点，判断其与剩余每个烽火台是否可以望见。
- 需要知道原点到每个烽火台连线上最近的山脉。

Solution

- 枚举每个烽火台作为原点，判断其与剩余每个烽火台是否可以望见。
- 需要知道原点到每个烽火台连线上最近的山脉。
- 按极角扫描，关键点：(1) 加入山脉。(2) 删除山脉。(3) 询问烽火台。

Solution

- 枚举每个烽火台作为原点，判断其与剩余每个烽火台是否可以望见。
- 需要知道原点到每个烽火台连线上最近的山脉。
- 按极角扫描，关键点：(1) 加入山脉。(2) 删除山脉。(3) 询问烽火台。
- 每个山脉可以隔离掉与原点距离超过其切线长度的所有烽火台。

Solution

- 枚举每个烽火台作为原点，判断其与剩余每个烽火台是否可以望见。
- 需要知道原点到每个烽火台连线上最近的山脉。
- 按极角扫描，关键点：(1) 加入山脉。(2) 删除山脉。(3) 询问烽火台。
- 每个山脉可以隔离掉与原点距离超过其切线长度的所有烽火台。
- 用堆维护扫描线中还存在的山脉对应的最小切线长度。

Solution

- 枚举每个烽火台作为原点，判断其与剩余每个烽火台是否可以望见。
- 需要知道原点到每个烽火台连线上最近的山脉。
- 按极角扫描，关键点：(1) 加入山脉。(2) 删除山脉。(3) 询问烽火台。
- 每个山脉可以隔离掉与原点距离超过其切线长度的所有烽火台。
- 用堆维护扫描线中还存在的山脉对应的最小切线长度。
- 时间复杂度 $O(n^2 \log n)$。

线段排序

平面上有 n 条不相交的线段，每次可以选择一条线段，将其往上移动直至移除，中途不得触碰其它线段。

线段排序

平面上有 n 条不相交的线段，每次可以选择一条线段，将其往上移动直至移除，中途不得触碰其它线段。

目标：将所有线段移除。试求一个可行的移除方案。

线段排序

平面上有 n 条不相交的线段，每次可以选择一条线段，将其往上移动直至移除，中途不得触碰其它线段。

目标：将所有线段移除。试求一个可行的移除方案。

- $1 \leq n \leq 100000$。

线段排序

平面上有 n 条不相交的线段，每次可以选择一条线段，将其往上移动直至移除，中途不得触碰其它线段。

目标：将所有线段移除。试求一个可行的移除方案。

- $1 \leq n \leq 100000$。
- $0 \leq x_i, y_i \leq 10^9$。

Solution

- 考虑枚举两条线段 i 与 j，判断 i 是否阻挡住了 j，若是则连边 $i \to j$，表示 j 要在 i 后移除。

Solution

- 考虑枚举两条线段 i 与 j，判断 i 是否阻挡住了 j，若是则连边 $i \to j$，表示 j 要在 i 后移除。
- 对这个图拓扑排序即可得到一组合法方案。

Solution

- 考虑枚举两条线段 i 与 j，判断 i 是否阻挡住了 j，若是则连边 $i \to j$，表示 j 要在 i 后移除。
- 对这个图拓扑排序即可得到一组合法方案。
- 时间复杂度 $O(n^2)$，不能接受。

Solution

- 对于一条线段 i，只需考虑端点是否被阻挡，那么需要知道其往上最近的线段。

Solution

- 对于一条线段 i，只需考虑端点是否被阻挡，那么需要知道其往上最近的线段。
- 从左往右扫描线，用 set 维护经过扫描线的线段，比较函数需要根据扫描线的位置现算，但是 set 中线段的相对顺序是不会变化的。

Solution

- 对于一条线段 i，只需考虑端点是否被阻挡，那么需要知道其往上最近的线段。
- 从左往右扫描线，用 set 维护经过扫描线的线段，比较函数需要根据扫描线的位置现算，但是 set 中线段的相对顺序是不会变化的。
- 每条线段只可能影响其前驱后继。

Solution

- 对于一条线段 i，只需考虑端点是否被阻挡，那么需要知道其往上最近的线段。
- 从左往右扫描线，用 set 维护经过扫描线的线段，比较函数需要根据扫描线的位置现算，但是 set 中线段的相对顺序是不会变化的。
- 每条线段只可能影响其前驱后继。
- 时间复杂度 $O(n \log n)$。

圆包含

平面上有 n 个不相交 (但可能互相包含) 的圆，试求出其包含关系，以树的形式表示。

圆包含

平面上有 n 个不相交 (但可能互相包含) 的圆，试求出其包含关系，以树的形式表示。

- $1 \leq n \leq 100000$。

圆包含

平面上有 n 个不相交 (但可能互相包含) 的圆，试求出其包含关系，以树的形式表示。

- $1 \leq n \leq 100000$。
- $1 \leq x_i, y_i, r_i \leq 10^9$。

Solution

- 将每个圆拆成上半圆和下半圆。

Solution

- 将每个圆拆成上半圆和下半圆。
- 从左往右扫描线，用 set 按从上到下的顺序维护经过扫描线的半圆。

Solution

- 将每个圆拆成上半圆和下半圆。
- 从左往右扫描线，用 set 按从上到下的顺序维护经过扫描线的半圆。
- 对于一个新加入的圆 i，找到其上方第一个半圆 j。

Solution

- 将每个圆拆成上半圆和下半圆。
- 从左往右扫描线，用 set 按从上到下的顺序维护经过扫描线的半圆。
- 对于一个新加入的圆 i，找到其上方第一个半圆 j。
- 若是一个上半圆，那么 i 的父亲即为 j。

Solution

- 将每个圆拆成上半圆和下半圆。
- 从左往右扫描线，用 set 按从上到下的顺序维护经过扫描线的半圆。
- 对于一个新加入的圆 i，找到其上方第一个半圆 j。
- 若是一个上半圆，那么 i 的父亲即为 j。
- 若是一个下半圆，那么 i 的父亲为 $father_j$。

Solution

- 将每个圆拆成上半圆和下半圆。
- 从左往右扫描线，用 set 按从上到下的顺序维护经过扫描线的半圆。
- 对于一个新加入的圆 i，找到其上方第一个半圆 j。
- 若是一个上半圆，那么 i 的父亲即为 j。
- 若是一个下半圆，那么 i 的父亲为 $father_j$。
- 时间复杂度 $O(n \log n)$。

二维数点

平面上有 n 个点，m 次询问一个平行坐标轴的矩形内部的点数。

二维数点

平面上有 n 个点，m 次询问一个平行坐标轴的矩形内部的点数。

- $1 \leq n, m \leq 100000$。

二维数点

平面上有 n 个点，m 次询问一个平行坐标轴的矩形内部的点数。

- $1 \leq n, m \leq 100000$。
- $1 \leq x_i, y_i \leq 10^9$。

Solution

- 拆分询问，利用容斥去掉下限。

Solution

- 拆分询问，利用容斥去掉下限。
- 从左往右扫描线，用树状数组维护扫描线扫过部分纵区间内点数。

Solution

- 拆分询问，利用容斥去掉下限。
- 从左往右扫描线，用树状数组维护扫描线扫过部分纵区间内点数。
- 时间复杂度 $O(n \log n)$。

Cow Confinement

一个 $10^6 \times 10^6$ 的网格图，上面有 n 头牛、m 朵花和 f 个矩形围栏。围栏在格子的边界上，牛和花在格子里。栅栏不会相互触碰。

Cow Confinement

一个 $10^6 \times 10^6$ 的网格图，上面有 n 头牛、m 朵花和 f 个矩形围栏。围栏在格子的边界上，牛和花在格子里。栅栏不会相互触碰。

牛只能向下或向右走，不能穿过围栏和地图边界，求每头牛能到达的花的数量。

Cow Confinement

一个 $10^6 \times 10^6$ 的网格图，上面有 n 头牛、m 朵花和 f 个矩形围栏。围栏在格子的边界上，牛和花在格子里。栅栏不会相互触碰。

牛只能向下或向右走，不能穿过围栏和地图边界，求每头牛能到达的花的数量。

- $0 \leq n, m, f \leq 200000$。

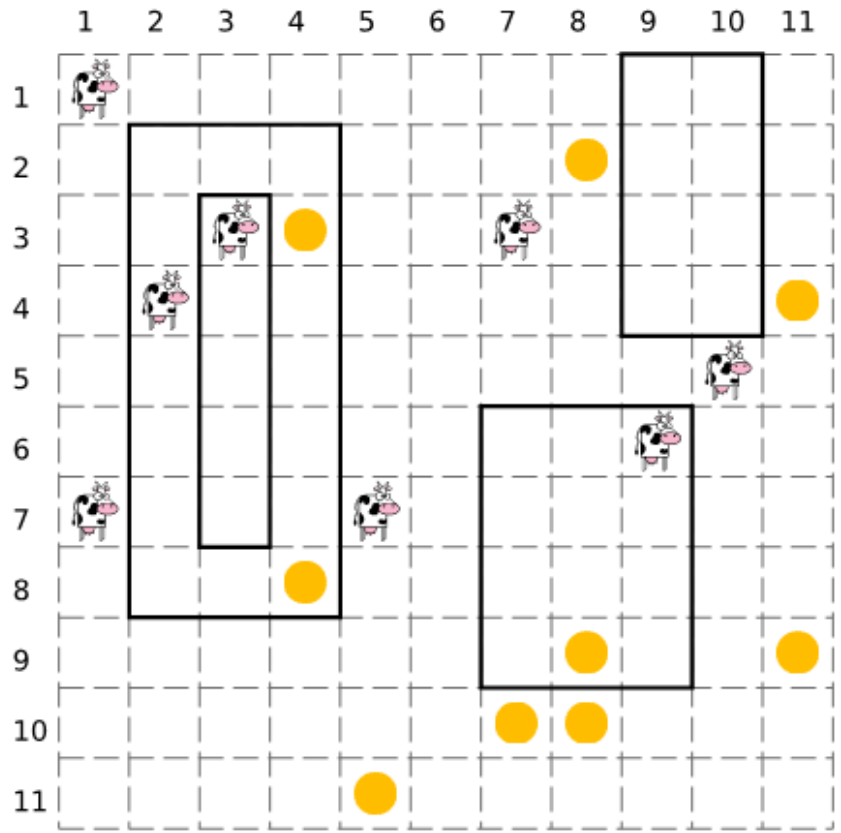
Cow Confinement

一个 $10^6 \times 10^6$ 的网格图，上面有 n 头牛、m 朵花和 f 个矩形围栏。围栏在格子的边界上，牛和花在格子里。栅栏不会相互触碰。

牛只能向下或向右走，不能穿过围栏和地图边界，求每头牛能到达的花的数量。

- $0 \leq n, m, f \leq 200000$。
- Source： CERC 2015

Cow Confinement



Solution

- 从右往左扫描线，用线段树维护扫描线上每一个点能达到的花的数量，并支持最近篱笆的查询。

Solution

- 从右往左扫描线，用线段树维护扫描线上每一个点能达到的花的数量，并支持最近篱笆的查询。
- 对于一朵花，找到它上方最近的篱笆，那么它对这中间的每头牛的贡献都是 1。

Solution

- 从右往左扫描线，用线段树维护扫描线上每一个点能达到的花的数量，并支持最近篱笆的查询。
- 对于一朵花，找到它上方最近的篱笆，那么它对这中间的每头牛的贡献都是 1。
- 当扫到一个篱笆的右边界时，这中间的答案都要清零。

Solution

- 从右往左扫描线，用线段树维护扫描线上每一个点能达到的花的数量，并支持最近篱笆的查询。
- 对于一朵花，找到它上方最近的篱笆，那么它对这中间的每头牛的贡献都是 1。
- 当扫到一个篱笆的右边界时，这中间的答案都要清零。
- 当扫到一个篱笆的左边界时，这中间的答案同理都要清零，但是要向上直到最近的篱笆为止都加上下面的答案。

Solution

- 从右往左扫描线，用线段树维护扫描线上每一个点能达到的花的数量，并支持最近篱笆的查询。
- 对于一朵花，找到它上方最近的篱笆，那么它对这中间的每头牛的贡献都是 1。
- 当扫到一个篱笆的右边界时，这中间的答案都要清零。
- 当扫到一个篱笆的左边界时，这中间的答案同理都要清零，但是要向上直到最近的篱笆为止都加上下面的答案。
- 这中间对这个篱笆右下角的贡献会重复计数，因此需要减掉。

Solution

- 从右往左扫描线，用线段树维护扫描线上每一个点能达到的花的数量，并支持最近篱笆的查询。
- 对于一朵花，找到它上方最近的篱笆，那么它对这中间的每头牛的贡献都是 1。
- 当扫到一个篱笆的右边界时，这中间的答案都要清零。
- 当扫到一个篱笆的左边界时，这中间的答案同理都要清零，但是要向上直到最近的篱笆为止都加上下面的答案。
- 这中间对这个篱笆右下角的贡献会重复计数，因此需要减掉。
- 时间复杂度 $O(n \log n)$。

String Reconstruction

有一个小写字符串 S，它包含 n 个字符串片段作为子串。

String Reconstruction

有一个小写字符串 S，它包含 n 个字符串片段作为子串。

每个片段在该串中出现了至少 k_i 次，且这 k_i 次出现位置给定，片段可以相互重叠。

String Reconstruction

有一个小写字符串 S，它包含 n 个字符串片段作为子串。

每个片段在该串中出现了至少 k_i 次，且这 k_i 次出现位置给定，片段可以相互重叠。

已知信息不会矛盾，求字典序最小的满足条件的 S。

String Reconstruction

有一个小写字符串 S，它包含 n 个字符串片段作为子串。

每个片段在该串中出现了至少 k_i 次，且这 k_i 次出现位置给定，片段可以相互重叠。

已知信息不会矛盾，求字典序最小的满足条件的 S。

- $1 \leq n \leq 10^5, \sum k_i \leq 10^6$。

String Reconstruction

有一个小写字符串 S，它包含 n 个字符串片段作为子串。

每个片段在该串中出现了至少 k_i 次，且这 k_i 次出现位置给定，片段可以相互重叠。

已知信息不会矛盾，求字典序最小的满足条件的 S。

- $1 \leq n \leq 10^5, \sum k_i \leq 10^6$。
- $1 \leq l_i \leq r_i \leq 10^6$。

String Reconstruction

有一个小写字符串 S，它包含 n 个字符串片段作为子串。

每个片段在该串中出现了至少 k_i 次，且这 k_i 次出现位置给定，片段可以相互重叠。

已知信息不会矛盾，求字典序最小的满足条件的 S。

- $1 \leq n \leq 10^5, \sum k_i \leq 10^6$。
- $1 \leq l_i \leq r_i \leq 10^6$。
- Source：Codeforces Round #423 Div. 1 A

Solution

- 显然 $|S| = \max r_i$。

Solution

- 显然 $|S| = \max r_i$。
- 将片段填充上，剩下位置的部分全部填 a 即可。

Solution

- 显然 $|S| = \max r_i$。
- 将片段填充上，剩下位置的部分全部填 a 即可。
- 直接填充复杂度为 $O(n^2)$，不能接受。

Solution

- 关键点：在 l_i 处开始填充片段 i，结束位置为 r_i，将 i 这一事件加入 l_i 处的桶中。

Solution

- 关键点：在 l_i 处开始填充片段 i，结束位置为 r_i，将 i 这一事件加入 l_i 处的桶中。
- 从左往右扫描线，用队列维护所有开始填充的片段。

Solution

- 关键点：在 l_i 处开始填充片段 i，结束位置为 r_i，将 i 这一事件加入 l_i 处的桶中。
- 从左往右扫描线，用队列维护所有开始填充的片段。
- 若队首已经结束，那么出队，如此重复直到队列为空或者队首仍然在填充。

Solution

- 关键点：在 l_i 处开始填充片段 i，结束位置为 r_i，将 i 这一事件加入 l_i 处的桶中。
- 从左往右扫描线，用队列维护所有开始填充的片段。
- 若队首已经结束，那么出队，如此重复直到队列为空或者队首仍然在填充。
- 那么根据此时的情况，可以 $O(1)$ 确定当前位置应该填什么字符。

Solution

- 关键点：在 l_i 处开始填充片段 i，结束位置为 r_i，将 i 这一事件加入 l_i 处的桶中。
- 从左往右扫描线，用队列维护所有开始填充的片段。
- 若队首已经结束，那么出队，如此重复直到队列为空或者队首仍然在填充。
- 那么根据此时的情况，可以 $O(1)$ 确定当前位置应该填什么字符。
- 时间复杂度 $O(n)$。

风力观测

维护一个序列 $a_1, a_2, \ldots, a_n$，支持 m 次操作：

风力观测

维护一个序列 $a_1, a_2, \ldots, a_n$，支持 m 次操作：

(1) 将 $a_l, a_{l+1}, \ldots, a_r$ 加上 x。

风力观测

维护一个序列 $a_1, a_2, \ldots, a_n$，支持 m 次操作：

(1) 将 $a_l, a_{l+1}, \ldots, a_r$ 加上 x。

(2) 询问 $|a_p|$ 的历史最大值。

风力观测

维护一个序列 $a_1, a_2, \ldots, a_n$，支持 m 次操作：

(1) 将 $a_l, a_{l+1}, \ldots, a_r$ 加上 x。

(2) 询问 $|a_p|$ 的历史最大值。

- $1 \leq n, m \leq 10^5$。

风力观测

维护一个序列 $a_1, a_2, \ldots, a_n$，支持 m 次操作：

(1) 将 $a_l, a_{l+1}, \ldots, a_r$ 加上 x。

(2) 询问 $|a_p|$ 的历史最大值。

- $1 \leq n, m \leq 10^5$。
- $|a_i|, |x| \leq 10000$。

风力观测

维护一个序列 $a_1, a_2, \ldots, a_n$，支持 m 次操作：

(1) 将 $a_l, a_{l+1}, \ldots, a_r$ 加上 x。

(2) 询问 $|a_p|$ 的历史最大值。

- $1 \leq n, m \leq 10^5$。
- $|a_i|, |x| \leq 10000$。
- Source：“盛大游戏杯”第 15 届上海大学程序设计联赛夏季赛暨上海高校金马五校赛

Solution

- 显然只需要求出 p 这个位置改变量的历史最大值和最小值即可，方便起见我们只讨论最大值。

Solution

- 显然只需要求出 p 这个位置改变量的历史最大值和最小值即可，方便起见我们只讨论最大值。
- 维护历史最大值只需要维护历史标记最大值，这是线段树可以实现的。

Solution

- 显然只需要求出 p 这个位置改变量的历史最大值和最小值即可，方便起见我们只讨论最大值。
- 维护历史最大值只需要维护历史标记最大值，这是线段树可以实现的。
- 只会线段树基本操作，不会维护历史标记最大值？

Solution

- 假设所有询问 p 都相同，不妨换一个角度，对时间建立线段树，即设 s_i 表示时间为 i 时 a_p 的增量。

Solution

- 假设所有询问 p 都相同，不妨换一个角度，对时间建立线段树，即设 s_i 表示时间为 i 时 a_p 的增量。
- 将修改看成两类事件：(1) 在 l_i 处加入修改 i。(2) 在 r_i 处撤销修改 i。

Solution

- 假设所有询问 p 都相同，不妨换一个角度，对时间建立线段树，即设 s_i 表示时间为 i 时 a_p 的增量。
- 将修改看成两类事件：(1) 在 l_i 处加入修改 i。(2) 在 r_i 处撤销修改 i。
- 从左往右扫描线，加入和撤销修改对应时间轴上 s 的一段后缀的区间加减。

Solution

- 假设所有询问 p 都相同，不妨换一个角度，对时间建立线段树，即设 s_i 表示时间为 i 时 a_p 的增量。
- 将修改看成两类事件：(1) 在 l_i 处加入修改 i。(2) 在 r_i 处撤销修改 i。
- 从左往右扫描线，加入和撤销修改对应时间轴上 s 的一段后缀的区间加减。
- 处理所有 $p = i$ 的询问，即为询问时间轴上 s 的一段前缀的区间最大值。

Solution

- 假设所有询问 p 都相同，不妨换一个角度，对时间建立线段树，即设 s_i 表示时间为 i 时 a_p 的增量。
- 将修改看成两类事件：(1) 在 l_i 处加入修改 i。(2) 在 r_i 处撤销修改 i。
- 从左往右扫描线，加入和撤销修改对应时间轴上 s 的一段后缀的区间加减。
- 处理所有 $p = i$ 的询问，即为询问时间轴上 s 的一段前缀的区间最大值。
- 只需要一个支持区间加减、查询区间最大值的线段树。

Solution

- 假设所有询问 p 都相同，不妨换一个角度，对时间建立线段树，即设 s_i 表示时间为 i 时 a_p 的增量。
- 将修改看成两类事件：(1) 在 l_i 处加入修改 i。(2) 在 r_i 处撤销修改 i。
- 从左往右扫描线，加入和撤销修改对应时间轴上 s 的一段后缀的区间加减。
- 处理所有 $p = i$ 的询问，即为询问时间轴上 s 的一段前缀的区间最大值。
- 只需要一个支持区间加减、查询区间最大值的线段树。
- 时间复杂度 $O(n \log n)$。

Match

字符串和括号序列匹配定义为：

Match

字符串和括号序列匹配定义为：

(1) 长度必须相等。

Match

字符串和括号序列匹配定义为：

(1) 长度必须相等。

(2) 对于一对匹配的左括号和右括号 i, j，必须有 $S_i = S_j$。

Match

字符串和括号序列匹配定义为：

(1) 长度必须相等。

(2) 对于一对匹配的左括号和右括号 i, j，必须有 $S_i = S_j$。

给定一个长度为 n 的小写字母串 S，要你构造一个字典序最小的（认为左括号的字典序比右括号小）合法括号序列与 S 匹配，或判断无解。

Match

字符串和括号序列匹配定义为：

(1) 长度必须相等。

(2) 对于一对匹配的左括号和右括号 i, j，必须有 $S_i = S_j$。

给定一个长度为 n 的小写字母串 S，要你构造一个字典序最小的（认为左括号的字典序比右括号小）合法括号序列与 S 匹配，或判断无解。

- $1 \leq n \leq 10^6$。

Match

字符串和括号序列匹配定义为：

(1) 长度必须相等。

(2) 对于一对匹配的左括号和右括号 i, j，必须有 $S_i = S_j$。

给定一个长度为 n 的小写字母串 S，要你构造一个字典序最小的（认为左括号的字典序比右括号小）合法括号序列与 S 匹配，或判断无解。

- $1 \leq n \leq 10^6$。
- Source： CEOI 2016

Solution

- 首先考虑如何判断是否有解。

Solution

- 首先考虑如何判断是否有解。
- 考虑从左往右贪心，维护一个栈，保存所有尚未配对的左括号。

Solution

- 首先考虑如何判断是否有解。
- 考虑从左往右贪心，维护一个栈，保存所有尚未配对的左括号。
- 对于当前字符 c，若栈非空且栈顶字符与待考虑字符相同，则出栈，否则入栈。

Solution

- 首先考虑如何判断是否有解。
- 考虑从左往右贪心，维护一个栈，保存所有尚未配对的左括号。
- 对于当前字符 c，若栈非空且栈顶字符与待考虑字符相同，则出栈，否则入栈。
- 若最后栈非空则无解。

Solution

- 设 f_i 表示考虑 $1..i$ 后栈的情况，可以用 Trie 上的位置来表示。

Solution

- 设 f_i 表示考虑 $1..i$ 后栈的情况，可以用 Trie 上的位置来表示。
- 那么若 $[l, r]$ 合法，则 $f_{l-1} = f_r$。

Solution

- 设 f_i 表示考虑 $1..i$ 后栈的情况，可以用 Trie 上的位置来表示。
- 那么若 $[l, r]$ 合法，则 $f_{l-1} = f_r$。
- 对于区间 $[l, r]$，第一个位置肯定是 (，然后要找到一个最靠右的 x 作为)，满足字符相同且 $f_{l-1} = f_x$。

Solution

- 设 f_i 表示考虑 $1..i$ 后栈的情况，可以用 Trie 上的位置来表示。
- 那么若 $[l, r]$ 合法，则 $f_{l-1} = f_r$。
- 对于区间 $[l, r]$，第一个位置肯定是 (，然后要找到一个最靠右的 x 作为)，满足字符相同且 $f_{l-1} = f_x$。
- 然后递归处理区间 $[l+1, x-1]$ 和 $[x+1, r]$，即可得到字典序最小的解。

Solution

- 设 f_i 表示考虑 $1..i$ 后栈的情况，可以用 Trie 上的位置来表示。
- 那么若 $[l, r]$ 合法，则 $f_{l-1} = f_r$。
- 对于区间 $[l, r]$，第一个位置肯定是 (，然后要找到一个最靠右的 x 作为)，满足字符相同且 $f_{l-1} = f_x$。
- 然后递归处理区间 $[l+1, x-1]$ 和 $[x+1, r]$，即可得到字典序最小的解。
- 瓶颈在于找到最靠右的 x，注意到每个询问产生的两个子区间的右端点不会比当前询问大，从这里入手优化。

Solution

- 从右往左扫描线，按 r 从大到小依次处理每个区间。

Solution

- 从右往左扫描线，按 r 从大到小依次处理每个区间。
- 在扫描线的过程中维护 $v_{i,j}$ 表示扫描线左侧第一个满足 f 为 i 且字符为 j 的位置，当扫描线往左移时只需要将其设置为其前驱的位置。

Solution

- 从右往左扫描线，按 r 从大到小依次处理每个区间。
- 在扫描线的过程中维护 $v_{i,j}$ 表示扫描线左侧第一个满足 f 为 i 且字符为 j 的位置，当扫描线往左移时只需要将其设置为其前驱的位置。
- 对于每个询问，只需要访问 v 数组，就可以 $O(1)$ 得到 x。

Solution

- 从右往左扫描线，按 r 从大到小依次处理每个区间。
- 在扫描线的过程中维护 $v_{i,j}$ 表示扫描线左侧第一个满足 f 为 i 且字符为 j 的位置，当扫描线往左移时只需要将其设置为其前驱的位置。
- 对于每个询问，只需要访问 v 数组，就可以 $O(1)$ 得到 x。
- 每个询问产生的两个子区间的右端点不会比当前询问大，因此对每个位置开桶就可以维护。

Solution

- 从右往左扫描线，按 r 从大到小依次处理每个区间。
- 在扫描线的过程中维护 $v_{i,j}$ 表示扫描线左侧第一个满足 f 为 i 且字符为 j 的位置，当扫描线往左移时只需要将其设置为其前驱的位置。
- 对于每个询问，只需要访问 v 数组，就可以 $O(1)$ 得到 x。
- 每个询问产生的两个子区间的右端点不会比当前询问大，因此对每个位置开桶就可以维护。
- 时间复杂度 $O(n)$。

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)
- [JLoi2016] 圆的异或并 (BZOJ 4561)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)
- [JLoi2016] 圆的异或并 (BZOJ 4561)
- [SCOI2012]Blinker 的噩梦 (BZOJ 2758)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)
- [JLoi2016] 圆的异或并 (BZOJ 4561)
- [SCOI2012]Blinker 的噩梦 (BZOJ 2758)
- Jewel Heist (CERC 2012)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)
- [JLoi2016] 圆的异或并 (BZOJ 4561)
- [SCOI2012]Blinker 的噩梦 (BZOJ 2758)
- Jewel Heist (CERC 2012)
- Visual Python++ (WF 2017)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)
- [JLoi2016] 圆的异或并 (BZOJ 4561)
- [SCOI2012]Blinker 的噩梦 (BZOJ 2758)
- Jewel Heist (CERC 2012)
- Visual Python++ (WF 2017)
- 布娃娃 (BZOJ 2161)

课后习题

- Tourists (Codeforces Round #176 Div. 1 D)
- Oil (WF 2016)
- [JLoi2016] 圆的异或并 (BZOJ 4561)
- [SCOI2012]Blinker 的噩梦 (BZOJ 2758)
- Jewel Heist (CERC 2012)
- Visual Python++ (WF 2017)
- 布娃娃 (BZOJ 2161)
- [Zjoi2016] 大森林 (BZOJ 4573)

Thank you!