DP 杂题选讲

Claris

Hangzhou Dianzi University

2016 年 2 月 9 日

Nim z utrudnieniem

A 和 B 两个人玩游戏，一共有 m 颗石子，A 把它们分成了 n 堆，每堆石子数分别为 $a_1, a_2, ..., a_n$。每轮可以选择一堆石子，取掉任意颗石子，但不能不取。谁先不能操作，谁就输了。

Nim z utrudnieniem

A 和 B 两个人玩游戏，一共有 m 颗石子，A 把它们分成了 n 堆，每堆石子数分别为 $a_1, a_2, ..., a_n$。每轮可以选择一堆石子，取掉任意颗石子，但不能不取。谁先不能操作，谁就输了。

在游戏开始前，B 可以扔掉若干堆石子，但是必须保证扔掉的堆数是 d 的倍数，且不能扔掉所有石子。A 先手，请问 B 有多少种扔的方式，使得 B 能够获胜。

Nim z utrudnieniem

A 和 B 两个人玩游戏，一共有 m 颗石子，A 把它们分成了 n 堆，每堆石子数分别为 $a_1, a_2, ..., a_n$。每轮可以选择一堆石子，取掉任意颗石子，但不能不取。谁先不能操作，谁就输了。

在游戏开始前，B 可以扔掉若干堆石子，但是必须保证扔掉的堆数是 d 的倍数，且不能扔掉所有石子。A 先手，请问 B 有多少种扔的方式，使得 B 能够获胜。

- $1 \le n \le 500000, 1 \le d \le 10, 1 \le a_i \le 10^6, m \le 10^7$。

Nim z utrudnieniem

A 和 B 两个人玩游戏，一共有 m 颗石子，A 把它们分成了 n 堆，每堆石子数分别为 $a_1, a_2, ..., a_n$。每轮可以选择一堆石子，取掉任意颗石子，但不能不取。谁先不能操作，谁就输了。

在游戏开始前，B 可以扔掉若干堆石子，但是必须保证扔掉的堆数是 d 的倍数，且不能扔掉所有石子。A 先手，请问 B 有多少种扔的方式，使得 B 能够获胜。

- $1 \leq n \leq 500000, 1 \leq d \leq 10, 1 \leq a_i \leq 10^6, m \leq 10^7$。
- Memory Limit: 64MB

Nim z utrudnieniem

A 和 B 两个人玩游戏，一共有 m 颗石子，A 把它们分成了 n 堆，每堆石子数分别为 $a_1, a_2, ..., a_n$。每轮可以选择一堆石子，取掉任意颗石子，但不能不取。谁先不能操作，谁就输了。

在游戏开始前，B 可以扔掉若干堆石子，但是必须保证扔掉的堆数是 d 的倍数，且不能扔掉所有石子。A 先手，请问 B 有多少种扔的方式，使得 B 能够获胜。

- $1 \leq n \leq 500000, 1 \leq d \leq 10, 1 \leq a_i \leq 10^6, m \leq 10^7$。
- Memory Limit: 64MB
- Source: POI 2016

Nim z utrudnieniem

- B 获胜的充要条件为剩下的石子每堆数量的异或和为 0。

Nim z utrudnieniem

- B 获胜的充要条件为剩下的石子每堆数量的异或和为 0。
- 设 $f(i,j,k)$ 表示考虑了前 i 堆的石子，当前扔掉的堆数模 d 为 j，没有扔掉的石子的异或和为 k 的方案数，转移显然。最后注意特判 n 是 d 的倍数的情况，此时答案应该减去 1。

Nim z utrudnieniem

- B 获胜的充要条件为剩下的石子每堆数量的异或和为 0。
- 设 $f(i, j, k)$ 表示考虑了前 i 堆的石子，当前扔掉的堆数模 d 为 j，没有扔掉的石子的异或和为 k 的方案数，转移显然。最后注意特判 n 是 d 的倍数的情况，此时答案应该减去 1。
- 时间复杂度 $O(nd\max(a_i))$，不能承受。

Nim z utrudnieniem

- 将石子数从小到大排序，那么有效状态数降为 $O(md)$，可以承受。

Nim z utrudnieniem

- 将石子数从小到大排序，那么有效状态数降为 $O(md)$，可以承受。
- 即使用滚动数组优化，空间使用也达到了大约 80MB，不能承受。

Nim z utrudnieniem

- 将石子数从小到大排序，那么有效状态数降为 $O(md)$，可以承受。
- 即使用滚动数组优化，空间使用也达到了大约 80MB，不能承受。
- 注意到 $f(i,j,k)$ 和 $f(i,j,k\ xor\ a_i)$ 的转移是互补的，于是可以同时处理，省去滚动数组，直接做到原地 DP，当然对于 $f(i,0,k)$ 要特别处理。

Nim z utrudnieniem

- 将石子数从小到大排序，那么有效状态数降为 $O(md)$，可以承受。
- 即使用滚动数组优化，空间使用也达到了大约 80MB，不能承受。
- 注意到 $f(i, j, k)$ 和 $f(i, j, k\ xor\ a_i)$ 的转移是互补的，于是可以同时处理，省去滚动数组，直接做到原地 DP，当然对于 $f(i, 0, k)$ 要特别处理。
- 空间使用大约为 40MB，可以承受。

Badania naukowe

给定三个数字串 A,B,C，请找到一个 A,B 的最长公共子序列，满足 C 是该子序列的子串。

Badania naukowe

给定三个数字串 A,B,C，请找到一个 A,B 的最长公共子序列，满足 C 是该子序列的子串。

- $0 \leq n \leq 3000$。

Badania naukowe

给定三个数字串 A,B,C，请找到一个 A,B 的最长公共子序列，满足 C 是该子序列的子串。

- $0 \leq n \leq 3000$。
- Source：ONTAK 2015

Badania naukowe

- 设 $f(i,j)$ 为 $A[1..i]$ 与 $B[1..j]$ 的 LCS，$g(i,j)$ 为 $A[i..n]$ 与 $B[j..m]$ 的 LCS。

Badania naukowe

- 设 $f(i,j)$ 为 $A[1..i]$ 与 $B[1..j]$ 的 LCS，$g(i,j)$ 为 $A[i..n]$ 与 $B[j..m]$ 的 LCS。
- 若 C 为空串，则 $ans = f(n,m)$。

- 设 $f(i,j)$ 为 $A[1..i]$ 与 $B[1..j]$ 的 LCS，$g(i,j)$ 为 $A[i..n]$ 与 $B[j..m]$ 的 LCS。
- 若 C 为空串，则 $ans = f(n,m)$。
- 否则，设 d_i 为 A 中从 i 开始往右匹配上 C 串的结束位置，e_i 为 B 中从 i 开始往右匹配上 C 串的结束位置。

Badania naukowe

- 设 $f(i,j)$ 为 $A[1..i]$ 与 $B[1..j]$ 的 LCS，$g(i,j)$ 为 $A[i..n]$ 与 $B[j..m]$ 的 LCS。
- 若 C 为空串，则 $ans = f(n,m)$。
- 否则，设 d_i 为 A 中从 i 开始往右匹配上 C 串的结束位置，e_i 为 B 中从 i 开始往右匹配上 C 串的结束位置。
- 那么此时 $ans = \max(f(i-1,j-1) + g(d_i+1, e_j+1) + |C|)$。

Badania naukowe

- 设 $f(i,j)$ 为 $A[1..i]$ 与 $B[1..j]$ 的 LCS，$g(i,j)$ 为 $A[i..n]$ 与 $B[j..m]$ 的 LCS。
- 若 C 为空串，则 $ans = f(n,m)$。
- 否则，设 d_i 为 A 中从 i 开始往右匹配上 C 串的结束位置，e_i 为 B 中从 i 开始往右匹配上 C 串的结束位置。
- 那么此时 $ans = \max(f(i-1,j-1) + g(d_i+1, e_j+1) + |C|)$。
- 时间复杂度 $O(n^2)$。

Myjnie

有 n 家洗车店从左往右排成一排，每家店都有一个正整数价格 p_i。

Myjnie

有 n 家洗车店从左往右排成一排，每家店都有一个正整数价格 p_i。

有 m 个人要来消费，第 i 个人会驶过第 a_i 个开始一直到第 b_i 个洗车店，且会选择这些店中最便宜的一个进行一次消费。但是如果这个最便宜的价格大于 c_i，那么这个人就不洗车了。

Myjnie

有 n 家洗车店从左往右排成一排，每家店都有一个正整数价格 p_i。

有 m 个人要来消费，第 i 个人会驶过第 a_i 个开始一直到第 b_i 个洗车店，且会选择这些店中最便宜的一个进行一次消费。但是如果这个最便宜的价格大于 c_i，那么这个人就不洗车了。

请给每家店指定一个价格，使得所有人花的钱的总和最大。

Myjnie

有 n 家洗车店从左往右排成一排，每家店都有一个正整数价格 p_i。

有 m 个人要来消费，第 i 个人会驶过第 a_i 个开始一直到第 b_i 个洗车店，且会选择这些店中最便宜的一个进行一次消费。但是如果这个最便宜的价格大于 c_i，那么这个人就不洗车了。

请给每家店指定一个价格，使得所有人花的钱的总和最大。

- $1 \le n \le 50, 1 \le m \le 4000, 1 \le a_i \le b_i \le n, 1 \le c_i \le 500000$。

Myjnie

有 n 家洗车店从左往右排成一排，每家店都有一个正整数价格 p_i。

有 m 个人要来消费，第 i 个人会驶过第 a_i 个开始一直到第 b_i 个洗车店，且会选择这些店中最便宜的一个进行一次消费。但是如果这个最便宜的价格大于 c_i，那么这个人就不洗车了。

请给每家店指定一个价格，使得所有人花的钱的总和最大。

- $1 \le n \le 50, 1 \le m \le 4000, 1 \le a_i \le b_i \le n, 1 \le c_i \le 500000$。
- Source： POI 2015

Myjnie

- 首先将 c 离散化，考虑区间 DP，设 $f(i,j,k)$ 为区间 $[i,j]$ 最小值为 k 时的最大收益。

Myjnie

- 首先将 c 离散化，考虑区间 DP，设 $f(i,j,k)$ 为区间 $[i,j]$ 最小值为 k 时的最大收益。
- 转移时枚举最小值所在位置 x，那么可以用 $f(i,x-1,\geq k)+f(x+1,j,\geq k)+cost(x)$ 来更新 $f(i,j,k)$。

Myjnie

- 首先将 c 离散化，考虑区间 DP，设 $f(i,j,k)$ 为区间 $[i,j]$ 最小值为 k 时的最大收益。
- 转移时枚举最小值所在位置 x，那么可以用 $f(i,x-1,\geq k)+f(x+1,j,\geq k)+cost(x)$ 来更新 $f(i,j,k)$。
- 时间复杂度 $O(n^3m)$。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

- $1 \le n, m \le 100000$。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

- $1 \leq n, m \leq 100000$。
- Source： quailty's Contest #1

组队活动

- 设 f_i 表示 i 个人自由分组的方案数，初始值 $f_0 = 1$。

组队活动

- 设 f_i 表示 i 个人自由分组的方案数，初始值 $f_0 = 1$。
- 考虑枚举第 i 个人所在队伍剩下的人数，假设有 j 人，那么有 $f_{i-j-1}C(i-1,j)$ 种方案，所以

$$
\begin{aligned}
f_i &= \sum_{j=0}^{\min(i,m)-1} f_{i-j-1}C(i-1,j) \\
&= (i-1)! \sum_{j=0}^{\min(i,m)-1} \frac{f_{i-j-1}}{(i-j-1)!} \times \frac{1}{j!}
\end{aligned}
$$

组队活动

- 设 f_i 表示 i 个人自由分组的方案数，初始值 $f_0=1$。
- 考虑枚举第 i 个人所在队伍剩下的人数，假设有 j 人，那么有 $f_{i-j-1}C(i-1,j)$ 种方案，所以

$$
\begin{aligned}
f_i &= \sum_{j=0}^{\min(i,m)-1} f_{i-j-1}C(i-1,j) \\
&= (i-1)! \sum_{j=0}^{\min(i,m)-1} \frac{f_{i-j-1}}{(i-j-1)!} \times \frac{1}{j!}
\end{aligned}
$$

- 这个转移显然是个卷积的形式，用分治 NTT 即可做到 $O(n\log^2 n)$。

Tree chain problem

给定一棵有 n 个点的树，以及 m 条树链，其中第 i 条树链的价值为 w_i，请选择一些没有公共点的树链，使得价值和最大。

Tree chain problem

给定一棵有 n 个点的树，以及 m 条树链，其中第 i 条树链的价值为 w_i，请选择一些没有公共点的树链，使得价值和最大。

- $1 \le n, m \le 100000$。

Tree chain problem

给定一棵有 n 个点的树，以及 m 条树链，其中第 i 条树链的价值为 w_i，请选择一些没有公共点的树链，使得价值和最大。

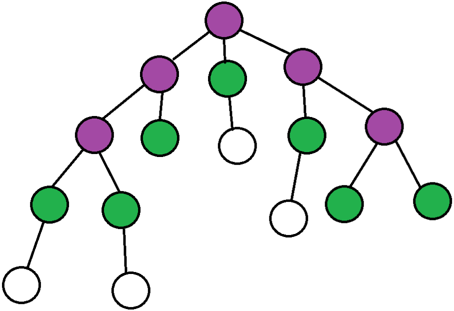
- $1 \le n, m \le 100000$。
- Source：2015 Multi-University Training Contest 1

Tree chain problem

- 考虑树形 DP，设 $f(x)$ 为以 x 为根的子树内选取不相交树链的价值和的最大值，枚举一条 LCA 为 x 的链 u, v, w，那么当前方案的价值为 $w+$ 去除 u 到 v 路径上的点后深度最小的点的 f 的和。

Tree chain problem

- 考虑树形 DP，设 $f(x)$ 为以 x 为根的子树内选取不相交树链的价值和的最大值，枚举一条 LCA 为 x 的链 u, v, w，那么当前方案的价值为 $w+$ 去除 u 到 v 路径上的点后深度最小的点的 f 的和。
- 如图，紫色部分为 u 到 v 路径上的点，绿色部分计入当前贡献：



Tree chain problem

- 设 $g(x)$ 为 x 所有孩子的 f 的和，那么对于一条 LCA 为 x 的链 u, v, w，当前方案的价值为 $w + \sum(g_y - f_y)$，其中 y 为 u 到 v 路径上的点。

Tree chain problem

- 设 $g(x)$ 为 x 所有孩子的 f 的和，那么对于一条 LCA 为 x 的链 u, v, w，当前方案的价值为 $w + \sum(g_y - f_y)$，其中 y 为 u 到 v 路径上的点。
- 这就变成了单点修改，链和查询问题，按 dfs 序维护树状数组即可。

Tree chain problem

- 设 $g(x)$ 为 x 所有孩子的 f 的和，那么对于一条 LCA 为 x 的链 u, v, w，当前方案的价值为 $w + \sum(g_y - f_y)$，其中 y 为 u 到 v 路径上的点。
- 这就变成了单点修改，链和查询问题，按 dfs 序维护树状数组即可。
- 时间复杂度 $O((n+m)\log n)$。

Bombing plan

给定一棵 n 个点的无根树，节点 i 的权值为 w_i，攻击节点 i 的同时，会把所有与 i 节点距离在 w_i 内的节点全部摧毁，任意一边的长度都为 1。

Bombing plan

给定一棵 n 个点的无根树，节点 i 的权值为 w_i，攻击节点 i 的同时，会把所有与 i 节点距离在 w_i 内的节点全部摧毁，任意一边的长度都为 1。

问至少需要几次攻击才能把整个树的节点全部摧毁。

Bombing plan

给定一棵 n 个点的无根树，节点 i 的权值为 w_i，攻击节点 i 的同时，会把所有与 i 节点距离在 w_i 内的节点全部摧毁，任意一边的长度都为 1。

问至少需要几次攻击才能把整个树的节点全部摧毁。

- $1 \le n \le 100000, 0 \le w_i \le 100$。

Bombing plan

给定一棵 n 个点的无根树，节点 i 的权值为 w_i，攻击节点 i 的同时，会把所有与 i 节点距离在 w_i 内的节点全部摧毁，任意一边的长度都为 1。

问至少需要几次攻击才能把整个树的节点全部摧毁。

- $1 \leq n \leq 100000, 0 \leq w_i \leq 100$。
- Source：2015 Multi-University Training Contest 1

Bombing plan

- 设 $f(i,j)$ 为以 i 为根的子树内的点全部被摧毁，并且还能向上破坏至多 j 个距离的最小代价，$g(i,j)$ 为以 i 为根的子树内的点还没被全部摧毁，且子树中未破坏的点离 i 点的距离最多为 j 的最小代价。

Bombing plan

- 设 $f(i,j)$ 为以 i 为根的子树内的点全部被摧毁，并且还能向上破坏至多 j 个距离的最小代价，$g(i,j)$ 为以 i 为根的子树内的点还没被全部摧毁，且子树中未破坏的点离 i 点的距离最多为 j 的最小代价。
- 如果要攻击 i 点，设 y 是 i 的孩子，那么有：$f(i,w_i) = 1 + \sum \min(f(y, 0..w_i), g(y, 0..w_i - 1))$。

Bombing plan

- 如果不攻击 i 点，考虑枚举 i 的一个孩子 u，y 是除 u 之外的其它孩子，那么有：
  $f(i,j) = min(f(i,j), f(u,j+1) + \sum \min(f(y, 0..j+1), g(y, 0..j)))$
  $g(i,j) = min(g(i,j), g(u,j-1) + \sum \min(f(y, 0..j), g(y, 0..j-1)))$

Bombing plan

- 如果不攻击 i 点，考虑枚举 i 的一个孩子 u，y 是除 u 之外的其它孩子，那么有：
  $f(i,j) = min(f(i,j), f(u,j+1) + \sum \mathrm{min}(f(y,0..j+1), g(y,0..j)))$
  $g(i,j) = min(g(i,j), g(u,j-1) + \sum \mathrm{min}(f(y,0..j), g(y,0..j-1)))$
- 对于每个 i，在 DP 完之后扫描所有 $f(i,j)$ 以及 $g(i,j)$，维护其单调性。

Bombing plan

- 如果不攻击 i 点，考虑枚举 i 的一个孩子 u，y 是除 u 之外的其它孩子，那么有：

  $f(i,j) = min(f(i,j), f(u,j+1) + \sum \min(f(y,0..j+1), g(y,0..j)))$

  $g(i,j) = min(g(i,j), g(u,j-1) + \sum \min(f(y,0..j), g(y,0..j-1)))$
- 对于每个 i，在 DP 完之后扫描所有 $f(i,j)$ 以及 $g(i,j)$，维护其单调性。
- 通过维护前缀最小值可以将转移优化到 $O(nw)$。

Dominating Set

给定一个 n 个点，m 条边的二分图，请选择其中的一些点，使得对于任意一个点，要么它被选择，要么与它相邻的点中至少有一个点被选择。

Dominating Set

给定一个 n 个点，m 条边的二分图，请选择其中的一些点，使得对于任意一个点，要么它被选择，要么与它相邻的点中至少有一个点被选择。

求所有可以选的集合的方案数。

Dominating Set

给定一个 n 个点，m 条边的二分图，请选择其中的一些点，使得对于任意一个点，要么它被选择，要么与它相邻的点中至少有一个点被选择。

求所有可以选的集合的方案数。

- $1 \leq n \leq 30, 0 \leq m \leq 225$。

Dominating Set

给定一个 n 个点，m 条边的二分图，请选择其中的一些点，使得对于任意一个点，要么它被选择，要么与它相邻的点中至少有一个点被选择。

求所有可以选的集合的方案数。

- $1 \le n \le 30, 0 \le m \le 225$。
- Source：Grand Prix of China

Dominating Set

- 显然每个连通块互不影响，对每个连通块算出方案数后乘起来即可得到答案。

Dominating Set

- 显然每个连通块互不影响，对每个连通块算出方案数后乘起来即可得到答案。
- 对于一个连通块，将其黑白染色，并找到点数较小的那一侧，设点数为 t，那么有 $t \leq 15$。

Dominating Set

- 显然每个连通块互不影响，对每个连通块算出方案数后乘起来即可得到答案。
- 对于一个连通块，将其黑白染色，并找到点数较小的那一侧，设点数为 t，那么有 $t \leq 15$。
- 不妨设黑色为小的那侧，暴力枚举这一侧的点的选择情况，我们需要额外选取一些白色的点，使得黑色点中未选择的点都被覆盖到。

Dominating Set

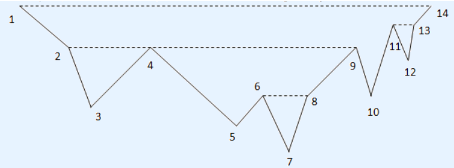
- 显然每个连通块互不影响，对每个连通块算出方案数后乘起来即可得到答案。
- 对于一个连通块，将其黑白染色，并找到点数较小的那一侧，设点数为 t，那么有 $t \le 15$。
- 不妨设黑色为小的那侧，暴力枚举这一侧的点的选择情况，我们需要额外选取一些白色的点，使得黑色点中未选择的点都被覆盖到。
- 设黑色点选了 u 个，现在的问题就等价于，给定 k 个 $t-u$ 位的二进制数，求并集为满集的方案数，直接 $O(k2^{t-u})$ 的 DP 即可解决。

Dominating Set

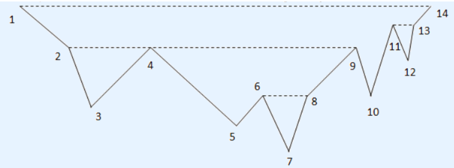
- 显然每个连通块互不影响，对每个连通块算出方案数后乘起来即可得到答案。
- 对于一个连通块，将其黑白染色，并找到点数较小的那一侧，设点数为 t，那么有 $t \leq 15$。
- 不妨设黑色为小的那侧，暴力枚举这一侧的点的选择情况，我们需要额外选取一些白色的点，使得黑色点中未选择的点都被覆盖到。
- 设黑色点选了 u 个，现在的问题就等价于，给定 k 个 $t-u$ 位的二进制数，求并集为满集的方案数，直接 $O(k2^{t-u})$ 的 DP 即可解决。
- 时间复杂度相当于枚举一个集合，再枚举它的子集的复杂度，即 $O(n3^{\frac{n}{2}})$。

Aerial Tramway

给定 n 个点 (x_i, y_i)，形成一条折线，满足 $x_i, y_i > 0$，$x_i < x_{i+1}$，$y_i \neq y_{i+1}$，如下图：



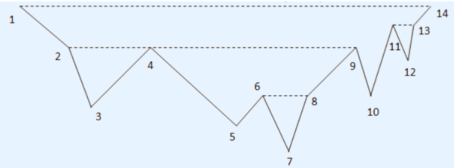
给定 n 个点 (x_i, y_i)，形成一条折线，满足 $x_i, y_i > 0$，$x_i < x_{i+1}$，$y_i \neq y_{i+1}$，如下图：



对于两个点，如果它们高度相等，且中间所有点都严格低于这两个点，那么你可以在两点间连一条边。

Aerial Tramway

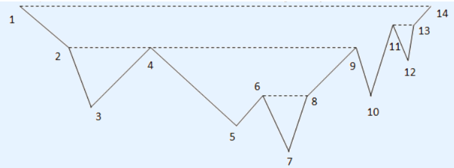
给定 n 个点 (x_i, y_i)，形成一条折线，满足 $x_i, y_i > 0$，$x_i < x_{i+1}$，$y_i \neq y_{i+1}$，如下图：



对于两个点，如果它们高度相等，且中间所有点都严格低于这两个点，那么你可以在两点间连一条边。

你需要连恰好 m 条边，满足每个点严格低于不超过 k 条边，请最大化所有边的长度和。

给定 n 个点 (x_i, y_i)，形成一条折线，满足 $x_i, y_i > 0$，$x_i < x_{i+1}$，$y_i \neq y_{i+1}$，如下图：



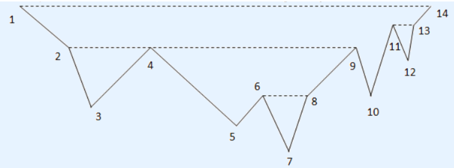
对于两个点，如果它们高度相等，且中间所有点都严格低于这两个点，那么你可以在两点间连一条边。

你需要连恰好 m 条边，满足每个点严格低于不超过 k 条边，请最大化所有边的长度和。

- $1 \leq n, m \leq 200, 2 \leq k \leq 10$。

Aerial Tramway

给定 n 个点 (x_i, y_i)，形成一条折线，满足 $x_i, y_i > 0$，$x_i < x_{i+1}$，$y_i \neq y_{i+1}$，如下图：



对于两个点，如果它们高度相等，且中间所有点都严格低于这两个点，那么你可以在两点间连一条边。

你需要连恰好 m 条边，满足每个点严格低于不超过 k 条边，请最大化所有边的长度和。

- $1 \le n, m \le 200, 2 \le k \le 10$。
- Source：2015 年湖南省大学生程序设计竞赛

Aerial Tramway

- 可以连的边数不超过 n。

Aerial Tramway

- 可以连的边数不超过 n。
- 对于一条边，若它仅连接 i 和 $i+1$，那么它不受 k 的限制，把这种边额外取出，按长度从大到小排序。

Aerial Tramway

- 可以连的边数不超过 n。
- 对于一条边，若它仅连接 i 和 $i+1$，那么它不受 k 的限制，把这种边额外取出，按长度从大到小排序。
- 剩下的边两两之间要么是包含关系，要么没有任何交集，按照包含关系可以建出一棵树。

Aerial Tramway

- 设 $f(i,j,k)$ 表示以 i 为根的子树中修建了 j 个缆车，且该区间内所有点上面的缆车数的最大值为 k 的最优解。

Aerial Tramway

- 设 $f(i,j,k)$ 表示以 i 为根的子树中修建了 j 个缆车，且该区间内所有点上面的缆车数的最大值为 k 的最优解。
- 对于 x 的孩子 i，通过枚举 a,b,c,d，可以得到：
$f'(x, a+c, \max(b,d)) = \max(f(x,a,b) + f(i,c,d))$

Aerial Tramway

- 设 $f(i,j,k)$ 表示以 i 为根的子树中修建了 j 个缆车，且该区间内所有点上面的缆车数的最大值为 k 的最优解。
- 对于 x 的孩子 i，通过枚举 a,b,c,d，可以得到：$f'(x,a+c,\max(b,d)) = \max(f(x,a,b)+f(i,c,d))$
- 可以通过讨论 b 和 d 的大小关系，维护两个前缀最大值来省去 d 的枚举。

Aerial Tramway

- 设 $f(i,j,k)$ 表示以 i 为根的子树中修建了 j 个缆车，且该区间内所有点上面的缆车数的最大值为 k 的最优解。
- 对于 x 的孩子 i，通过枚举 a,b,c,d，可以得到：
  $f'(x,a+c,\max(b,d)) = \max(f(x,a,b)+f(i,c,d))$
- 可以通过讨论 b 和 d 的大小关系，维护两个前缀最大值来省去 d 的枚举。
- 如果把 a 的上界定为 x 之前部分的子树大小，c 的上界定为 i 的子树大小，那么对于树上的两个点，只会在它们的 LCA 处被 DP 到，所以这个树形 DP 的时间复杂度为 $O(kn^2)$。

Aerial Tramway

- 设 $f(i,j,k)$ 表示以 i 为根的子树中修建了 j 个缆车，且该区间内所有点上面的缆车数的最大值为 k 的最优解。
- 对于 x 的孩子 i，通过枚举 a,b,c,d，可以得到：
  $f'(x,a+c,\max(b,d))=\max(f(x,a,b)+f(i,c,d))$
- 可以通过讨论 b 和 d 的大小关系，维护两个前缀最大值来省去 d 的枚举。
- 如果把 a 的上界定为 x 之前部分的子树大小，c 的上界定为 i 的子树大小，那么对于树上的两个点，只会在它们的 LCA 处被 DP 到，所以这个树形 DP 的时间复杂度为 $O(kn^2)$。
- 最后统计答案时，只要枚举受 k 限制的边数和不受 k 限制的边数，取个最大值即可。

Rozstaw szyn

给定一棵有 n 个点，m 个叶子节点的树，其中 m 个叶子节点分别为 1 到 m 号点，每个叶子节点有一个权值 r_i。

Rozstaw szyn

给定一棵有 n 个点，m 个叶子节点的树，其中 m 个叶子节点分别为 1 到 m 号点，每个叶子节点有一个权值 r_i。

你需要给剩下 $n-m$ 个点各指定一个权值，使得树上相邻两个点的权值差的绝对值之和最小。

Rozstaw szyn

给定一棵有 n 个点，m 个叶子节点的树，其中 m 个叶子节点分别为 1 到 m 号点，每个叶子节点有一个权值 r_i。

你需要给剩下 $n-m$ 个点各指定一个权值，使得树上相邻两个点的权值差的绝对值之和最小。

- $2 \le m \le n \le 500000$。

Rozstaw szyn

给定一棵有 n 个点，m 个叶子节点的树，其中 m 个叶子节点分别为 1 到 m 号点，每个叶子节点有一个权值 r_i。

你需要给剩下 $n-m$ 个点各指定一个权值，使得树上相邻两个点的权值差的绝对值之和最小。

- $2 \le m \le n \le 500000$。
- Source：PA 2015

Rozstaw szyn

- 将叶子一层层剥去，可以得到一棵有根树。

Rozstaw szyn

- 将叶子一层层剥去，可以得到一棵有根树。
- 对于所有儿子都是叶子的点，显然它取中位数时最优，所以最优解中每个点的取值范围是一段连续的区间。

Rozstaw szyn

- 将叶子一层层剥去，可以得到一棵有根树。
- 对于所有儿子都是叶子的点，显然它取中位数时最优，所以最优解中每个点的取值范围是一段连续的区间。
- 从底向上 DP，设 $fl(x)$ 表示 x 点的最小可行值，$fr(x)$ 表示最大可行值。

Rozstaw szyn

- 将叶子一层层剥去，可以得到一棵有根树。
- 对于所有儿子都是叶子的点，显然它取中位数时最优，所以最优解中每个点的取值范围是一段连续的区间。
- 从底向上 DP，设 $fl(x)$ 表示 x 点的最小可行值，$fr(x)$ 表示最大可行值。
- $fl(x), fr(x)$ 只可能是儿子的左右端点，对儿子作扫描线即可得到。

Rozstaw szyn

- 将叶子一层层剥去，可以得到一棵有根树。
- 对于所有儿子都是叶子的点，显然它取中位数时最优，所以最优解中每个点的取值范围是一段连续的区间。
- 从底向上 DP，设 $fl(x)$ 表示 x 点的最小可行值，$fr(x)$ 表示最大可行值。
- $fl(x), fr(x)$ 只可能是儿子的左右端点，对儿子作扫描线即可得到。
- 时间复杂度 $O(n \log n)$。

Shopping

给定一棵有 n 个点的树，第 i 个点有 d_i 件商品，价格为 c_i，价值为 w_i。

Shopping

给定一棵有 n 个点的树，第 i 个点有 d_i 件商品，价格为 c_i，价值为 w_i。

你手头有 m 块钱，且你要保证你买过的点在树上互相连通，问买到的物品的总价值最多是多少。

Shopping

给定一棵有 n 个点的树，第 i 个点有 d_i 件商品，价格为 c_i，价值为 w_i。

你手头有 m 块钱，且你要保证你买过的点在树上互相连通，问买到的物品的总价值最多是多少。

- $1 \le n \le 500, 1 \le m \le 4000, d_i \le 100$。

Shopping

给定一棵有 n 个点的树，第 i 个点有 d_i 件商品，价格为 c_i，价值为 w_i。

你手头有 m 块钱，且你要保证你买过的点在树上互相连通，问买到的物品的总价值最多是多少。

- $1 \leq n \leq 500, 1 \leq m \leq 4000, d_i \leq 100$。
- Source：BZOJ 4182

Shopping

- 考虑某个点必选的情况，只要以这个点为根进行树形依赖背包即可，具体做法不再赘述，通过二进制拆分可以做到 $O(nm \log d)$。

Shopping

- 考虑某个点必选的情况，只要以这个点为根进行树形依赖背包即可，具体做法不再赘述，通过二进制拆分可以做到 $O(nm\log d)$。
- 对这棵树进行点分治，重心要么选，要么不选，第一种情况用上述方法 DP 即可，第二种情况则是子问题，递归处理即可。

Shopping

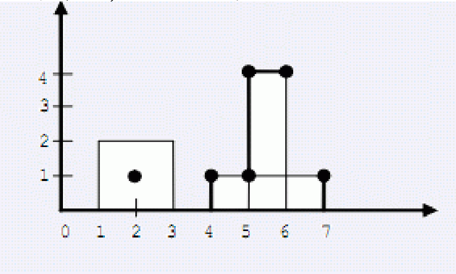
- 考虑某个点必选的情况，只要以这个点为根进行树形依赖背包即可，具体做法不再赘述，通过二进制拆分可以做到 $O(nm\log d)$。
- 对这棵树进行点分治，重心要么选，要么不选，第一种情况用上述方法 DP 即可，第二种情况则是子问题，递归处理即可。
- 时间复杂度 $O(nm\log n\log d)$。

Photo

平面上有 n 个点，要求用最少的底边在 x 轴上且面积不超过 A 的矩形覆盖所有点，这些矩形可以重叠。

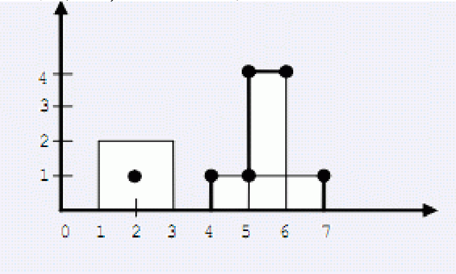
Photo

平面上有 n 个点，要求用最少的底边在 x 轴上且面积不超过 A 的矩形覆盖所有点，这些矩形可以重叠。



Photo

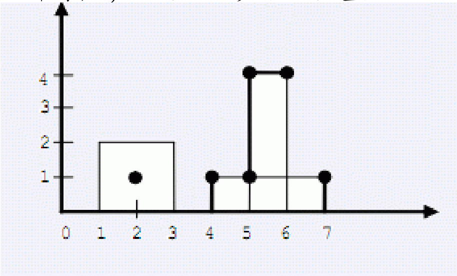
平面上有 n 个点，要求用最少的底边在 x 轴上且面积不超过 A 的矩形覆盖所有点，这些矩形可以重叠。



- $1 \le n \le 100, 1 \le A \le 200000, 0 \le x \le 3000000, 1 \le y \le A$。

Photo

平面上有 n 个点，要求用最少的底边在 x 轴上且面积不超过 A 的矩形覆盖所有点，这些矩形可以重叠。



- $1 \le n \le 100, 1 \le A \le 200000, 0 \le x \le 3000000, 1 \le y \le A$。
- Source: CEOI 2009

Photo

- 如果两个矩形相交且不是包含关系，那么完全可以让它们不相交。

Photo

- 如果两个矩形相交且不是包含关系，那么完全可以让它们不相交。
- 将坐标离散化后，设 $f(i,j,k)$ 表示区间 $[i,j]$ 纵坐标不小于 k 的部分的最优解。

Photo

- 如果两个矩形相交且不是包含关系，那么完全可以让它们不相交。
- 将坐标离散化后，设 $f(i, j, k)$ 表示区间 $[i, j]$ 纵坐标不小于 k 的部分的最优解。
- 对于 $f(i, j, k)$，要么枚举分割线，分成两部分分别 DP，要么放入一个尽量大的矩形，转化为子区间的问题。

Photo

- 如果两个矩形相交且不是包含关系，那么完全可以让它们不相交。
- 将坐标离散化后，设 $f(i,j,k)$ 表示区间 $[i,j]$ 纵坐标不小于 k 的部分的最优解。
- 对于 $f(i,j,k)$，要么枚举分割线，分成两部分分别 DP，要么放入一个尽量大的矩形，转化为子区间的问题。
- 时间复杂度 $O(n^4)$。

isn

给定一个长度为 n 的序列 $a_1, a_2, ..., a_n$。如果序列 a 不是单调不下降的，你必须从中删去一个数，直到 a 不下降为止。

给定一个长度为 n 的序列 $a_1, a_2, \ldots, a_n$。如果序列 a 不是单调不下降的，你必须从中删去一个数，直到 a 不下降为止。

求有多少种不同的操作方案。

isn

给定一个长度为 n 的序列 $a_1, a_2, ..., a_n$。如果序列 a 不是单调不下降的，你必须从中删去一个数，直到 a 不下降为止。

求有多少种不同的操作方案。

- $1 \leq n \leq 2000$。

isn

给定一个长度为 n 的序列 $a_1, a_2, ..., a_n$。如果序列 a 不是单调不下降的，你必须从中删去一个数，直到 a 不下降为止。

求有多少种不同的操作方案。

- $1 \le n \le 2000$。
- Source：BZOJ 4361

isn

- 有个直观的想法就是，枚举最终序列的长度 k，设长度为 k 的不下降子序列个数为 f_k，那么有 $(n-k)!f_k$ 种方案。

isn

- 有个直观的想法就是，枚举最终序列的长度 k，设长度为 k 的不下降子序列个数为 f_k，那么有 $(n-k)!f_k$ 种方案。
- 但是这样的正确性并不对，可能会多算。

- 有个直观的想法就是，枚举最终序列的长度 k，设长度为 k 的不下降子序列个数为 f_k，那么有 $(n-k)!f_k$ 种方案。
- 但是这样的正确性并不对，可能会多算。
- 注意到多算只可能是从一个长度为 $k+1$ 的不下降子序列中删去一个数，方案数为 $(n-k-1)!(k+1)f_{k+1}$。

- 有个直观的想法就是，枚举最终序列的长度 k，设长度为 k 的不下降子序列个数为 f_k，那么有 $(n-k)!f_k$ 种方案。
- 但是这样的正确性并不对，可能会多算。
- 注意到多算只可能是从一个长度为 $k+1$ 的不下降子序列中删去一个数，方案数为 $(n-k-1)!(k+1)f_{k+1}$。
- 所以用树状数组 DP 求出 f 后，$ans=\sum((n-k)!f_k-(n-k-1)!(k+1)f_{k+1})$。

- 有个直观的想法就是，枚举最终序列的长度 k，设长度为 k 的不下降子序列个数为 f_k，那么有 $(n-k)!f_k$ 种方案。
- 但是这样的正确性并不对，可能会多算。
- 注意到多算只可能是从一个长度为 $k+1$ 的不下降子序列中删去一个数，方案数为 $(n-k-1)!(k+1)f_{k+1}$。
- 所以用树状数组 DP 求出 f 后，$ans = \sum((n-k)!f_k - (n-k-1)!(k+1)f_{k+1})$。
- 时间复杂度 $O(n^2 \log n)$。

Data Structure You've Never Heard Of

给定一个长度为 n 的序列 $a_1, a_2, \ldots, a_n$，每个元素都是一个 d 维 01 向量，求所有不下降子序列的个数。

Data Structure You've Never Heard Of

给定一个长度为 n 的序列 $a_1, a_2, \ldots, a_n$，每个元素都是一个 d 维 01 向量，求所有不下降子序列的个数。

对于 d 维向量，$a_i \le a_j$ 等价于 a_i 的每一维都不大于 a_j。

Data Structure You've Never Heard Of

给定一个长度为 n 的序列 $a_1, a_2, \ldots, a_n$，每个元素都是一个 d 维 01 向量，求所有不下降子序列的个数。

对于 d 维向量，$a_i \le a_j$ 等价于 a_i 的每一维都不大于 a_j。

- $1 \le n \le 200000, 1 \le d \le 16$。

给定一个长度为 n 的序列 $a_1, a_2, \dots, a_n$，每个元素都是一个 d 维 01 向量，求所有不下降子序列的个数。

对于 d 维向量，$a_i \le a_j$ 等价于 a_i 的每一维都不大于 a_j。

- $1 \le n \le 200000, 1 \le d \le 16$。
- Source：ftiasch's Contest #4

- DP 方程显然，设 f_i 表示以 a_i 为结尾的不下降子序列的个数，则

Data Structure You've Never Heard Of

- DP 方程显然，设 f_i 表示以 a_i 为结尾的不下降子序列的个数，则
- $$\begin{aligned} f_i &= 1 + \sum_{1 \le j < i, a_j \le a_i} f_j \\ ans &= \sum_{i=1}^{n} f_i \end{aligned}$$

Data Structure You've Never Heard Of

- DP 方程显然，设 f_i 表示以 a_i 为结尾的不下降子序列的个数，则
- $$\begin{aligned} f_i &= 1 + \sum_{1 \le j < i, a_j \le a_i} f_j \\ ans &= \sum_{i=1}^{n} f_i \end{aligned}$$
- 直接 DP 是 $O(n^2)$ 的，不能承受。

Data Structure You've Never Heard Of

- 一个 d 维向量可以用一个 d 位二进制数来表示，$a \leq b$ 等价于 a 是 b 的子集。

Data Structure You've Never Heard Of

- 一个 d 维向量可以用一个 d 位二进制数来表示，$a \leq b$ 等价于 a 是 b 的子集。
- 考虑两种暴力，一种是记录每种数字的贡献，修改 $O(1)$，查询 $O(2^d)$。

- 一个 d 维向量可以用一个 d 位二进制数来表示，$a \le b$ 等价于 a 是 b 的子集。
- 考虑两种暴力，一种是记录每种数字的贡献，修改 $O(1)$，查询 $O(2^d)$。
- 另一种则是维护一个高维前缀和，修改 $O(2^d)$，查询 $O(1)$。

Data Structure You've Never Heard Of

- 一个 d 维向量可以用一个 d 位二进制数来表示，$a \le b$ 等价于 a 是 b 的子集。
- 考虑两种暴力，一种是记录每种数字的贡献，修改 $O(1)$，查询 $O(2^d)$。
- 另一种则是维护一个高维前缀和，修改 $O(2^d)$，查询 $O(1)$。
- 两种暴力在修改和查询上各有所长，所以把它们结合在一起即可得到复杂度优秀的算法。

Data Structure You've Never Heard Of

- 考虑 $d = 16$ 的情况，将 16 维划分为前 8 维和 8 维，对于前 8 维的每一个数维护后 8 维的前缀和。

- 考虑 $d = 16$ 的情况，将 16 维划分为前 8 维和 8 维，对于前 8 维的每一个数维护后 8 维的前缀和。
- 修改的时候，只要暴力修改某一个前 8 维里的前缀和，时间复杂度 $O(2^8)$。

Data Structure You've Never Heard Of

- 考虑 $d = 16$ 的情况，将 16 维划分为前 8 维和 8 维，对于前 8 维的每一个数维护后 8 维的前缀和。
- 修改的时候，只要暴力修改某一个前 8 维里的前缀和，时间复杂度 $O(2^8)$。
- 查询的时候，暴力枚举前 8 维，后 8 维可以 $O(1)$ 得到，时间复杂度 $O(2^8)$。

- 考虑 $d = 16$ 的情况，将 16 维划分为前 8 维和 8 维，对于前 8 维的每一个数维护后 8 维的前缀和。
- 修改的时候，只要暴力修改某一个前 8 维里的前缀和，时间复杂度 $O(2^8)$。
- 查询的时候，暴力枚举前 8 维，后 8 维可以 $O(1)$ 得到，时间复杂度 $O(2^8)$。
- 总时间复杂度 $O(n2^{\frac{d}{2}})$。

Tourism

给定一个 n 个点，m 条边的无向图，在第 i 个点建立旅游站点的费用为 C_i。在这张图中，任意两点间不存在节点数超过 10 的简单路径。

Tourism

给定一个 n 个点，m 条边的无向图，在第 i 个点建立旅游站点的费用为 C_i。在这张图中，任意两点间不存在节点数超过 10 的简单路径。

请找到一种费用最小的建立旅游站点的方案，使得每个点要么建立了旅游站点，要么与它有边直接相连的点里至少有一个点建立了旅游站点。

Tourism

给定一个 n 个点，m 条边的无向图，在第 i 个点建立旅游站点的费用为 C_i。在这张图中，任意两点间不存在节点数超过 10 的简单路径。

请找到一种费用最小的建立旅游站点的方案，使得每个点要么建立了旅游站点，要么与它有边直接相连的点里至少有一个点建立了旅游站点。

- $2 \leq n \leq 20000, 0 \leq m \leq 25000$。

Tourism

给定一个 n 个点，m 条边的无向图，在第 i 个点建立旅游站点的费用为 C_i。在这张图中，任意两点间不存在节点数超过 10 的简单路径。

请找到一种费用最小的建立旅游站点的方案，使得每个点要么建立了旅游站点，要么与它有边直接相连的点里至少有一个点建立了旅游站点。

- $2 \leq n \leq 20000, 0 \leq m \leq 25000$。
- Source：POI 2014

Tourism

- 对于一个连通块，取一个点进行 dfs，得到一棵 dfs 搜索树，则这棵树的深度不超过 10，且所有非树边都是前向边。

Tourism

- 对于一个连通块，取一个点进行 dfs，得到一棵 dfs 搜索树，则这棵树的深度不超过 10，且所有非树边都是前向边。
- 对于每个点 x，设 S 为三进制状态，S 第 i 位表示根到 x 路径上深度为 i 的点的状态：

  0：选了

  1：没选，且没满足

  2：没选，且已满足

Tourism

- 对于一个连通块，取一个点进行 dfs，得到一棵 dfs 搜索树，则这棵树的深度不超过 10，且所有非树边都是前向边。
- 对于每个点 x，设 S 为三进制状态，S 第 i 位表示根到 x 路径上深度为 i 的点的状态：
  0：选了
  1：没选，且没满足
  2：没选，且已满足
- 设 $f(i,j)$ 为考虑根到 x 路径上深度为 i 的点时这些点的状态为 j 时的最小费用，然后按 DFS 序进行 DP 即可。

Tourism

- 对于一个连通块，取一个点进行 dfs，得到一棵 dfs 搜索树，则这棵树的深度不超过 10，且所有非树边都是前向边。
- 对于每个点 x，设 S 为三进制状态，S 第 i 位表示根到 x 路径上深度为 i 的点的状态：
  0：选了
  1：没选，且没满足
  2：没选，且已满足
- 设 $f(i,j)$ 为考虑根到 x 路径上深度为 i 的点时这些点的状态为 j 时的最小费用，然后按 DFS 序进行 DP 即可。
- 时间复杂度 $O((n+m)3^{10})$。

Druzyny

体育课上，n 个小朋友排成一行 (从 1 到 n 编号)，老师想把他们分成若干组，每一组都包含编号连续的一段小朋友，每个小朋友属于且仅属于一个组。

Druzyny

体育课上，n 个小朋友排成一行 (从 1 到 n 编号)，老师想把他们分成若干组，每一组都包含编号连续的一段小朋友，每个小朋友属于且仅属于一个组。

第 i 个小朋友希望它所在的组的人数不多于 d_i，不少于 c_i，否则他就会不满意。

Druzyny

体育课上，n 个小朋友排成一行 (从 1 到 n 编号)，老师想把他们分成若干组，每一组都包含编号连续的一段小朋友，每个小朋友属于且仅属于一个组。

第 i 个小朋友希望它所在的组的人数不多于 d_i，不少于 c_i，否则他就会不满意。

在所有小朋友都满意的前提下，求可以分成的组的数目的最大值，以及有多少种分组方案能达到最大值。

Druzyny

体育课上，n 个小朋友排成一行 (从 1 到 n 编号)，老师想把他们分成若干组，每一组都包含编号连续的一段小朋友，每个小朋友属于且仅属于一个组。

第 i 个小朋友希望它所在的组的人数不多于 d_i，不少于 c_i，否则他就会不满意。

在所有小朋友都满意的前提下，求可以分成的组的数目的最大值，以及有多少种分组方案能达到最大值。

- $1 \le n \le 1000000$。

Druzyny

体育课上，n 个小朋友排成一行 (从 1 到 n 编号)，老师想把他们分成若干组，每一组都包含编号连续的一段小朋友，每个小朋友属于且仅属于一个组。

第 i 个小朋友希望它所在的组的人数不多于 d_i，不少于 c_i，否则他就会不满意。

在所有小朋友都满意的前提下，求可以分成的组的数目的最大值，以及有多少种分组方案能达到最大值。

- $1 \leq n \leq 1000000$。
- Source：PA 2014

Druzyny

- 设 f_i 为将前 i 个小朋友分组的最优解，则 $f_i = \max(f_j + 1)$，其中 $1 \le j < i$，且满足 $\max(c_{j+1}, c_{j+2}, ..., c_i) \le i - j \le \min(d_{j+1}, d_{j+2}, ..., d_i)$

Druzyny

- 设 f_i 为将前 i 个小朋友分组的最优解，则 $f_i = \max(f_j + 1)$，其中 $1 \le j < i$，且满足 $\max(c_{j+1}, c_{j+2}, ..., c_i) \le i - j \le \min(d_{j+1}, d_{j+2}, ..., d_i)$
- 设 $g_i = \min(j)$，且 $i - j \le \min(d_{j+1}, d_{j+2}, ..., d_i)$，容易发现 g_i 单调不下降，可以通过双指针以及维护线段树在 $O(n \log n)$ 的时间内预处理出来。

Druzyny

- 设 f_i 为将前 i 个小朋友分组的最优解，则 $f_i = \max(f_j + 1)$，其中 $1 \le j < i$，且满足 $\max(c_{j+1}, c_{j+2}, ..., c_i) \le i - j \le \min(d_{j+1}, d_{j+2}, ..., d_i)$
- 设 $g_i = \min(j)$，且 $i - j \le \min(d_{j+1}, d_{j+2}, ..., d_i)$，容易发现 g_i 单调不下降，可以通过双指针以及维护线段树在 $O(n \log n)$ 的时间内预处理出来。
- 那么对于 $[g_i, i-1]$ 中的 j，只需要满足 c 的限制即可。

Druzyny

- 考虑通过分治优化 DP。

Druzyny

- 考虑通过分治优化 DP。
- 在 solve(l,r) 时，求出 $[l+1,r]$ 中 c 最大的位置，设为 k，以 k 为分界线可以递归处理 solve(l,k-1) 和 solve(k,r)。现在只需用 $[l,k-1]$ 的决策更新 $[k,r]$。

Druzyny

- 考虑通过分治优化 DP。
- 在 solve(l,r) 时，求出 $[l+1,r]$ 中 c 最大的位置，设为 k，以 k 为分界线可以递归处理 solve(l,k-1) 和 solve(k,r)。现在只需用 $[l,k-1]$ 的决策更新 $[k,r]$。
- 由于 c_k 最大，所以 $c_k \le i-j$，i 从 $\max(c_k+l,k)$ 开始，决策 j 一开始的取值范围为 $[\max(l,g_i), i-c_k]$，此时用线段树求出区间最大值即可。

Druzyny

- 考虑通过分治优化 DP。
- 在 solve(l,r) 时，求出 $[l+1,r]$ 中 c 最大的位置，设为 k，以 k 为分界线可以递归处理 solve(l,k-1) 和 solve(k,r)。现在只需用 $[l,k-1]$ 的决策更新 $[k,r]$。
- 由于 c_k 最大，所以 $c_k \le i-j$，i 从 $\max(c_k+l,k)$ 开始，决策 j 一开始的取值范围为 $[\max(l,g_i),i-c_k]$，此时用线段树求出区间最大值即可。
- 每当 i 往右移一位时，j 的上限也往右移一位，可以做到 $O(1)$ 更新。

Druzyny

- j 的下限可能会右移到 g_i，此时有 $l \le g_i \le k-1$，由于所有更新 i 的区间 $[l, k-1]$ 均不相交，所以只存在一个区间 $[l, k-1]$ 满足 $l \le g_i \le k-1$，即每个 i 最多只会发生一次下限右移。对于每次右移用线段树查询新区间内的最优解即可。

Druzyny

- j 的下限可能会右移到 g_i，此时有 $l \leq g_i \leq k-1$，由于所有更新 i 的区间 $[l, k-1]$ 均不相交，所以只存在一个区间 $[l, k-1]$ 满足 $l \leq g_i \leq k-1$，即每个 i 最多只会发生一次下限右移。对于每次右移用线段树查询新区间内的最优解即可。
- 当 i 循环到 $k+c_k$ 时，$[k+c_k, r]$ 内所有 i 的可行决策 j 的上限都为 $k-1$，所以按 g 值的不同将这些 i 分割成若干段，对于每一段用线段树进行区间更新即可。

Druzyny

- j 的下限可能会右移到 g_i，此时有 $l \le g_i \le k-1$，由于所有更新 i 的区间 $[l, k-1]$ 均不相交，所以只存在一个区间 $[l, k-1]$ 满足 $l \le g_i \le k-1$，即每个 i 最多只会发生一次下限右移。对于每次右移用线段树查询新区间内的最优解即可。
- 当 i 循环到 $k+c_k$ 时，$[k+c_k, r]$ 内所有 i 的可行决策 j 的上限都为 $k-1$，所以按 g 值的不同将这些 i 分割成若干段，对于每一段用线段树进行区间更新即可。
- 时间复杂度 $O(n \log n)$。

Thank you for your attention

Thank you!