杂题选讲

陈嘉乐

杭州第二中学

2018 年 3 月 13 日

Description

Description

- 给你两个字符串$S(1 \le |S| \le 10^5), T(1 \le |T| \le 10^5)$，每个字符串都只包含字符'A','B','C'。

Description

- 给你两个字符串$S(1 \le |S| \le 10^5), T(1 \le |T| \le 10^5)$，每个字符串都只包含字符'A','B','C'。
- 有四种变换：

  A→BC；B→AC；C→AB；AAA→empty string。

Description

- 给你两个字符串$S(1 \le |S| \le 10^5), T(1 \le |T| \le 10^5)$，每个字符串都只包含字符'A','B','C'。
- 有四种变换：
  A→BC；B→AC；C→AB；AAA→empty string。
- $Q(1 \le Q \le 10^5)$次询问,每次问$S[a_i : b_i]$能否变换成为$T[c_i : d_i]$。

Solution

Solution

- 性质1

Solution

- 性质1
- 这四种变换都不能在末尾添加'A'。

Solution

- 性质1
- 这四种变换都不能在末尾添加'A'。
- 这四种变换都不能减少'B'和'C'的总量。

Solution

Solution

- 性质2

Solution

- 性质2
- $B \to AC \to AAB \to AAAC \to C$

Solution

- 性质2
- $B \to AC \to AAB \to AAAC \to C$
- $C \to AB \to AAC \to AAAB \to B$

Solution

- 性质2
- $B \to AC \to AAB \to AAAC \to C$
- $C \to AB \to AAC \to AAAB \to B$
- 即'B','C'可以互换，因此不妨全部看成'B'。

Solution

Solution

- 性质3

Solution

- 性质3
- $AB \to AAC \to AAAB \to B$

Solution

- 性质3
- $AB \to AAC \to AAAB \to B$
- $B \to AC \to AB$

Solution

- 性质3
- $AB \rightarrow AAC \rightarrow AAAB \rightarrow B$
- $B \rightarrow AC \rightarrow AB$
- 即'B'之前的'A'可删除可添加，不妨将询问的两字符串中的'B'或'C'之前的'A'全部删去。

Solution

Solution

- 性质4

Solution

- 性质4
- $B \to AC \to BCC \to BBB$

Solution

- 性质4
- $B \to AC \to BCC \to BBB$
- $A \to BC \to BB$

Solution

- 性质4
- $B \to AC \to BCC \to BBB$
- $A \to BC \to BB$
- 即'B'的个数只能两个两个添加，其奇偶性不会变。

Solution

Solution

- 我们根据以上四条性质判断串S_1能否变换成S_2。

Solution

- 我们根据以上四条性质判断串S_1能否变换成S_2。
- 根据性质1，S_1末尾'A'的数量要大于等于S_2，S_1中'B'的数量要小于等于S_2。

Solution

- 我们根据以上四条性质判断串S_1能否变换成S_2。
- 根据性质1，S_1末尾'A'的数量要大于等于S_2，S_1中'B'的数量要小于等于S_2。
- 根据性质4，S_1的'B'的数量的奇偶性需要与S_2相同

Solution

- 我们根据以上四条性质判断串S_1能否变换成S_2。
- 根据性质1，S_1末尾'A'的数量要大于等于S_2，S_1中'B'的数量要小于等于S_2。
- 根据性质4，S_1的'B'的数量的奇偶性需要与S_2相同
- 分类讨论，如果两者'B'的数量相同，唯一能进行的操作是删去末尾连续的三个'A'，因此只需要判断两者末尾'A'数量模3是否相等。

Solution

- 我们根据以上四条性质判断串S_1能否变换成S_2。
- 根据性质1，S_1末尾'A'的数量要大于等于S_2，S_1中'B'的数量要小于等于S_2。
- 根据性质4，S_1的'B'的数量的奇偶性需要与S_2相同
- 分类讨论，如果两者'B'的数量相同，唯一能进行的操作是删去末尾连续的三个'A'，因此只需要判断两者末尾'A'数量模3是否相等。
- 如果'B'数量不同且S_1末尾的'A'数量严格大于S_2，一定可行。

Solution

- 我们根据以上四条性质判断串S_1能否变换成S_2。
- 根据性质1，S_1末尾'A'的数量要大于等于S_2，S_1中'B'的数量要小于等于S_2。
- 根据性质4，S_1的'B'的数量的奇偶性需要与S_2相同
- 分类讨论，如果两者'B'的数量相同，唯一能进行的操作是删去末尾连续的三个'A'，因此只需要判断两者末尾'A'数量模3是否相等。
- 如果'B'数量不同且S_1末尾的'A'数量严格大于S_2，一定可行。
- 否则，如果串S_1的开头有'B'，那么可行，否则不可行。

Description

Description

- 给你一个$n \times n(1 \le n \le 10^5)$ 的网格，有$m(1 \le m \le 10^5)$个格子被染黑了，其余的都是白色的。

Description

- 给你一个$n \times n(1 \leq n \leq 10^5)$ 的网格，有$m(1 \leq m \leq 10^5)$个格子被染黑了，其余的都是白色的。
- 小H喜欢黑色，因此如果$(x, y), (y, z)$都是黑色的，小H会将(z, x)染黑。

Description

- 给你一个$n \times n(1 \le n \le 10^5)$ 的网格，有$m(1 \le m \le 10^5)$个格子被染黑了，其余的都是白色的。
- 小H喜欢黑色，因此如果$(x, y), (y, z)$都是黑色的，小H会将(z, x)染黑。
- 不断重复上述过程，求最终网格中黑色格子的数量。

Solution

Solution

- 将黑格(a, b)视为a到b的一条有向边，显然每个弱连通块是独立的，最终答案将每个弱连通块求和即可。

Solution

- 将黑格(a, b)视为a到b的一条有向边，显然每个弱连通块是独立的，最终答案将每个弱连通块求和即可。
- 下面只考虑一个弱连通块的情况。

Solution

Solution

- 尝试给每个点一个标号$x_i(0 \le x_i \le 2)$，使得对于一条有向边(a, b)，有$x_a + 1 \equiv x_b \pmod 3$。

Solution

- 尝试给每个点一个标号$x_i (0 \le x_i \le 2)$，使得对于一条有向边(a, b)，有$x_a + 1 \equiv x_b \pmod 3$。
- 如果存在这样的标号方式并且0,1,2三种标号都存在，那么我们必然可以找到一条长度为3的链（否则至多只有两种标号）。

Solution

- 尝试给每个点一个标号$x_i (0 \le x_i \le 2)$，使得对于一条有向边(a, b)，有$x_a + 1 \equiv x_b \pmod 3$。
- 如果存在这样的标号方式并且0,1,2三种标号都存在，那么我们必然可以找到一条长度为3的链（否则至多只有两种标号）。
- 设点集S一开始只包含链上的3个点，此时S中标号不同的点之间恰好有一条有向边。

Solution

- 尝试给每个点一个标号$x_i (0 \le x_i \le 2)$，使得对于一条有向边(a, b)，有$x_a + 1 \equiv x_b \pmod 3$。
- 如果存在这样的标号方式并且0,1,2三种标号都存在，那么我们必然可以找到一条长度为3的链（否则至多只有两种标号）。
- 设点集S一开始只包含链上的3个点，此时S中标号不同的点之间恰好有一条有向边。
- 我们任选一个和点集中的点相邻的点i，那么i和点集中与i标号不同的点之间也恰好能连一条有向边，后我们把i加入点集S。最终标号不同的点之间恰好有一条有向边。

Solution

Solution

- 如果存在这样的标号方式并且0,1,2三种标号至少有一种不存在，那么我们无法继续连边。

Solution

Solution

- 如果不存在这样的标号方式，我们不妨先取出一个存在长度为3的链的生成树进行合法的标号以及加边。

Solution

- 如果不存在这样的标号方式，我们不妨先取出一个存在长度为3的链的生成树进行合法的标号以及加边。
- 余下一定存在边(a, b)其中
$$x_a \equiv x_b \pmod 3$$
或
$$x_a \equiv x_b + 1 \pmod 3$$
。

Solution

- 如果不存在这样的标号方式，我们不妨先取出一个存在长度为3的链的生成树进行合法的标号以及加边。
- 余下一定存在边(a, b)其中
  $$x_a \equiv x_b \pmod 3$$
  或
  $$x_a \equiv x_b + 1 \pmod 3$$
  。
- 若$x_a \equiv x_b \pmod 3$，存在点$c, x_c \equiv x_a + 1 \pmod 3$，那么由于有边$(a, b), (b, c)$，会生成边$(c, a)$。而边$(a, c)$存在，因此出现自环。

Solution

- 如果不存在这样的标号方式，我们不妨先取出一个存在长度为3的链的生成树进行合法的标号以及加边。
- 余下一定存在边(a, b)其中

  $$x_a \equiv x_b \pmod 3$$

  或

  $$x_a \equiv x_b + 1 \pmod 3$$

  。
- 若$x_a \equiv x_b \pmod 3$，存在点$c, x_c \equiv x_a + 1 \pmod 3$，那么由于有边$(a, b), (b, c)$，会生成边$(c, a)$。而边$(a, c)$存在，因此出现自环。
- 若$x_a \equiv x_b + 1 \pmod 3$，那么原来就有边$(b, a)$，出现自环。

Solution

- 如果不存在这样的标号方式，我们不妨先取出一个存在长度为3的链的生成树进行合法的标号以及加边。
- 余下一定存在边(a, b)其中

$$x_a \equiv x_b \pmod 3$$

或

$$x_a \equiv x_b + 1 \pmod 3$$

。

- 若$x_a \equiv x_b \pmod 3$，存在点$c, x_c \equiv x_a + 1 \pmod 3$，那么由于有边$(a, b), (b, c)$，会生成边$(c, a)$。而边$(a, c)$存在，因此出现自环。
- 若$x_a \equiv x_b + 1 \pmod 3$，那么原来就有边$(b, a)$，出现自环。
- 也就是说，只要不存在合法的标号方式，一定会出现自环。

Solution

Solution

- 设点集S一开始只包含自环所在点，此时S中的点两两都有连边。

Solution

- 设点集S一开始只包含自环所在点，此时S中的点两两都有连边。
- 我们任选一个和点集中的点相邻的点i，那么i和点集中所有点两两都可以连上边，然后我们把i加入点集S。

Solution

- 设点集S一开始只包含自环所在点，此时S中的点两两都有连边。
- 我们任选一个和点集中的点相邻的点i，那么i和点集中所有点两两都可以连上边，然后我们把i加入点集S。
- 最终边数就是点数^2。

Description

Description

- 给你一棵以1为根的大小为$n(1 \leq n \leq 10^6)$ 的树，保证根节点度数大于等于2，点有黑白两种颜色。

Description

- 给你一棵以1为根的大小为$n(1 \leq n \leq 10^6)$ 的树，保证根节点度数大于等于2，点有黑白两种颜色。
- 小H从根出发，每次会随机走向相邻的点，走到叶子节点就会停下来。

Description

- 给你一棵以1为根的大小为$n(1 \le n \le 10^6)$ 的树，保证根节点度数大于等于2，点有黑白两种颜色。
- 小H从根出发，每次会随机走向相邻的点，走到叶子节点就会停下来。
- 有一个计数器cnt,初值为0。如果一个点是白点，小H第一次经过它时cnt++；否则每次经过时cnt++。

Description

- 给你一棵以1为根的大小为$n(1 \le n \le 10^6)$ 的树，保证根节点度数大于等于2，点有黑白两种颜色。
- 小H从根出发，每次会随机走向相邻的点，走到叶子节点就会停下来。
- 有一个计数器cnt,初值为0。如果一个点是白点，小H第一次经过它时cnt++；否则每次经过时cnt++。
- 问最终cnt的期望。

Solution

Solution

- 黑白点分开考虑。

Solution

Solution

- 对于白点，考虑独立计算每个点被经过的概率。

Solution

- 对于白点，考虑独立计算每个点被经过的概率。
- 设deg_i为点i的度数，$f_{i,j}$表示在点i时能走到一个相邻的点j的概率，p_i为i的父亲节点，dp_i表示到达点i 的概率。

Solution

- 对于白点，考虑独立计算每个点被经过的概率。
- 设deg_i为点i的度数，$f_{i,j}$表示在点i时能走到一个相邻的点j的概率，p_i为i的父亲节点，dp_i表示到达点i 的概率。
- 一条从根出发的路径如果经过点i显然要先经过p_i，因此$dp_i = dp_{p_i} \times f_{p_i,i}$，下面考虑如何计算$f_{i,j}$。

Solution

Solution

- 若i为叶子，$f_{i,j}$为0。

Solution

- 若i为叶子，$f_{i,j}$为0。
- 否则，令$s_{i,j} = \frac{1}{deg_i} \sum_{k \neq j,(i,k) \in E} f_{k,i}$，也就是从点$i$出发，没有走到$j$，再回到$i$的概率。

Solution

- 若i为叶子，$f_{i,j}$为0。
- 否则，令$s_{i,j} = \frac{1}{deg_i} \sum_{k \neq j,(i,k) \in E} f_{k,i}$，也就是从点$i$出发，没有走到$j$，再回到$i$的概率。
- 那么$f_{i,j} = \frac{1}{deg_i(1-s_{i,j})}$。

Solution

- 若i为叶子，$f_{i,j}$为0。
- 否则，令$s_{i,j} = \frac{1}{deg_i} \sum_{k \neq j,(i,k) \in E} f_{k,i}$，也就是从点$i$出发，没有走到$j$，再回到$i$的概率。
- 那么$f_{i,j} = \frac{1}{deg_i(1-s_{i,j})}$。
- 先自下而上dp出所有的f_{i,p_i}，再自上而下计算$f_{p_i,i}$即可。

Solution

Solution

- 对于黑点，设g_i表示从i号点出发期望经过的黑点次数。

Solution

- 对于黑点，设g_i表示从i号点出发期望经过的黑点次数。
- 如果i是叶子，那么$g_i = [i\text{为黑点}]$

Solution

- 对于黑点，设g_i表示从i号点出发期望经过的黑点次数。
- 如果i是叶子，那么$g_i = [i\text{为黑点}]$
- 否则，$g_i = \frac{1}{deg_i} \sum_{j,(i,j) \in E} g_j + [i\text{为黑点}]$

Solution

- 对于黑点，设g_i表示从i号点出发期望经过的黑点次数。
- 如果i是叶子，那么$g_i = [i为黑点]$
- 否则，$g_i = \frac{1}{deg_i} \sum_{j,(i,j)\in E} g_j + [i为黑点]$
- 由于树状结构，g_i可以表示成$k_i \times g_{p_i} + b_i$，自下而上推到点1时解方程求出g_1即为答案。

Description

Description

- 给你一个字符串$S(1 \leq |S| \leq 10^5)$，字符集大小为8。

Description

- 给你一个字符串$S(1 \le |S| \le 10^5)$，字符集大小为8。
- 在图G中，点i与点j有边当且仅当$S_i = S_j$或者$|i - j| = 1$，且边长均为1。

Description

- 给你一个字符串$S(1 \le |S| \le 10^5)$，字符集大小为8。
- 在图G中，点i与点j有边当且仅当$S_i = S_j$或者$|i - j| = 1$，且边长均为1。
- 求图G的直径（任意两点间的最短路的最大值）以及距离是直径的点对数。

Solution

Solution

- 设Σ表示字符集大小，那么直径至多为$2 \times \Sigma - 1$

Solution

- 设Σ表示字符集大小，那么直径至多为$2 \times \Sigma - 1$
- 令$f_{i,c}$表示从点i出发走到字符为c的位置的最短距离，可以bfs得到。

Solution

- 设Σ表示字符集大小，那么直径至多为$2 \times \Sigma - 1$
- 令$f_{i,c}$表示从点i出发走到字符为c的位置的最短距离，可以bfs得到。
- 令g_{c_1,c_2}表示从任意一个字符为c_1的位置走到字符为c_2的位置的最短距离，可以通过$f_{i,c}$得到，并且容易发现的是$g_{s_i,c} \leq f_{i,c} \leq g_{s_i,c} + 1$。

Solution

Solution

- 考虑点i与点j的距离，
  $dis(i,j) = min\left(|i-j|, min(f_{i,c} + 1 + f_{j,c})\right)$。

Solution

- 考虑点i与点j的距离，$dis(i,j) = min\big(|i-j|, min(f_{i,c}+1+f_{j,c})\big)$。
- 我们从小到大枚举i，计算比i小的所有j到i的最短距离的最大值。

Solution

- 考虑点i与点j的距离，
  $dis(i,j) = min\left(|i-j|, min(f_{i,c}+1+f_{j,c})\right)$。
- 我们从小到大枚举i，计算比i小的所有j到i的最短距离的最大值。
- 由于直径至多为$2 \times \Sigma - 1$，由$|i-j|$贡献的j只可能是末尾的几个，之前的j的贡献方式都是$min(f_{i,c}+1+f_{j,c})$。

Solution

- 考虑点i与点j的距离，
  $dis(i,j) = min\left(|i-j|, min(f_{i,c} + 1 + f_{j,c})\right)$。
- 我们从小到大枚举i，计算比i小的所有j到i的最短距离的最大值。
- 由于直径至多为$2 \times \Sigma - 1$，由$|i-j|$贡献的j只可能是末尾的几个，之前的j的贡献方式都是$min(f_{i,c} + 1 + f_{j,c})$。
- 而$g_{s_j,c} \le f_{j,c} \le g_{s_j,c} + 1$，那么每个点到各个字符的距离实际上可以表示成一个二进制数，每一位的01表示是否比字符到字符的距离大。

Solution

- 考虑点i与点j的距离，
  $dis(i,j) = min\left(|i-j|, min(f_{i,c}+1+f_{j,c})\right)$。
- 我们从小到大枚举i，计算比i小的所有j到i的最短距离的最大值。
- 由于直径至多为$2\times\Sigma-1$，由$|i-j|$贡献的j只可能是末尾的几个，之前的j的贡献方式都是$min(f_{i,c}+1+f_{j,c})$。
- 而$g_{s_j,c}\le f_{j,c}\le g_{s_j,c}+1$，那么每个点到各个字符的距离实际上可以表示成一个二进制数，每一位的01表示是否比字符到字符的距离大。
- 对那些字符相同二进制数相同的位置计算出的直径是相同的，我们枚举后统计答案即可。

Description

Description

- 给你一个$n(1 \leq n \leq 15)$个点$m(1 \leq m \leq n(n-1)/2)$条边的没有重边的有向图$G$，保证任意一条边$(x_i, y_i)$中$x_i < y_i$。

Description

- 给你一个$n(1 \le n \le 15)$个点$m(1 \le m \le n(n-1)/2)$条边的没有重边的有向图G，保证任意一条边(x_i, y_i)中$x_i < y_i$。
- 考虑m条边组成的集合的每一个子集，从图G中删去这些边，会形成新的图G'。

Description

- 给你一个$n(1 \le n \le 15)$个点$m(1 \le m \le n(n-1)/2)$条边的没有重边的有向图G，保证任意一条边(x_i, y_i)中$x_i < y_i$。
- 考虑m条边组成的集合的每一个子集，从图G中删去这些边，会形成新的图G'。
- 小H和小R会在G'上玩游戏，小H先手，两人轮流操作。

Description

- 给你一个$n(1 \le n \le 15)$个点$m(1 \le m \le n(n-1)/2)$条边的没有重边的有向图G，保证任意一条边(x_i, y_i)中$x_i < y_i$。
- 考虑m条边组成的集合的每一个子集，从图G中删去这些边，会形成新的图G'。
- 小H和小R会在G'上玩游戏，小H先手，两人轮流操作。
- 起初有两颗石子，一颗在点1，一颗在点2。

Description

- 给你一个$n(1 \le n \le 15)$个点$m(1 \le m \le n(n-1)/2)$条边的没有重边的有向图G，保证任意一条边(x_i, y_i)中$x_i < y_i$。
- 考虑m条边组成的集合的每一个子集，从图G中删去这些边，会形成新的图G'。
- 小H和小R会在G'上玩游戏，小H先手，两人轮流操作。
- 起初有两颗石子，一颗在点1，一颗在点2。
- 每次操作可以选择一条边i，要求x_i上有石子，然后将这颗石子移到y_i。一个点上可以有两颗石子。

Description

- 给你一个$n(1 \le n \le 15)$个点$m(1 \le m \le n(n-1)/2)$条边的没有重边的有向图G，保证任意一条边(x_i, y_i)中$x_i < y_i$。
- 考虑m条边组成的集合的每一个子集，从图G中删去这些边，会形成新的图G'。
- 小H和小R会在G'上玩游戏，小H先手，两人轮流操作。
- 起初有两颗石子，一颗在点1，一颗在点2。
- 每次操作可以选择一条边i，要求x_i上有石子，然后将这颗石子移到y_i。一个点上可以有两颗石子。
- 不能操作者输。

Description

- 给你一个$n(1 \le n \le 15)$个点$m(1 \le m \le n(n-1)/2)$条边的没有重边的有向图G，保证任意一条边(x_i, y_i)中$x_i < y_i$。
- 考虑m条边组成的集合的每一个子集，从图G中删去这些边，会形成新的图G'。
- 小H和小R会在G'上玩游戏，小H先手，两人轮流操作。
- 起初有两颗石子，一颗在点1，一颗在点2。
- 每次操作可以选择一条边i，要求x_i上有石子，然后将这颗石子移到y_i。一个点上可以有两颗石子。
- 不能操作者输。
- 问这2^m个图G'中有几个是小H必胜的。

Solution

Solution

- 一个很朴素的想法是，枚举每一张图，算出点1和点2的SG值，如果不等则小H必胜。

Solution

- 一个很朴素的想法是，枚举每一张图，算出点1和点2的SG值，如果不等则小H必胜。
- 但枚举的代价太高，考虑dp出多少张图中点1和点2的SG值不同。

Solution

Solution

- 不妨先观察一下一个DAG中的SG值分布。

Solution

- 不妨先观察一下一个DAG中的SG值分布。
- 假如我们把点按照SG值分层，我们发现，第0层的点没有出边，第$i(i > 0)$层的点内部不会有连边，会向小于i 的每一层中至少一个点连边。

Solution

- 不妨先观察一下一个DAG中的SG值分布。
- 假如我们把点按照SG值分层，我们发现，第0层的点没有出边，第$i(i > 0)$层的点内部不会有连边，会向小于i 的每一层中至少一个点连边。
- 容易发现，这同时也是一个合法分布的充分条件。

Solution

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。
- 具体与哪些边有关在以下说明。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。
- 具体与哪些边有关在以下说明。
- 首先，S'内部的边全部删去。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。
- 具体与哪些边有关在以下说明。
- 首先，S'内部的边全部删去。
- 对于那些边两端的点SG值不同的边，我们在SG值较小点被加入点集时考虑这条边是否删除。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。
- 具体与哪些边有关在以下说明。
- 首先，S'内部的边全部删去。
- 对于那些边两端的点SG值不同的边，我们在SG值较小点被加入点集时考虑这条边是否删除。
- 即，对那些由尚未确定的点连向S'中的边，要求每个点至少有一条保留。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。
- 具体与哪些边有关在以下说明。
- 首先，S'内部的边全部删去。
- 对于那些边两端的点SG值不同的边，我们在SG值较小点被加入点集时考虑这条边是否删除。
- 即，对那些由尚未确定的点连向S'中的边，要求每个点至少有一条保留。
- 而，由S'中连向尚未确定的点的边，可删除可保留。

Solution

- 因此，我们令f_S表示，考虑了点集为S的删边方案。
- 枚举新的一层点集S'转移即可。
- 具体与哪些边有关在以下说明。
- 首先，S'内部的边全部删去。
- 对于那些边两端的点SG值不同的边，我们在SG值较小点被加入点集时考虑这条边是否删除。
- 即，对那些由尚未确定的点连向S'中的边，要求每个点至少有一条保留。
- 而，由S'中连向尚未确定的点的边，可删除可保留。
- 注意$1, 2$号点不要在同一层加入点集。

Description

Description

- 给你一个长度为$n(1 \leq n \leq 5 \times 10^5)$的序列$\{a_i\}$，初始全为0。

Description

- 给你一个长度为$n(1 \leq n \leq 5 \times 10^5)$的序列$\{a_i\}$，初始全为0。
- 小H会给这个序列染$m(1 \leq m \leq 5 \times 10^5)$次色。

Description

- 给你一个长度为$n(1 \leq n \leq 5 \times 10^5)$的序列$\{a_i\}$，初始全为0。
- 小H会给这个序列染$m(1 \leq m \leq 5 \times 10^5)$次色。
- 在第i次染色操作中，小H可以选择任意一对$l, r(1 \leq l \leq r \leq n)$，并将$a_l, \ldots, a_r$的值赋为$i$。

Description

- 给你一个长度为$n(1 \le n \le 5 \times 10^5)$的序列$\{a_i\}$，初始全为0。

- 小H会给这个序列染$m(1 \le m \le 5 \times 10^5)$次色。
- 在第i次染色操作中，小H可以选择任意一对$l, r(1 \le l \le r \le n)$，并将$a_l, \ldots, a_r$的值赋为$i$。
- 小H想知道，最终可能的不同的序列有多少种。

Solution

Solution

- 容易发现一个简单的性质。

Solution

- 容易发现一个简单的性质。
- 一个序列是合法的当且仅当对于任意两个颜色为i的位置之间不存在颜色小于i的位置。

Solution

- 容易发现一个简单的性质。
- 一个序列是合法的当且仅当对于任意两个颜色为i的位置之间不存在颜色小于i的位置。
- 换句话说，如果我们将最终颜色序列中大于i的位置删去后，颜色为i的位置是连续的一段。

Solution

- 容易发现一个简单的性质。
- 一个序列是合法的当且仅当对于任意两个颜色为i的位置之间不存在颜色小于i的位置。
- 换句话说，如果我们将最终颜色序列中大于i的位置删去后，颜色为i的位置是连续的一段。
- 这启发我们不再以题目里的操作进行染色，而是每次在任意间隔中插入颜色为i的一段，求形成一个长度为n的序列的方案数。

Solution

Solution

- 考虑dp。

Solution

- 考虑dp。
- 令$f_{i,j}$表示插入了前i种颜色，总长度为j的方案数。

Solution

- 考虑dp。
- 令$f_{i,j}$表示插入了前i种颜色，总长度为j的方案数。
- 枚举第i种颜色的长度k，有$j-k+1$种转移。

Solution

- 考虑dp。
- 令$f_{i,j}$表示插入了前i种颜色，总长度为j的方案数。
- 枚举第i种颜色的长度k，有$j-k+1$种转移。
- 注意到当$k=0$时只有1种转移。

Solution

- 考虑dp。
- 令$f_{i,j}$表示插入了前i种颜色，总长度为j的方案数。
- 枚举第i种颜色的长度k，有$j-k+1$种转移。
- 注意到当$k=0$时只有1种转移。
- 特别的，第m种颜色的长度不能为0 。

Solution

- 考虑dp。
- 令$f_{i,j}$表示插入了前i种颜色，总长度为j的方案数。
- 枚举第i种颜色的长度k，有$j-k+1$种转移。
- 注意到当$k=0$时只有1种转移。
- 特别的，第m种颜色的长度不能为0 。
- 即$f_{i,j}=\sum_{k=1}^{j} f_{i-1,j-k}\times(j-k+1)+f_{i-1,j}$

Solution

- 考虑dp。
- 令$f_{i,j}$表示插入了前i种颜色，总长度为j的方案数。
- 枚举第i种颜色的长度k，有$j-k+1$种转移。
- 注意到当$k=0$时只有1种转移。
- 特别的，第m种颜色的长度不能为0 。
- 即$f_{i,j}=\sum_{k=1}^{j} f_{i-1,j-k}\times(j-k+1)+f_{i-1,j}$
- $f_{m,n}$即为答案。

Solution

Solution

- 直接不太好算，不妨先除掉那些$k = 0$的转移，枚举其数量最后组合数乘回去即可。

Solution

- 直接不太好算，不妨先除掉那些$k = 0$的转移，枚举其数量最后组合数乘回去即可。
- 考虑有效转移为m'次，每次转移前的长度为$a_1, \ldots, a_{m'}$且$0 \le a_1 < \ldots < a_{m'} < n$。

Solution

- 直接不太好算，不妨先除掉那些$k=0$的转移，枚举其数量最后组合数乘回去即可。
- 考虑有效转移为m'次，每次转移前的长度为$a_1,\dots,a_{m'}$且$0\le a_1<\dots<a_{m'}<n$。
- 那么最终答案就是$\prod_{i=1}^{m'}(a_i+1)$。

Solution

- 直接不太好算，不妨先除掉那些$k = 0$的转移，枚举其数量最后组合数乘回去即可。
- 考虑有效转移为m'次，每次转移前的长度为$a_1, \ldots, a_{m'}$且$0 \le a_1 < \ldots < a_{m'} < n$。
- 那么最终答案就是$\prod_{i=1}^{m'}(a_i + 1)$。
- 该如何求得对于1到n中的每一个m'的方案数呢。

Solution

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n - k$的方案数。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n - k$的方案数。
- 很容易想到用分治FFT计算$F(x)$,但直接算是$O(nlog^2 n)$的。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n - k$的方案数。
- 很容易想到用分治FFT计算$F(x)$,但直接算是$O(nlog^2 n)$的。
- 考虑分治的时候根据左半边的计算结果直接得出右半边的多项式，而非递归下去做。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n-k$的方案数。
- 很容易想到用分治FFT计算$F(x)$,但直接算是$O(nlog^2 n)$的。
- 考虑分治的时候根据左半边的计算结果直接得出右半边的多项式，而非递归下去做。
- 当左右两边长度相同时，假设左边的结果是$G(x)$，右边的结果就是$G(x+len)$。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n - k$的方案数。
- 很容易想到用分治FFT计算$F(x)$,但直接算是$O(nlog^2n)$的。
- 考虑分治的时候根据左半边的计算结果直接得出右半边的多项式，而非递归下去做。
- 当左右两边长度相同时，假设左边的结果是$G(x)$，右边的结果就是$G(x+len)$。
- 考虑二项式定理，$G(x+len)$可以快速求得。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n - k$的方案数。
- 很容易想到用分治FFT计算$F(x)$,但直接算是$O(nlog^2n)$的。
- 考虑分治的时候根据左半边的计算结果直接得出右半边的多项式，而非递归下去做。
- 当左右两边长度相同时，假设左边的结果是$G(x)$，右边的结果就是$G(x+len)$。
- 考虑二项式定理，$G(x+len)$可以快速求得。
- 而当长度不同时，先计算相同的那一部分，再乘上多出来的一项即可。

Solution

- 考虑这样一个多项式$F(x) = (x+1)(x+2)\cdots(x+n)$。
- 其最终结果中x^k项的系数就是$m' = n - k$的方案数。
- 很容易想到用分治FFT计算$F(x)$,但直接算是$O(nlog^2n)$的。
- 考虑分治的时候根据左半边的计算结果直接得出右半边的多项式，而非递归下去做。
- 当左右两边长度相同时，假设左边的结果是$G(x)$，右边的结果就是$G(x+len)$。
- 考虑二项式定理，$G(x+len)$可以快速求得。
- 而当长度不同时，先计算相同的那一部分，再乘上多出来的一项即可。
- 总复杂度$O(nlogn)$

Description

Description

- 小H喜欢森林，她定义一个森林的价值为森林中每棵树的点数的平方和。

Description

- 小H喜欢森林，她定义一个森林的价值为森林中每棵树的点数的平方和。
- 现在给你一棵$n(1 \le n \le 10^5)$个点的树，小H想知道，对于$i = 0 \ldots n-1$，随机删去i条边后形成的森林的价值的期望。

Solution

Solution

- 考虑森林的价值的等价形式。

Solution

- 考虑森林的价值的等价形式。
- 即为选出一个点v_1，再选出一个点v_2，这样连通的(v_1, v_2)的总对数。

Solution

- 考虑森林的价值的等价形式。
- 即为选出一个点v_1，再选出一个点v_2，这样连通的(v_1, v_2)的总对数。
- 再根据期望的线性性，森林的价值的期望即为对于每一对(v_1, v_2)连通的概率的和。

Solution

Solution

- 如果确定删去i条边，且$v1, v2$之间的边数为j，它对答案的贡献是$\binom{n-1-j}{i}/2^{n-1}$，分母是定值，暂时不考虑。

Solution

- 如果确定删去i条边，且$v1, v2$之间的边数为j，它对答案的贡献是$\binom{n-1-j}{i}/2^{n-1}$，分母是定值，暂时不考虑。
- 假设我们求出了边数为$j(0 \le j \le n-1)$的链的数量$\{cnt_j\}$（这个问题我们接下来解决）。

Solution

- 如果确定删去i条边，且$v1, v2$之间的边数为j，它对答案的贡献是$\binom{n-1-j}{i}/2^{n-1}$，分母是定值，暂时不考虑。
- 假设我们求出了边数为$j(0 \le j \le n-1)$的链的数量$\{cnt_j\}$（这个问题我们接下来解决）。
- 那么

$$ans_i = \sum_{j=0}^{n-1} cnt_j \binom{n-1-j}{i} = \sum_{j=0}^{n-1} cnt_j (n-1-j)! \frac{1}{(n-1-j-i)!} \frac{1}{i!}$$

Solution

- 如果确定删去i条边，且$v1, v2$之间的边数为j，它对答案的贡献是$\binom{n-1-j}{i}/2^{n-1}$，分母是定值，暂时不考虑。
- 假设我们求出了边数为$j(0 \le j \le n-1)$的链的数量$\{cnt_j\}$（这个问题我们接下来解决）。
- 那么
$$ans_i = \sum_{j=0}^{n-1} cnt_j \binom{n-1-j}{i} = \sum_{j=0}^{n-1} cnt_j (n-1-j)! \frac{1}{(n-1-j-i)!} \frac{1}{i!}$$
- $$i!ans_i = \sum_{j=0}^{n-1} \left(cnt_j (n-1-j)!\right) \frac{1}{(n-1-j-i)!}$$

Solution

- 如果确定删去i条边，且$v1, v2$之间的边数为j，它对答案的贡献是$\binom{n-1-j}{i}/2^{n-1}$，分母是定值，暂时不考虑。
- 假设我们求出了边数为$j(0 \le j \le n-1)$的链的数量$\{cnt_j\}$（这个问题我们接下来解决）。
- 那么

$$ans_i = \sum_{j=0}^{n-1} cnt_j \binom{n-1-j}{i} = \sum_{j=0}^{n-1} cnt_j (n-1-j)! \frac{1}{(n-1-j-i)!} \frac{1}{i!}$$

- $$i!ans_i = \sum_{j=0}^{n-1} (cnt_j(n-1-j)!) \frac{1}{(n-1-j-i)!}$$

- FFT计算。

Solution

Solution

- 考虑计算*cnt*数组。

Solution

- 考虑计算cnt数组。
- 点分治，在每个点分重心处，将孩子按照深度从小到大排序，FFT合并。

GL&HF

GL&HF

- 祝愿各位都能在省选中取得满意的成绩。

GL&HF

- 祝愿各位都能在省选中取得满意的成绩。
- 谢谢大家!