杂题选讲

浙江省镇海中学
范舒翼 岑若虚 邹逍遥

April 1, 2015

Peaks

- 对于一个$1 \sim n$的排列$\langle a_1, a_2, \cdots, a_n \rangle$，若$a_{i-1} < a_i < a_{i+1}$，则称$a_i$为一个峰（为了方便，令$a_0 = a_{n+1} = 0$）
- 求n的排列中峰的个数为k的数目对239取模的值
- $n \leq 10^{15}, k \leq 30$

sample

- input:
  3 1
- output:
  4

- 1 2 3
  1 3 2
  2 3 1
  3 2 1

solution

- 我们用$f[i][j]$表示$1 \sim i$的排列中峰数为j的数目
- $f[1][1] = 1$
- $f[i][j] = f[i-1][j] \times 2j + f[i-1][j-1] \times (i-2(j-1))(i > 1)$
- $f[n][k]$即为所求
- 由于n很大，我们可以用矩阵乘法来加速

solution

- 我们用$f[i][j]$表示$1 \sim i$的排列中峰数为j的数目
- $f[1][1] = 1$
- $f[i][j] = f[i-1][j] \times 2j + f[i-1][j-1] \times (i - 2(j-1))(i > 1)$
- $f[n][k]$即为所求
- 由于n很大，我们可以用矩阵乘法来加速
- 但是系数又和i有关

solution

- 注意到模数只有239，我们可以把矩阵模239，这样就会出现一个长度为239的循环节，用快速幂加速

solution

- 注意到模数只有239，我们可以把矩阵模239，这样就会出现一个长度为239的循环节，用快速幂加速
- 如果模数是几个较小数的乘积的话，我们把模数分解，再用中国剩余定理合并答案就可以了

Fibonacci System

- 把一个数X表示成斐波那契进制：$X = \overline{a_n a_{n-1} \cdots a_1}_F$
- 不能有连续两个1，$X = a_n F_n + a_{n-1} F_{n-1} + \cdots + a_1 F_1$
  $F_0 = F_1 = 1, F_i = F_{i-1} + F_{i-2}$
- 如果把1到无穷大转换成斐波那契进制，并连续排列，就会变成这样： 11010010110001001101O…
- 求这个无穷序列的前n位中有几个1
- $n \leq 10^{15}$

sample

- input:
  21
- output:
  10

- $1 = 1F$
  $2 = 10F$
  $3 = 100F$
  $4 = 101F$
  $5 = 1000F$
  $6 = 1001F$
  $7 = 1010F$

solution

- 首先我们需要知道前n位大概是由多少数转成斐波那契进制连起来的
- 即求最大的m，满足$1 \sim m$的斐波那契进制位数和不大于n
- 可以二分来求

solution

- 这样问题就变成了求$1\sim m$的数位和
- 我们先把m转成斐波那契进制来数位DP
- 考虑第i位（从右数起），如果是0，跳过；
  如果是1，对答案的贡献是$i-1$位斐波那契数的数位和 $+\ i-1$位斐波那契数的个数 $\times$ 第i位左边的1的数目

solution

- $f[i][j]$表示共j位首位是i的斐波那契数的数位和
- $g[i][j]$表示共j位首位是i的斐波那契数的个数
- 预处理
- $f[0][j] = f[1][j-1] + f[0][j-1]$
- $f[1][j] = f[0][j-1] + g[0][j-1]$
- $g[0][j] = g[0][j-1] + g[1][j-1]$
- $g[1][j] = g[0][j-1]$

Hospital

- 医院里有两种工作人员A和B，A类休假的时候不需要找人来代替他，B类休假的时候一定要找人来代班，如果来代班的人也属于B类，就要再找人代班
- 现有n个工作人员，已知每个B类员工可以由哪些人来代班
- 求有多少对(X,Y)满足X和Y单独休假时都没有问题，但他们一起休假时会出现问题
- $n \le 5000, m \le 20000$(m为替代关系的总数)

sample

- input:
  7 5
  2 6 7
  1 7
  2 2 7
  1 5
  1 4

- output:
  3
- (2,3)
  (2,7)
  (3,7)

solution

- 把每个A类员工看成一个白点，每个B类看成一个黑点，v能代替u看成一条u到v的有向边。
- 一个员工能否单独休假就是能否找一条从这个点开始以一个白点结束的路径
- 两个员工能否单独休假就是能否找两条分别从这两个点开始，都以白点结束的路径，且两条路径不相交

solution

- 答案有两种情况：
- （1）X,Y一个白点，一个黑点
- 以每一个黑点为起点bfs，如果一个黑点恰好只能到达一个白点，那这两个点就满足条件。

- （2）X,Y两个都是黑点
- 考虑判断一对(X,Y)是否可行，我们可以用网络流来检验
- 那如果我们枚举X，任意找一条以白点结束的路径，修改那条路径上的流量，我们会发现，所有边的流量不是0就是1，求Y的时候我们只要再做一遍bfs就可以了。

ZCC Loves Minecraft

- 一个点集S的正交凸包C是满足如下条件的点集中面积最小的一个。

1. S中的点都在C中；
2. C和任意一条水平或竖直直线的交要么为空，要么是连续的一段（可以是一个点）。

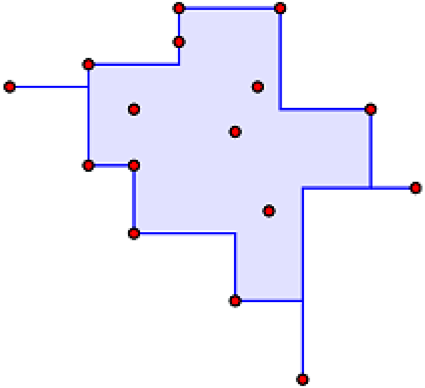
ZCC Loves Minecraft

- 一个点集S的正交凸包C是满足如下条件的点集中面积最小的一个。

1. S中的点都在C中；
2. C和任意一条水平或竖直直线的交要么为空，要么是连续的一段（可以是一个点）。

- 一开始S为空。在S中依次插入N个点，求每次插入后正交凸包的面积。数据保证C总是连续的。
- $N \leq 100000$

例子



无插入

- 正交凸包由四段折线构成。
- 求右上方的一段折线，可以从最上面的点开始，每次找当前点右边的点中最上面的点，与当前点L形连接，作为新的当前点，不断重复直到选到最右边的点。其它三段是类似的。

有插入

- 用四棵平衡树分别维护四段折线。
- 插入点时尝试将其分别插入四段折线的相应位置，若在当前折线外部需要更新折线，并计算面积的增量。

有插入

- 用四棵平衡树分别维护四段折线。
- 插入点时尝试将其分别插入四段折线的相应位置，若在当前折线外部需要更新折线，并计算面积的增量。
- 可以利用对称性和STL set简化代码。

有插入

- 用四棵平衡树分别维护四段折线。
- 插入点时尝试将其分别插入四段折线的相应位置，若在当前折线外部需要更新折线，并计算面积的增量。
- 可以利用对称性和STL set简化代码。
- 由于每个点最多被插入一次删除一次，总的时间复杂度是$O(n \log n)$.

Light

- $N \times M$的棋盘，已知每个格子的颜色（黑或白）。可以进行以下两种操作：

1. 选择一个格子，改变与它相邻的格子的颜色；
2. 选择一个格子，改变与它相邻的格子以及它本身的颜色。

Light

- $N \times M$的棋盘，已知每个格子的颜色（黑或白）。可以进行以下两种操作：

1. 选择一个格子，改变与它相邻的格子的颜色；
2. 选择一个格子，改变与它相邻的格子以及它本身的颜色。

Sample

$$\begin{array}{c} 001 \\ 10\textcolor{red}{0} \\ 000 \end{array} \rightarrow \begin{array}{c} 000 \\ 110 \\ 0\textcolor{red}{0}1 \end{array} \rightarrow \begin{array}{c} 000 \\ 100 \\ \textcolor{red}{1}10 \end{array} \rightarrow \begin{array}{c} 000 \\ 000 \\ 000 \end{array}$$

Light

- $N \times M$的棋盘，已知每个格子的颜色（黑或白）。可以进行以下两种操作：

1. 选择一个格子，改变与它相邻的格子的颜色；
2. 选择一个格子，改变与它相邻的格子以及它本身的颜色。

Sample

$$\begin{matrix}001\\10\color{red}{0}\\000\end{matrix} \rightarrow \begin{matrix}000\\110\\00\color{red}{1}\end{matrix} \rightarrow \begin{matrix}000\\100\\\color{red}{1}10\end{matrix} \rightarrow \begin{matrix}000\\000\\000\end{matrix}$$

- 求将所有格子变白的最小操作步数。
- $N, M \leq 10$

分析

- 轮廓线DP。
- 显然每种操作最多进行一次。

- 轮廓线DP。
- 显然每种操作最多进行一次。
- 一种比较朴素的方法是，对轮廓线上每个格子记录当前颜色、是否进行了操作1、是否进行了操作2。状态数为$O(nm8^n)$.
- 转移时枚举新格子的操作，计算附近格子的颜色，保证轮廓线内的格子为白色。当前格子的颜色可以从左边和上面格子的操作得出。

优化状态

- 只需要知道操作对其它格子的贡献。如果一个格子进行了操作1和操作2中的一个，它对相邻格子的贡献为1。如果没有操作或进行了两个操作，贡献为0。

优化状态

- 只需要知道操作对其它格子的贡献。如果一个格子进行了操作1和操作2中的一个，它对相邻格子的贡献为1。如果没有操作或进行了两个操作，贡献为0。
- 贡献为1时，这个格子的颜色是可以任意指定的，因为只要换一种操作就可以改变它的颜色。两种颜色的状态可以合并。
- 每个格子只要记录3种状态。时间复杂度$O(nm3^n)$.

Periodic Binary String

- T是一个长度为k的01串。S是以T为周期的无穷01串。例如T="101"，则S="101101101..."。

Periodic Binary String

- T是一个长度为k的01串。S是以T为周期的无穷01串。例如T=“101”，则S=“101101101...”.
- $f[l,r]$表示S中第l位到第r位的子串看作二进制数的值。
- 问有多少个T满足$f[l,r] \equiv x \pmod p$.

Periodic Binary String

- T是一个长度为k的01串。S是以T为周期的无穷01串。例如T=“101”，则S=“101101101...”.
- $f[l,r]$表示S中第l位到第r位的子串看作二进制数的值。
- 问有多少个T满足$f[l,r] \equiv x \pmod{p}$.
- p为质数，$p, l, r, k \leq 10^{18}$，答案模10^9+7.

Solution

- 由于是循环串，我们只关心长度不关心位置。
- 设$r-l+1=qk+t$，则子串$[l,r]$中，一个长度为t的01串出现$q+1$次，一个长度为$k-t$的01串出现q次，它们不相关。

Solution

- 由于是循环串，我们只关心长度不关心位置。
- 设$r-l+1=qk+t$，则子串$[l,r]$中，一个长度为t的01串出现$q+1$次，一个长度为$k-t$的01串出现q次，它们不相关。
- 特判$t=0$.
- 令$c_1=1+2^k+2^{2k}+\cdots+2^{qk}$，$c_2=2^r+2^{r+k}+,+2^{r+(q-1)k}$，方程化为$c_1a+c_2b\equiv x\pmod p, 0\le a<2^t, 0\le b<2^{k-t}$. 求该方程解数。

Solution

- 特判$c_1 = 0$和$c_2 = 0$.
- 乘以c_1^{-1}，化为求$a \equiv y - cb \pmod p, 0 \le a < 2^t, 0 \le b < 2^{k-t}$的解数.

Solution

- 特判$c_1 = 0$和$c_2 = 0$.
- 乘以c_1^{-1}，化为求$a \equiv y - cb \pmod{p}, 0 \le a < 2^t, 0 \le b < 2^{k-t}$的解数.
- 若$2^t = q_1 p + r_1$，每个b在$0 \le a < q_1 p$的范围内都有q_1个解，这里的解数是$2^{k-t}\left\lfloor \frac{2^t}{p} \right\rfloor$，只要再求$0 \le a < r_1$的解数即可。同理把$b$的范围缩小到$p$以内。
- 答案是模10^9+7，这里求$\left\lfloor \frac{2^t}{p} \right\rfloor \bmod 10^9+7$，先对$(10^9+7)p$取模，再除$p$下取整。要高精度。

Solution

- 特判$c_1 = 0$和$c_2 = 0$.
- 乘以c_1^{-1}，化为求$a \equiv y - cb \pmod p, 0 \le a < 2^t, 0 \le b < 2^{k-t}$的解数.
- 若$2^t = q_1 p + r_1$，每个b在$0 \le a < q_1 p$的范围内都有q_1个解，这里的解数是$2^{k-t}\left\lfloor \frac{2^t}{p} \right\rfloor$，只要再求$0 \le a < r_1$的解数即可。同理把$b$的范围缩小到$p$以内。
- 答案是模10^9+7，这里求$\left\lfloor \frac{2^t}{p} \right\rfloor \bmod 10^9+7$，先对$(10^9+7)p$取模，再除$p$下取整。要高精度。
- 求有多少$0 \le b < r_2$，满足$0 \le (y - cb) \bmod p < r_1$.

Solution

- $$\sum_{b=0}^{r_2-1}[(y-cb) \bmod p < r_1] = \sum_{b=0}^{r_2-1}\left\lfloor \frac{y-cb}{p} \right\rfloor - \left\lfloor \frac{y-cb-r_1}{p} \right\rfloor$$
- 问题转化为求$\sum_{i=0}^{n-1}\left\lfloor \frac{a+di}{m} \right\rfloor$.

Solution

- $$\sum_{b=0}^{r_2-1}[(y-cb) \bmod p < r_1] = \sum_{b=0}^{r_2-1}\left\lfloor\frac{y-cb}{p}\right\rfloor - \left\lfloor\frac{y-cb-r_1}{p}\right\rfloor$$
- 问题转化为求$\sum_{i=0}^{n-1}\left\lfloor\frac{a+di}{m}\right\rfloor$.
- 利用
$$\sum_{i=0}^{n-1}\left\lfloor\frac{a+di}{m}\right\rfloor = \sum_{k=0}^{l-1}\left\lfloor\frac{(dn+a) \bmod m + mk}{d}\right\rfloor, l = \left\lfloor\frac{a+dn}{m}\right\rfloor$$
每次将整数提出保持$d < m$，可以用辗转相除法求解。

Solution

- $$\sum_{b=0}^{r_2-1}[(y-cb) \bmod p < r_1] = \sum_{b=0}^{r_2-1}\left\lfloor\frac{y-cb}{p}\right\rfloor - \left\lfloor\frac{y-cb-r_1}{p}\right\rfloor$$
- 问题转化为求$\sum_{i=0}^{n-1}\left\lfloor\frac{a+di}{m}\right\rfloor$.
- 利用
  $$\sum_{i=0}^{n-1}\left\lfloor\frac{a+di}{m}\right\rfloor = \sum_{k=0}^{l-1}\left\lfloor\frac{(dn+a) \bmod m + mk}{d}\right\rfloor, l = \left\lfloor\frac{a+dn}{m}\right\rfloor$$
  每次将整数提出保持$d < m$，可以用辗转相除法求解。
- 不计高精度$O(logn)$

Proof

$$\sum_{i=0}^{n-1}\left\lfloor\frac{a+di}{m}\right\rfloor=\sum_{i=0}^{n-1}\sum_{k=0}^{l-1}\left[\left\lfloor\frac{a+di}{m}\right\rfloor\geq k+1\right]$$

$$=\sum_{i=0}^{n-1}\sum_{k=0}^{l-1}[a+di\geq(k+1)m]$$

$$=\sum_{k=0}^{l-1}\sum_{i=0}^{n-1}\left[i\leq n-1-\frac{km+m-a}{d}\right]$$

$$=\sum_{k=0}^{l-1}\left\lfloor n-\frac{(l-1-k)m+m-a}{d}\right\rfloor$$

$$=\sum_{k=0}^{l-1}\left\lfloor\frac{(dn+a)\bmod m+mk}{d}\right\rfloor$$

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：

NEERC14-15 I

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：
- 不存在两个数$0 \le i < j < n$满足线段$[\min(a_i, a_{i+1}), \max(a_i, a_{i+1})]$和线段$[\min(a_j, a_{j+1}), \max(a_j, a_{j+1})]$有交但不互相包含

NEERC14-15 I

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：
- 不存在两个数$0 \le i < j < n$满足线段$[\min(a_i, a_{i+1}), \max(a_i, a_{i+1})]$和线段$[\min(a_j, a_{j+1}), \max(a_j, a_{j+1})]$有交但不互相包含
- a_0一直等于0

NEERC14-15 I

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：
- 不存在两个数$0 \le i < j < n$满足线段$[\min(a_i, a_{i+1}), \max(a_i, a_{i+1})]$和线段$[\min(a_j, a_{j+1}), \max(a_j, a_{j+1})]$有交但不互相包含
- a_0一直等于0
- 比如1 5 3 4和1 5 3 2是两个合法方案并且在后面加上任何一个数都将导致不合法。3 5 1不合法因为$[0, 3]$和$[1, 5]$相交但不互相包含。

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：
- 不存在两个数$0 \le i < j < n$满足线段$[\min(a_i, a_{i+1}), \max(a_i, a_{i+1})]$和线段$[\min(a_j, a_{j+1}), \max(a_j, a_{j+1})]$有交但不互相包含
- a_0一直等于0
- 比如1 5 3 4和1 5 3 2是两个合法方案并且在后面加上任何一个数都将导致不合法。3 5 1不合法因为$[0, 3]$和$[1, 5]$相交但不互相包含。
- $n \le 50$

NEERC14-15 I

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：
- 不存在两个数$0 \le i < j < n$满足线段$[\min(a_i, a_{i+1}), \max(a_i, a_{i+1})]$和线段$[\min(a_j, a_{j+1}), \max(a_j, a_{j+1})]$有交但不互相包含
- a_0一直等于0
- 比如1 5 3 4和1 5 3 2是两个合法方案并且在后面加上任何一个数都将导致不合法。3 5 1不合法因为$[0, 3]$和$[1, 5]$相交但不互相包含。
- $n \le 50$
- 其实刚才是骗你的，$n \le 2000$

- 给你一个$1 \sim n$的全排列，要求选出一个尽量长的子序列满足：
- 不存在两个数$0 \le i < j < n$满足线段$[\min(a_i, a_{i+1}), \max(a_i, a_{i+1})]$和线段$[\min(a_j, a_{j+1}), \max(a_j, a_{j+1})]$有交但不互相包含
- a_0一直等于0
- 比如1 5 3 4和1 5 3 2是两个合法方案并且在后面加上任何一个数都将导致不合法。3 5 1不合法因为$[0, 3]$和$[1, 5]$相交但不互相包含。
- $n \le 50$
- 其实刚才是骗你的，$n \le 2000$
- 其实刚才还是骗你的，$n \le 200000$

- 设b_i为i在a中的位置，那么经过观察发现b是由一段上升序列加一段下降序列组成的。

- 设b_i为i在a中的位置，那么经过观察发现b是由一段上升序列加一段下降序列组成的。
- 证明：$n = 1$时显然满足

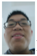
- 设b_i为i在a中的位置，那么经过观察发现b是由一段上升序列加一段下降序列组成的。
- 证明：$n = 1$时显然满足
- 假如$n = k$时满足条件，设$a_p = k$，那么根据定义$k + 1$只能在a_p前面或者a_p后面两个位置中选择一个插入。但不管插入在哪个位置依然满足性质。

- 设b_i为i在a中的位置，那么经过观察发现b是由一段上升序列加一段下降序列组成的。
- 证明：$n = 1$时显然满足
- 假如$n = k$时满足条件，设$a_p = k$，那么根据定义$k + 1$只能在a_p前面或者a_p后面两个位置中选择一个插入。但不管插入在哪个位置依然满足性质。
- 先求出原串的b数组，并求出b数组的前后缀最长上升子序列然后扫一遍即可。

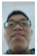
CERC14 L

- 有n个，每个会在a_i时刻出现在离你d_i米的地方，并在b_i时刻消失。

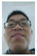
CERC14 L

- 有n个，每个会在a_i时刻出现在离你d_i米的地方，并在b_i时刻消失。
- 任意时刻你可以花费R点体力并把所有与你距离不超过R的都膜一遍。

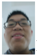
CERC14 L

- 有n个，每个会在a_i时刻出现在离你d_i米的地方，并在b_i时刻消失。
- 任意时刻你可以花费R点体力并把所有与你距离不超过R的都膜一遍。
- 问膜遍所有的最小的体力消耗。

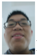
CERC14 L

- 有n个，每个会在a_i时刻出现在离你d_i米的地方，并在b_i时刻消失。
- 任意时刻你可以花费R点体力并把所有与你距离不超过R的都膜一遍。
- 问膜遍所有的最小的体力消耗。
- $a_i, b_i, d_i \leq 10000$

CERC14 L

- 有n个，每个会在a_i时刻出现在离你d_i米的地方，并在b_i时刻消失。
- 任意时刻你可以花费R点体力并把所有与你距离不超过R的都膜一遍。
- 问膜遍所有的最小的体力消耗。
- $a_i, b_i, d_i \le 10000$
- $n \le 1000000$

CERC14 L

- 有n个，每个会在a_i时刻出现在离你d_i米的地方，并在b_i时刻消失。
- 任意时刻你可以花费R点体力并把所有与你距离不超过R的都膜一遍。
- 问膜遍所有的最小的体力消耗。
- $a_i, b_i, d_i \leq 10000$
- $n \leq 1000000$
- 其实刚才是骗你的，$n \leq 500$

CERC14 L

- 贪心?

CERC14 L

- 贪心？
- 考虑$O(n^3)$dp：先离散，用$f(i,j)$表示处理掉完全包含在$i \sim j$这段区间内的的最小花费。

CERC14 L

- 贪心？
- 考虑$O(n^3)$dp：先离散，用$f(i,j)$表示处理掉完全包含在$i \sim j$这段区间内的的最小花费。
- 那么这个区间中的最远的一个一定是会被和它距离刚好相同的半径膜掉的。

CERC14 L

- 贪心？
- 考虑$O(n^3)$dp：先离散，用$f(i,j)$表示处理掉完全包含在$i \sim j$这段区间内的的最小花费。
- 那么这个区间中的最远的一个一定是会被和它距离刚好相同的半径膜掉的。
- 枚举在x时刻膜掉最远的，那么

CERC14 L

- 贪心?
- 考虑$O(n^3)$dp：先离散，用$f(i,j)$表示处理掉完全包含在$i\sim j$这段区间内的的最小花费。
- 那么这个区间中的最远的一个一定是会被和它距离刚好相同的半径膜掉的。
- 枚举在x时刻膜掉最远的，那么
- $f(i,j)=\min(f(i,x-1)+f(x+1,j)+R)$

CERC14 L

- 贪心？
- 考虑$O(n^3)$dp：先离散，用$f(i,j)$表示处理掉完全包含在$i\sim j$这段区间内的的最小花费。
- 那么这个区间中的最远的一个一定是会被和它距离刚好相同的半径膜掉的。
- 枚举在x时刻膜掉最远的，那么
- $f(i,j)=\min(f(i,x-1)+f(x+1,j)+R)$
- 其中R为最远那个的距离。

SEERC14 B

- 给你一个长为n的数字串。

SEERC14 B

- 给你一个长为n的数字串。
- 首先你需要把它切成两份（长度可以是0）并交换他们的位置（如“TRZtrz”可以变成“trzTRZ”）

SEERC14 B

- 给你一个长为n的数字串。
- 首先你需要把它切成两份（长度可以是0）并交换他们的位置（如“TRZtrz”可以变成“trzTRZ”）
- 接下来你需要把它切成k份，要求切完以后最大的数字最小。

SEERC14 B

- 给你一个长为n的数字串。
- 首先你需要把它切成两份（长度可以是0）并交换他们的位置（如“TRZtrz”可以变成“trzTRZ”）
- 接下来你需要把它切成k份，要求切完以后最大的数字最小。
- 如7654321输出176

SEERC14 B

- 给你一个长为n的数字串。
- 首先你需要把它切成两份（长度可以是0）并交换他们的位置（如“TRZtrz”可以变成“trzTRZ”）
- 接下来你需要把它切成k份，要求切完以后最大的数字最小。
- 如7654321输出176
- 数据范围？

SEERC14 B

- 容易发现最大串的长度$t = \frac{n-1}{m} + 1$

- 容易发现最大串的长度$t = \frac{n-1}{m} + 1$
- 二分最大串，这样就可以求出从每个位置开始到底是能走t步还是$t-1$步

SEERC14 B

- 容易发现最大串的长度$t = \frac{n-1}{m} + 1$
- 二分最大串，这样就可以求出从每个位置开始到底是能走t步还是$t-1$步
- 当然二分需要先对后缀排序，判断步数需要比较两个后缀的大小。所以用一个后辍数组就可以了。

- 容易发现最大串的长度$t = \frac{n-1}{m} + 1$
- 二分最大串，这样就可以求出从每个位置开始到底是能走t步还是$t-1$步
- 当然二分需要先对后缀排序，判断步数需要比较两个后缀的大小。所以用一个后辍数组就可以了。
- 通过倍增我们就能算出每个串走k步后的位置。

SEERC14 B

- 容易发现最大串的长度$t = \frac{n-1}{m} + 1$
- 二分最大串，这样就可以求出从每个位置开始到底是能走t步还是$t-1$步
- 当然二分需要先对后缀排序，判断步数需要比较两个后缀的大小。所以用一个后辍数组就可以了。
- 通过倍增我们就能算出每个串走k步后的位置。
- 假如有至少一个位置能用k步走完一圈那么就是合法的。

SEERC14 B

- 容易发现最大串的长度$t = \frac{n-1}{m} + 1$
- 二分最大串，这样就可以求出从每个位置开始到底是能走t步还是$t-1$步
- 当然二分需要先对后缀排序，判断步数需要比较两个后缀的大小。所以用一个后辍数组就可以了。
- 通过倍增我们就能算出每个串走k步后的位置。
- 假如有至少一个位置能用k步走完一圈那么就是合法的。
- 时间复杂度$O(n \log n)$

- fsy和fhygd在看NBA，但是网线只有一根，所以他们决定以回合制的方式来决定每个人看哪些NBA。

- fsy和fhygd在看NBA，但是网线只有一根，所以他们决定以回合制的方式来决定每个人看哪些NBA。
- 具体来说，双方轮流行动，每次可以选择看下一场NBA或者不动。

- fsy和fhygd在看NBA，但是网线只有一根，所以他们决定以回合制的方式来决定每个人看哪些NBA。
- 具体来说，双方轮流行动，每次可以选择看下一场NBA或者不动。
- 但是不动是很累的，双方各有一个精力值，每次不动将花费一点精力。当然假如没有精力了就不能不动。

- fsy和fhygd在看NBA，但是网线只有一根，所以他们决定以回合制的方式来决定每个人看哪些NBA。
- 具体来说，双方轮流行动，每次可以选择看下一场NBA或者不动。
- 但是不动是很累的，双方各有一个精力值，每次不动将花费一点精力。当然假如没有精力了就不能不动。
- 现在有n场NBA，第i场NBA看了以后可以获得a_i的愉♁悦值并回复b_i点精力值。

- fsy和fhygd在看NBA，但是网线只有一根，所以他们决定以回合制的方式来决定每个人看哪些NBA。
- 具体来说，双方轮流行动，每次可以选择看下一场NBA或者不动。
- 但是不动是很累的，双方各有一个精力值，每次不动将花费一点精力。当然假如没有精力了就不能不动。
- 现在有n场NBA，第i场NBA看了以后可以获得a_i的愉♂悦值并回复b_i点精力值。
- 告诉你每个人的初始精力值A,B，fsy想知道先手方最多能获得多少点愉♂悦值

ICPCOOC14 I

- fsy和fhygd在看NBA，但是网线只有一根，所以他们决定以回合制的方式来决定每个人看哪些NBA。
- 具体来说，双方轮流行动，每次可以选择看下一场NBA或者不动。
- 但是不动是很累的，双方各有一个精力值，每次不动将花费一点精力。当然假如没有精力了就不能不动。
- 现在有n场NBA，第i场NBA看了以后可以获得a_i的愉ô悦值并回复b_i点精力值。
- 告诉你每个人的初始精力值A,B，fsy想知道先手方最多能获得多少点愉ô悦值
- $a_i \leq 0, 0 \leq A, B, b_i \leq 10^9, \Sigma a_i \leq 150, n \leq 150$

- 容易发现精力值的大小是没有用的，只有差值有用。因为精力值占优势的那一方指使劣势方看下一场NBA那么劣势方也没有办法。

- 容易发现精力值的大小是没有用的，只有差值有用。因为精力值占优势的那一方指使劣势方看下一场NBA那么劣势方也没有办法。
- 考虑$f(i,j)$表示后i场NBA先手方至少需要比后手方多多少精力才能获得j点愉♂悦值。

- 容易发现精力值的大小是没有用的，只有差值有用。因为精力值占优势的那一方指使劣势方看下一场NBA那么劣势方也没有办法。
- 考虑$f(i,j)$表示后i场NBA先手方至少需要比后手方多多少精力才能获得j点愉♂悦值。
- 那么假如先手方选择看：
- $f(i, j + a_i) = -f(i + 1, sum(i + 1) - j) - b_i + 1$

- 容易发现精力值的大小是没有用的，只有差值有用。因为精力值占优势的那一方指使劣势方看下一场NBA那么劣势方也没有办法。
- 考虑$f(i,j)$表示后i场NBA先手方至少需要比后手方多多少精力才能获得j点愉♂悦值。
- 那么假如先手方选择看：
- $f(i, j + a_i) = -f(i + 1, sum(i + 1) - j) - b_i + 1$
- 选择不看：
- $f(i, j) = \max(f(i + 1, j) + b_i, 0) + 1$

Epilogue

- 谢谢大家。
- 祝大家省选顺利。