1001

对 $link-cut-tree$ 有所了解的选手不难发现，操作 1 是在模拟 lct 的 $access$ 操作

于是操作 1 可以在 lct 虚实边切换的时候用一个线段树区间修改，同时维护操作 4 的答案

操作 2 相当于单点询问，操作 3 相当于询问子树和然后再扣掉点 u 的答案

复杂度 $O(nlog^2n)$

1002

可以通过枚举字符a,b在字符串出现的次数x,y得到:

$$ans[i][j]=\sum_{x=0}^{L}\sum_{y=0}^{L-x}[n\mid x-i][n\mid y-j]\binom{L}{x}\binom{L-x}{y}(k-2)^{L-xy}$$

$$=\sum_{x=0}^{L}\sum_{y=0}^{L-x}\frac{1}{n}\sum_{p=0}^{n-1}w_n^{p\times(x-i)}\frac{1}{n}\sum_{q=0}^{n-1}w_n^{q\times(y-j)}\binom{L}{x}\binom{L-x}{y}(k-2)^{L-xy}$$

由单位根反演可得

$$=\frac{1}{n^2}\sum_{x=0}^{L}\sum_{y=0}^{L-x}\sum_{p=0}^{n-1}w_n^{p\times(x-i)}\sum_{q=0}^{n-1}w_n^{q\times(y-j)}\binom{L}{x}\binom{L-x}{y}(k-2)^{L-xy}$$

$$=\frac{1}{n^2}\sum_{p=0}^{n-1}\sum_{q=0}^{n-1}\sum_{x=0}^{L}\sum_{y=0}^{L-x}w_n^{p\times(x-i)}w_n^{q\times(y-j)}\binom{L}{x}\binom{L-x}{y}(k-2)^{L-xy}$$

$$=\frac{1}{n^2}\sum_{p=0}^{n-1}\sum_{q=0}^{n-1}\sum_{x=0}^{L}\sum_{y=0}^{L-x}w_n^{px}w_n^{qy}\binom{L}{x}\binom{L-x}{y}(k-2)^{L-xy}w_n^{-pi}w_n^{-qj}$$

$$=\frac{1}{n^2}\sum_{p=0}^{n-1}\sum_{q=0}^{n-1}(w_n^p+w_n^q+k-2)^Lw_n^{-pi}w_n^{-qj}$$

我们令

$$A[i][p]=w_n^{-ip}\ B[p][q]=\frac{1}{n^2}(w_n^p+w_n^q+k-2)^L\ C[q][j]=w_n^{-qj}$$

则$ans=A\times B\times C$

总时间复杂度为$O(n^3+n^2logL)$

1003

【等价题意】

n维空间中，求m的最大值，使得你可以找到m个点(自己给定坐标)，满足：

无论对这m个点如何二染色，也就是对于2^m种染色方案中的每一种，都总存在一个$n-1$维超平面，严格分开这两种颜色的点。

【结论】

$$m_{max}=n+1$$

【证明思路】

显然有单调性，可以分两步证明结论：

- 证明$m = n + 1$可行
  - 即：可构造一组点的坐标$(x_0, x_1, x_2, \cdots, x_n \in R^n)$，证明其任意一组染色方案，都可以找到一个线性超平面将其染成这个方案
  - 这里染成这个方案指的是，超平面(> 0)一侧是黑，一侧(< 0)是白
- 证明$m \geq n + 2$无解
  - 即：证明$n + 2$个点无论如何放置(设计坐标)，都总存在一种染色方案，使得没有一个线性超平面可以将其染成这个方案

【完整证明】

第一步证明：

令$x_0 = (0, 0, 0, \cdots, 0), x_i =$第$i$个下标是1其他是0的01向量，$x_0, x_i \in R^n, i \in [1, n]$； 令$f_w(x) = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n = w^T [1\ x]^T$；

注意向量默认是列向量，这里w和x是向量；

我们的线性分类超平面就是：$f_w(x) \geq 0$

令染色方案$S = \{(x_i, y_i)\}, i \in [0, n], y_i \in \{+1, -1\}$，我们直接构造一个$w$，使得$f_w(x_i) = y_i$ 那么：$f(x_i) = y_i, i \in [0, n]$，可解方程组确定$w$

$w_0 = y_0$

$w_0 + w_i = y_i, i \in [1, n]$

第二步证明：

以$x_i = [1\ x_i]$代替原输入x_i； 现在输入x_i，我们构造$y_i \in \{-1, +1\}$，使得w不存在

考虑在R^{n+1} 空间中(因为我们对输入加了一维1)，$n + 2$个点一定是线性相关的； 存在一组不全为0的系数，使得它们的加权和为0，令那个不为0的系数对应的项为x_0，那么x_0 可以由$x_i, i \in [1, n + 1]$线性表示；

令：

$$x_0 = \sum_{i=1}^{n+1} a_i x_i$$

我们先构造$y_i, i \in [1, n + 1]$，再构造y_0，想办法使得w无解：

思路是：让$w^T a_i x_i \geq 0$，但是：

$$w^T x_0 = \sum_{i=1}^{n+1} w^T a_i x_i < 0$$

从而制造矛盾。

构造方法：

对于$i \in [1, n + 1]$，如果$a_i \geq 0, y_i = 1$，如果$a_i < 0, y_i = -1$。且$y_0 = -1$。

如果$y_i = -1$，则$f_w(x_i) = w^T x_i < 0$，$w^T a_i x_i \geq 0$

如果$y_i = 1$，则$f_w(x_i) = w^T x_i > 0$，$w^T a_i x_i \geq 0$

这样的话：$w^T a_i x_i \geq 0$总是成立，

$$w^T x_0 = \sum_{i=1}^{n+1} w^T a_i x_i \geq 0$$

但是由于$y_0 = -1, f_w(x_0) = w^T x_0 < 0$矛盾!

所以这样的y构造，使得w无解，则不存在$n-1$维超平面f使得f可以严格分割这个染色方案S

1004

定义$F[i,j]$表示字符串S中分别以i,j为左端点的两个子串满足k-匹配的最大长度。

换句话说，$F[i,j]$等于使得$S[i,i+L-1]$与$S[j,j+L-1]$满足k-匹配的最大L。

考虑如何计算所有的$F[i,j](1 \leq i < j \leq n)$

- 我们枚举两个子串的左端点之差d $(d \in [1, n-1])$
- 对于每一个d，利用k-匹配的单调性，通过双指针，我们可以在线性时间内计算出所有的$F[i,i+d](1 \leq i \leq n-d)$
- 因此，我们可以用$O(n^2)$的时间复杂度计算出所有的$F[i,j]$

考虑如何计算串S的所有分割情况下的答案

- 定义$t(2 \leq t \leq n)$分割是将S分为$A = S[1,t-1]$和$B = S[t,n]$
- 对于t分割，不难发现，我们要统计的答案为$\sum_{i=1}^{t-1} \sum_{j=t}^{n} G[i,j]$，其中$G[i,j] = min(F[i,j], t-i)$
- 不妨从大到小枚举t，过程中维护G对答案的贡献
- 注意到G[i,j]的值不超过$t-i$。因此可以对每个i维护一个$count_i$数组来对有贡献的$G[i,j]$计数。换句话说，$count_i[x]$表示$\{G[i,j] | 2 \leq j \leq n\}$中此时对答案有贡献且值为x的个数
- 同时我们对每个i再维护一个max_i表示此时$\{G[i,j] | 2 \leq j \leq n\}$中对答案有贡献的最大值。对于t的每次减小，我们通过维护$count_i$数组和max_i，来计算答案的变化。
- 注意到过程中$G[i,j] \leq t-i$，因而max_i在单调减少。而每次新增产生贡献的G[i,j]的数量为$O(n)$且值也不超过$t-i$。因此整个过程的时间复杂度是$O(n^2)$

综上所述，整个算法的时间复杂度为$O(n^2)$

1005

记$\lambda_i = W_{i,i}, d_i = \sum_{j=1}^{n} W_{i,j}$

则有转移方程

$$A_{i,i} = \frac{\lambda_i}{\lambda_i + d_i} * 1 + \sum_{j=1, j \neq i}^{n} \frac{W_{i,j}}{\lambda_i + d_i} A_{j,i}$$

$$A_{i,j} = \sum_{k=1, k \neq i}^{n} \frac{W_{i,k}}{\lambda_i + d_i} A_{k,j} (i \neq j)$$

构造无自环矩阵W，自环的对角矩阵Λ，W度数矩阵D

$$(I - (\Lambda + D)^{-1} W) A = (\Lambda + D)^{-1} \Lambda$$

第一步列转移方程

第二步写出矩阵形式

第三步解矩阵方程，套板子

算出A^{-1}矩阵，套个求逆板子即可

1006

按题意模拟即可，输出树的节点个数即可

1007

对于最大代价的情况，显然我们需要填充满$n \times n \times n$的空间，并且为了最大化代价，需要将每一砖块从最高处丢下，因此有最大代价：$\sum_{x=1}^{n}\sum_{y=1}^{n}\sum_{z=1}^{n} x \times y^2 \times n = n^2 \times \frac{(n+1)n}{2} \times \frac{n(n+1)(2n+1)}{6}$

对于最小代价的情况，由于重力作用，需将底面铺满。为满足另外两个视角，还需要立一面随着x递增，y递减的竖墙。因此有最小代价：
$\sum_{x=1}^{n}\sum_{y=1}^{n} x \times y + \sum_{y=1}^{n}\sum_{z=2}^{n}(n+1-y) \times y^2 \times z = \frac{(n+1)^2n^2}{4} + \frac{(2+n)(n-1)}{2} \times (\frac{n(n+1)^2(2n+1)}{6} - \frac{n^2(n+1)^2}{4})$

1008

注意到题目是求条件概率

我们可以通过高维前缀和求出对于一个集合T有多少超集U，记为$cnt[T]$

显然答案为$\sum_{S\in[0,2^n)}\sum_{T\in[0,2^n)}\frac{cnt[T \vee S]}{cnt[S]}$

$T \vee S$又可以用高维前缀和再一次优化，记为$cnt0[T]$

显然答案为$\sum_{S\in[0,2^n)}\frac{cnt0[S] * 2^{bit(S)}}{cnt[S]}$

$bit(S)$表示为S中1的个数

所以复杂度为$O(n2^n)$

1009

暴力做法时间复杂度$O(k * nlogn)$

做法1、2：权值线段树、树状数组

代码略

做法3：考虑k如果变大 考虑根据出现次数分成>=sqrt(n)和<sqrt(n)的部分

复杂度$n\sqrt{n}*logn$

使用桶/哈希表/集合计数x的出现次数 按从大到小排序

1.当出现次数大于等于$\sqrt{n}$时，有出现的类别最多$\sqrt{n}$种，按做法1和2的时间复杂度,该部分$k\leq\sqrt{n}$，所以该部分复杂度是$\sqrt{n}*nlogn$

2.当出现次数小于$\sqrt{n}$时，考虑每一个数每次出现的位置，假设一个数出现t_i次,记录出现的位置,对于每一个数考虑枚举l_i,r_i,t_i，考虑左边右边合法的位置，对于一个数可以$O(ti^2)$求得，有
$f=t_1^2+t_2^2+...+t_n^2\leq\sqrt{n}(t_1+t_2+..+t_n)=n\sqrt{n}$ 所以该部分复杂度最大$n\sqrt{n}$

做法4：考虑对两种算法均摊

令分块的界限为d

1.当出现次数大于等于d时，有出现的类别最多n/d种，按做法1和2的时间复杂度,该部分$k\leq n/d$，所以该部分复杂度是$n/d*nlogn$

2.当出现次数小于d时，考虑每一个数每次出现的位置，假设一个数出现t_i次,记录出现的位置,对于每一个数考虑枚举l_i,r_i,t_i，考虑左边右边合法的位置，对于一个数可以$O(ti^2)$求得，有
$f=t_1^2+t_2^2+...+t_n^2\leq\sqrt{n}(t_1+t_2+..+t_n)=nd$ 所以该部分复杂度最大nd

解得$d=\sqrt{nlogn}$

复杂度为$O(n^{3/2}\sqrt{log_2n})$

做法5：树状数组

复杂度为$O(nlog_2n)$

使用桶/哈希表/集合计数x的出现 依次遍历每一个数 考虑合并相邻两块同一个数的影响

答案是把每个位置前缀和得到的去数前面有几个比前缀和小的

原始数组是 1 1 2 1 变一下 1 1 -1 1 前缀和一下 0 1 2 1 2 贡献 1 2 1 3

证明如果前缀和比之前某个前缀和大意味着前缀和-前缀和的这个区间 是符合条件的 SUM[NOW]-SUM[LAST_I]>0=>LAST_I+1->NOW区间满足条件 然后需要处理就是 每一个1的地方处理前面有几个-1对答案的贡献 当前这个1对答案的贡献 假如有k种数，每个数出现t次，每一次的代价是logn，对于一种数的代价是t_ilogn，sum(t_i)=n，复杂度就是$O(nlog_2n)$。可以通过。

做法6：

复杂度为$O(n)$

考虑对做法5进行优化，设sum为当前前缀和，f1[sum]表示前缀和为sum的点有f1[sum]个，对于每个点的贡献now就是所有小于sum的f1[sum]的和，如果接下来有一堆-1，就一下跳到这堆-1的末尾，需要做一次差分数组f2,在起点做-1标记，终点做+1标记，走到i时若f2[sum]不为0，则用f2[sum]更新f1，同时更新f2[sum+1]。代码略。

1010

$$y_i = \frac{e^{\frac{x_i+g_i-(x_k+y_k)}{\tau}}}{\sum_{j=1}^{k} e^{\frac{x_j+g_j-(x_k+g_k)}{\tau}}} \ for\ i = 1, \dots, k$$

$$u_i = x_i + g_i - (x_k + g_k)\ for\ i = 1, .., k-1$$

其中g_i是伪代码中的z，为随机变量，满足g_i服从$gumbel$分布.

$gumbel$分布的概率密度函数$f(z, \mu) = e^{\mu - z - e^{\mu - z}}$

$$p(u_1, \dots, u_{k-1}) = \int_{-\infty}^{\infty} dg_k p(u_1, \dots, u_k | g_k) p(g_k)$$

$$= \int_{-\infty}^{\infty} dg_k p(g_k) \Pi_{i=1}^{k-1} p(u_i | g_k)$$

$$= \int_{-\infty}^{\infty} dg_k f(g_k, 0) \Pi_{i=1}^{k-1} f(x_k + g_k, x_i - u_i)$$

$$= \int_{-\infty}^{\infty} dg_k e^{-g_k - e^{-g_k}} \Pi_{i=1}^{k-1} e^{x_i - u_i - x_k - g_k - e^{x_i - u_i - x_k - g_k}}$$

先求出u的联合概率密度函数

$$p(u_1, \dots, u_{k-1} = \delta(u_k = 0) \int_0^{\infty} dv \frac{1}{v} v e^{x_k - v} \Pi_{i=1}^{k-1} v e^{x_i - u_i - x_k - g_k - e^{x_i - u_i - x_k - g_k}}$$

$$= e^{x_k + \sum_{i=1}^{k-1}(x_i - u_i)} (e^{x_k} + \sum_{i=1}^{k-1} e^{x_i - u_i})^{-k} \Gamma(k)$$

$$= \Gamma(k) (\prod_{i=1}^{k} e^{x_i - u_i}) (\sum_{i=1}^{k} e^{x_i - u_i})^{-k}$$

$$y_{1:\ k-1} = h(u_{1:k-1}), h_i(u_{1:k-1}) = \frac{e^{u_i}\tau}{1 + \sum_{j=1}^{k-1} e^{\frac{u_j}{\tau}}}, \forall i = 1, \dots, k-1$$

接下来要求的是y的联合概率密度函数

$$\sum_{i=1}^{k} y_i = 1$$

$$y_k = 1 - \sum_{j=1}^{k-1} y_j$$

$$p(y_{1:k}) = p(h^{-1}(y_{1:k-1})) det(\frac{\partial h^{-1}(y_{1:k-1})}{\partial y_{1:k-1}})$$

采用换元法，由概率论知识：需要变换系数：雅可比行列式

需要求$h^{-1}(\dots), p(.), Jacobian(h^{-1}), det\ Jacobian(h^{-1})$

$$h^{-1}(y_{1:k-1}) = \tau * (ln(y_i) - ln(1 - \sum_{j=1}^{k-1} y_j)) = \tau * (ln(y_i) - ln(y_k))$$

$$\frac{\partial h^{-1}(y_{1:k-1})}{\partial y_{1:k-1}} = \tau * (diag(\frac{1}{y_{1:k-1}}) + \frac{1}{y_k}) = \begin{pmatrix} \frac{1}{y_1} + \frac{1}{y_k} & \frac{1}{y_k} & \cdots & \frac{1}{y_k} \\ \frac{1}{y_k} & \frac{1}{y_2} + \frac{1}{y_k} & \cdots & \frac{1}{y_k} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{1}{y_k} & \frac{1}{y_k} & \cdots & \frac{1}{y_n} + \frac{1}{y_k} \end{pmatrix}$$

$$det(\frac{\partial h^{-1}(y_{1:k-1})}{\partial y_{1:k-1}}) = \tau^{k-1} \Pi_{j=1}^{k-1} y_j^{-1}$$

$$p(y_1,y_2,..,y_k)=\Gamma(k)(\prod_{i=1}^{k}e^{x_i}\frac{y_k^\tau}{y_i^\tau})(\sum_{i=1}^{k}e^{x_i}\frac{y_k^\tau}{y_i^\tau})^{-k}\tau^{k-1}\prod_{i=1}^{k}y_i^{-1}$$

$$=\Gamma(k)\tau^{k-1}(\sum_{i=1}^{k}\frac{e^{x_i}}{y_i^\tau})^{-k}\prod_{i=1}^{k}\frac{e^{x_i}}{y_i^{\tau+1}}$$

1011

1.当$x=y$时

易证$sg_i=\lfloor\frac{i}{2x}\rfloor\times x+i\mod x$, 此时题目转为$nim$博弈, $\bigoplus_{i=1}^{n}sg_{a_i}>0$则先手胜, 否则先手败

以下我们令$sg_i=i$, 也就是nim博弈的sg值

2.当$x<y$时

不妨设$a_1\le a_2\le a_3\le\ldots\le a_{n-1}\le a_n$

如果$a_n<x$, 此时题目是nim博弈, 所以只讨论$a_n\ge x$的情况

当先手能通过一次操作使$\max_{i=1}^{n}a_i<x$, 此时胜负由$\bigoplus_{i=1}^{n}a_i$决定, 若先手能找到一个正数z, 且$\bigoplus_{i=1}^{n}a_i\bigoplus a_n\bigoplus(a_n-z)=0$, 此时先手必胜, 否则先手会考虑其他策略

当先手不能通过一次操作使$max_{i=1}^{n}a_i<x$, 此时我们将先手操作后的石子分三种情况考虑

①$a_{n-1}<x$, $a_n\ge x$

若此时$\bigoplus_{i=1}^{n}a_i\ne0$, 必定存在一堆石子$j$和正数$z$, 使得$\bigoplus_{i=1}^{n}a_i\bigoplus a_j\bigoplus(a_j-z)=0$, 若$z\ne y$, 此时后手应直接取$z$个石子让先手进入$\bigoplus_{i=1}^{n}a_i=0$的状态, 若$z=y$, 显然$j=n$, 此时后手应从$a_n$取$y-x$个石子, 让先手陷入不能从第$n$堆取$x$个石子的困境

若此时$\bigoplus_{i=1}^{n}a_i=0$, 假设此时后手从a_n中取x个石子, 根据nim博弈的性质以及$a_{n-1}<x<y$, 先手必定有应对方案, 此时后手应选择先手的应对方案, 让先手陷入不能从第n堆石子取x个石子的困境

综上先手所能得到的局面, 必定是$\bigoplus_{i=1}^{n}a_i=0$或者$\bigoplus_{i=1}^{n}a_i\bigoplus a_n\bigoplus(a_n-x)=0$, 后手必胜

②$a_{n-2}<x$, $a_{n-1}\ge x$, $a_n\ge x$

先手必然不会取石子使得$a_{n-1}<x$或$a_n<x$, 后手必然也不会取, 最后只剩下$a_{n-1}=x$和$a_n=x$两堆石子, 此时如果是先手行动, 后手进行对称操作, x后手必胜, 此时如果是后手行动, 后手直接取掉x个石子, 此时只剩下$a_n=x$一堆石子, 先手无法一次取完, 后手必胜

③$a_{n-2}\ge x$, $a_n-1\ge x$, $a_n\ge x$

后手只要保持取完石子之后还是情况③, 让先手自己进入情况②, 先手必然不会主动进入情况②, 最终会剩下三堆$a_i=x$的石子, 后手只需取掉其中某一堆, 之后进行对称操作即可, 后手必胜

综上当先手能通过一次操作使$\max_{i=1}^{n}a_i<x$且先$\bigoplus_{i=1}^{n}a_i\bigoplus a_n\bigoplus(a_n-z)=0$手胜，否则先手败

3.当$x>y$时

如果$a_n<y$, 此时题目是nim博弈, 所以只讨论$a_n\ge y$的情况

此时情况对应$x<y$时讨论的①②③情况, 先手必胜

1012

注意到有

$ans[i] = [\sum_{i=1}^{n}\sum_{j=1}^{r}S_{ij} == i]$

$= [\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{r}A_{ik}B_{kj} == i]$

$= [\sum_{k=1}^{r}(\sum_{i=1}^{n}A_{ik})\cdot(\sum_{j=1}^{n}B_{kj}) == i]$

记$F[q]$为$(\sum_{i=1}^{n}A_{ik}) = q$的方案数

记$G[q]$为$(\sum_{j=1}^{n}B_{kj}) = q$的方案数

容易发现$F[q] = G[q]$

考虑如何计算$F[x]$的方案数，我们有生成函数

$$g(x) = (1 + x + x^2 + \ldots + x^m)^n$$

则

$$F[q] = [x^q]g(x)$$

$g(x)$的计算方法有两种

- 发现$q = 0, 1, .. m$，并且生成函数的最高项也为m,则$g(x)$展开后的第q项$(q = 0, 1, 2, .., m)$的系数等价于$x_1 + x_2 + \ldots + xn = q$的非负整数解的个数，这是一个经典问题可以使用隔板法解决,最后的结果为$[x^q]g(x) = \binom{n+q-1}{n-1}$
- 直接使用多项式快速幂即求ln后再求exp

记$H[i]$为$(\sum_{i=1}^{n}A_{ik})\cdot(\sum_{j=1}^{n}B_{kj}) = i$ 的方案数

则有$H[x * y] += F[x] \times G[y]$ 可以通过枚举x, y得到$H[i]$, $i = 0, 1, 2, \ldots, m$的值

注意到$H[0] += F[0] \times G[y], y = m + 1, m + 2, \ldots, m * n$, 通过上式我们没有计算类似这样的值.

考虑到$H[0] = F[0] \times \sum_{i=0}^{n*m}G[i] + G[0] * \sum_{i=0}^{n*m}F[i] - F[0] * G[0]$

又$\sum_{i=0}^{n*m}G[i] = \sum_{i=0}^{n*m}F[i] = (m+1)^n$. $F[0] = G[0] = 1$

故修正$H[0] = (m+1)^n - 1$

记生成函数

$h[x] = (h[0] + h[1]x + h[2]x^2 + \ldots + h[m]x^m)^r$

则$ans[i] = [x^i]h(x)$

我们可以使用多项式快速幂求出$h(x)$, 注意到r很大,我们可以用欧拉定理处理

时间复杂度为$O(mlogm)$

1013

题目的意思等价于我们对n个点操作完之后使得树的直径最短。

考虑二分答案，我们每次把二分的答案作为一个阈值来限制树形DP的转移。

对于一个以 u为根的子树，我们维护两个状态$dp[u][0]$代表u这个节点的power值给儿子的情况下,u子树所有叶子节点到u的最长链,$dp[u][1]$代表u这个节点的power值给父亲的情况下,u子树所有叶子节点到u的最长链,

对于$dp[u][1]$转移,我们只需要在满足经过u节点的任意两个儿子都满足链长不大于阈值的情况下记录最长链即可,对于$dp[u][0]$转移我们只需要枚举把u这个点的power贡献给最长链和次长链下所能得到的满足任意两个儿子的链长之和不大于阈值的情况下最短的最长链.最后只需要查看一下根节点的dp值是否合法即可

时间复杂度$O(nlog(n*max_w))$