A 拼图

e_i 的答案等价于，给定一个由 e_i 个珠子构成的环，用红绿蓝三种颜色染色；要求绿色与蓝色不能染相邻的珠子，且旋转、翻转同构。

求所有 n 个 e_i 元环的染色方案数之和。

由 Burnside 引理，答案 $res=\dfrac{1}{|G|}\sum\limits_{g\in G}|X^g|$ 。

> G 为所有操作构成的群，$|G|$ 为它的阶。
>
> g 为其中的一种操作（一种置换）。
>
> X 指映射组成的集合，即每条边映射到"红绿蓝"构成的集合。
>
> $|X^g|$ 指不动点数（即进行操作 g 以后，和原状态一致的映射数）。

可以证明，对于 m 元环，旋转、翻转同构的操作构成的群 G ，仅包含 m 个旋转操作和 m 个翻转操作。其中，旋转操作的角度分别为 $0\times\frac{2\pi}{m},1\times\frac{2\pi}{m},2\times\frac{2\pi}{m},\cdots,(m-1)\times\frac{2\pi}{m}$；对称操作的对称轴角度分别为 $0\times\frac{\pi}{m},1\times\frac{\pi}{m},2\times\frac{\pi}{m},\cdots,(m-1)\times\frac{\pi}{m}$ 。

不妨假设环是半径为 ρ 的圆形的，m 颗珠子均匀分布在圆周上。以环的圆心为极点，圆心向某一颗珠子的方向为极径的正方向，建立极坐标系。将极坐标为 $(\rho,0\times\frac{2\pi}{m}),(\rho,1\times\frac{2\pi}{m}),(\rho,2\times\frac{2\pi}{m}),\cdots,(\rho,(m-1)\times\frac{2\pi}{m})$ 的珠子分别记为 0 号到 $(m-1)$ 号珠子。

对于操作旋转 $d\times\frac{2\pi}{m}$ 个弧度，则第 i 号珠子旋转到了第 $(i+d)\bmod m$ 号珠子的位置。则该操作的不动点需要满足第 i 号珠子和第 $(i+d)\bmod m$ 号珠子同色。

故对于 i ，有 $i,(i+d)\bmod m,(i+2d)\bmod m,\cdots$ 这些珠子必须同色。而 $(i+\frac{m}{\gcd(m,d)}d)\bmod m=i$ ，故这个等价类有 $\frac{m}{\gcd(m,d)}$ 个元素。所以共有 $\dfrac{m}{(\frac{m}{\gcd(m,d)})}=\gcd(m,d)$ 个等价类。

考虑对这 $\gcd(m,d)$ 个等价类进行染色的方案数，等价于对一个 $\gcd(m,d)$ 元环染色，方案旋转、翻转不同构的方案数；又等价于对一个长度为 $\gcd(m,d)+1$ 的链染色，头尾颜色相同的方案数。

记 f_{n,c_1,c_2} 表示给一个长度为 $n+1$ 的链染色，且第一个颜色为 c_1 ，且最后一个颜色为 c_2 的方案数。

很显然可以列出矩阵的转移形式：

$$\begin{pmatrix}f_{n,r,r}&f_{n,g,r}&f_{n,b,r}\\f_{n,r,g}&f_{n,g,g}&f_{n,b,g}\\f_{n,r,b}&f_{n,g,b}&f_{n,b,b}\end{pmatrix}=\begin{pmatrix}1&1&1\\1&0&1\\1&1&0\end{pmatrix}\begin{pmatrix}f_{n-1,r,r}&f_{n-1,g,r}&f_{n-1,b,r}\\f_{n-1,r,g}&f_{n-1,g,g}&f_{n-1,b,g}\\f_{n-1,r,b}&f_{n-1,g,b}&f_{n-1,b,b}\end{pmatrix}$$

简记为 $F_n=AF_{n-1}$ ，迭代可得 $F_n=A^nF_0=A^nE=A^n$

所以对于操作旋转 $d\times\frac{2\pi}{m}$ 个弧度，答案为 $f_{\gcd(m,d),r,r}+f_{\gcd(m,d),g,g}+f_{\gcd(m,d),b,b}$，为 $A^{\gcd(m,d)}$ 的主对角线和，记为 $f_1(\gcd(m,d))$。

当 m 为奇数时，显然每个对称操作的对称轴必过环上两相邻珠子的中心和另一珠子。该操作的不动点要求对对称轴的一侧染色，则相邻的一对珠子必须为红色。

故答案等价于对一个长度为 $\frac{m+1}{2}$ 的链染色，要求头为红色的方案数，即为 $A^{\frac{m-1}{2}}$ 的第一列和，记为 $f_2(\frac{m-1}{2})$ 。

当 m 为偶数时，有 $\frac{m}{2}$ 个对称轴过环上两相对的珠子；有 $\frac{m}{2}$ 个对称轴过两对相邻珠子的中心。

对于前者，等价于对一个长度为 $\frac{m}{2}$ 的链染色，答案为 $A^{\frac{m}{2}-1}$ 的所有元素和，记为 $f_3(\frac{m}{2}-1)$；对于后者，由于相邻珠子必须为红色，故等价于对一个长度为 $\frac{m}{2}$ 的链染色，且要求头尾均为红色，答案为 $A^{\frac{m}{2}-1}$ 的第一行第一个元素，记为 $f_4(\frac{m}{2}-1)$ 。

总结得到：

$$res=\begin{cases}\dfrac{1}{2m}\left(\displaystyle\sum_{i=0}^{m-1}f_1(\gcd(m,i))+m\cdot f_2(\frac{m-1}{2})\right) & ,m\text{ 是奇数}\\ \dfrac{1}{2m}\left(\displaystyle\sum_{i=0}^{m-1}f_1(\gcd(m,i))+\frac{m}{2}\cdot f_3(\frac{m}{2}-1)+\frac{m}{2}\cdot f_4(\frac{m}{2}-1)\right) & ,m\text{ 是偶数}\end{cases}$$

很显然，$f_2(n),f_3(n),f_4(n)$ 均可递推得到，$\frac{1}{2m}$ 的贡献可以最后计算。瓶颈在于如何快速计算 $\displaystyle\sum_{i=0}^{m-1}f_1(\gcd(m,i))$ 。

对该式由莫比乌斯反演得到 $\displaystyle\sum_{d|m}f_1(d)\varphi(\frac{m}{d})$ 。

先通过递推得出 $f_1(1)$ 到 $f_1(M)$ ，以及采用欧拉筛得到 $[1,M]$ 范围内的欧拉函数值；通过枚举倍数的方式可以计算出答案，复杂度为 $T(M)=O(3^3M)+O(M)+\displaystyle\sum_{i=1}^{M}\frac{M}{i}=O(M\log M)$ 。

B FFT

根据题目对于升高的定义，一个长度为 n 的排列，升高最多出现在排列 $1, 2, 3, \cdots, n$ ，共有 n 个；而升高最少出现在排列 $n, n-1, n-2, \cdots, 1$ ，共有 0 个。

因此式子 $\sum\limits_{k=0}^{n} \left\langle {n \atop i} \right\rangle$ 的组合意义为，求一个长度为 n 的排列，升高次数介于 $[0, n]$ 的排列个数。

由于长度为 n 的排列不可能升高次数小于 0 或大于 n ，故所有长度为 n 的排列都是满足题意的，共有 $n!$ 个排列。

直接输出 $n! \bmod 998244353$ 即可。

C 魔法师

先考虑这道题只算前缀时的做法。

使用给定的字符串集建出一个Trie树，Trie树上的一个节点即代表字符串的一个前缀，考虑某个前缀对答案的贡献。假设有一个长度为 x 的前缀 $str(x)$ ，有 k 个未被使用的串以它为前缀，则它对答案有 $\lfloor \frac{k}{2} \rfloor \times x^2$ 的贡献。在字典树上统计串的出现次数，dfs整个字典树,自底向上统计，即可维护出答案。

但题目所求贡献为最长公共前缀和最长公共后缀的较小值，对此要将原串改造一下，用新串的奇数位记录原串，用新串的偶数位记录原串的reverse。

假设两个串S,T最长公共前缀长度为L_1，最长公共后缀长度为L_2。不妨假设$L_1 \le L_2$,则$S_{L_1+1} \ne T_{L_1+1}$。于是，进行上述操作后，新得到的串S',T'满足$S'_i = T'_i$对于$1 \le i \le L_1 + L_1$均满足，且$S'_{2L_1+1} \ne T'_{2L_1+1}$。若新串最长公共前缀长度为$x$，则答案为$\lfloor \frac{x}{2} \rfloor = L_1 = \min(L_1, L_2)$。同理，当$L_1 > L_2$时，答案也为$\lfloor \frac{x}{2} \rfloor$。

执行上述算法，统计答案时将长度除 2 即可得到答案。

设 $|\Sigma|$ 为字符集大小，则时间复杂度为$O(\sum |S| \cdot |\Sigma|)$。

空间复杂度也为$O(\sum |S| \cdot |\Sigma|)$。

D 剪纸

满足 $l+r$ 的和最小的纸肯定是尽量每剪一刀都是对答案有贡献的，并且令没贡献的纸最小。

所以反过来思考，就是从一张 1×1 的纸开始进行扩展，直到扩展不了。

假设当前扩展到一个长为 x ，宽为 y 的纸，表示为 (x,y) ，接下来有两种可能的扩展： $(x+y,y)$ 或 $(x+y,x)$ 。

如果扩展到的是 $(x+y,y)$ ，那这一次的扩展是毫无意义的，因为在 (x,y) 时裁剪的也是边长为 y 的正方形。

所以只会扩展到 $(x+y,x)$ 。

我们发现其实这就是斐波那契数列，所以暴力算一下就可以了。

注意对于 $n=2$ 的情况需要特判，因为不论 1×1 还是 1×2 的纸，都是只能剪出边长为 1 的正方形。

时间复杂度 $O(\log n)$ 。

E 黑白大陆

给定一个黑白染色的矩阵，每次操作可以改变一个同色的四联通块的颜色，问最少多少次操作可以把整个矩阵变为白色。

建图，考虑将同色的极大联通块缩点，在相邻的不同色联通块之间连边。每次操作一个点，即可使其和相邻的点，变为同一颜色，相当于在图上走一条边，将两个联通块 u, v 变为同色的最少次数即为 u 和 v 在图上的最短距离。

对于一个联通块，操作 1 次后，相邻的联通块都会变成与其同色，操作 k 次后，距离为 k 的联通块都会变为与 i 同色。对于联通块 i，假设 i 与其他联通块的最短路的最大值为 x，则操作 x 次后整个棋盘变为与 i 同色。

最终的答案即为所有点出发的最远点距离的最小值。

注意当最长的最短路起点和终点都为黑色时，需要多操作一次，因为最终目标是使整个矩阵变为白色。

时间复杂度为$O(n^2m^2)$。

F 望舒客栈的每日委托

考虑到两种人的落座方式，社交牛逼症会选择第一张剩余可坐人数$\geq x_i$的四人座，社交恐惧症会选择第一张剩余可坐人数$= 4$的四人座，用4个set维护剩余可坐人数分别为1、2、3、4的四人座，按照时间顺序，每当有人落座是，便修改对应的set，时间复杂度$O(nlogn)$。

G Rock&Frog

$Hint$: 题目所给条件等价于 $\forall 1 \le i,\ j \le n,\ (a_i - a_j)(b_i - b_j) \ge 0$

利用动态规划解决问题。

设 $f(i)$ 为青蛙从位置 1 跳到位置 i 的最小代价，则
$f(i) = \max\{f(j) + a_j \times c_i^2 + b_j \times c_i\},\ 1 \le j < i$。

将 $f(j) + a_j \times c_i^2 + b_j \times c_i$ 看成关于 c_i 的一元二次函数，则每次相当于在 $i-1$ 个函数里对在 $x = c_i$ 位置的值取最小值。

根据求根公式可得，题目所给条件可推出所有一元二次函数在 x 正半轴上两两最多只有一个交点。

利用李超树即可动态维护这些一元二次函数的最低点。

李超树维护一元二次函数具体方法与维护直线时完全相同，只需将涉及直线的部分替换为一元二次函数即可。

时间复杂度为 $O(n \log(\max c))$ 。

空间复杂度为 $O(n \log(\max c))$ 。

H 梅花易数

根据题意模拟即可。

将输入时的 12 种地支串转化为 $1, 2, 3, \cdots, 12$ 的对应数字。

而观察三爻卦象的排位与其阴阳爻的关系。不难发现对于一个除以 8 的余数 i，$(i-1)$ 的二进制最后一位为 $0/1$ 分别对应了卦象最上一爻是阴爻还是阳爻；同理，$(i-1)$ 的二进制倒数第二位与倒数第三位，分别对应卦象的中间爻和最下爻分别是阴爻还是阳爻。

因此，我们不妨用一个 $[0, 2^6)$ 内的二进制数表示一个六爻的卦象。其低三位为 $y+m+d-1 \bmod 8$，高三位为 $y+m+d+h-1 \bmod 8$。输出时从高往低遍历，遇到二进制位为 0 时输出"短线-短线-短线"，遇到为 1 时输出"短线-空格-短线"。

而对于变卦，其变化的位置为该二进制数的从低向高第 $6-(y+m+d+h) \bmod 6$ 位，直接将结果异或上 $2^{6-(y+m+d+h) \bmod 6}$ 即可。同样采用上述方法输出，即可得到正确答案。

I Rock&String

要求等价于：给一个子串，求它在原串中两次出现之间的最小间距。

做法很多，std 的做法是：先把原串的 SAM 跑出来，用线段树合并在 $fail$ 树上自下而上地维护 $right$ 集合。

合并的同时维护线段树上区间最左点到区间左端的距离 $ldis$，节点区间最右点到区间右端的距离 $rdis$ ，节点区间相邻两点间的最小值 ans。

询问的时候利用树上倍增在 $fail$ 树上定位节点，直接输出答案即可。

字符集大小为 $|\Sigma|$ ，则时间复杂度为 $O(n\log n+n|\Sigma|)$ 。

空间复杂度也为 $O(n\log n+n|\Sigma|)$ 。

J 新英雄

如果第 1 个是敌方小兵就无解，否则在第 0 个和第 1 个小兵之间来回“踏”就能积攒法力。

对于每一段小兵，先假设全用“踏”，遇到敌方小兵就用“斩”。“斩”的法力不够就来回“踏”攒一下法力（有解就一定能攒法力）。到终点后，对于剩余法力，有一半（下取整）的法力可以用于在己方小兵上“斩”。把一个“踏”改成“斩”，答案就会减二；原来是获得一点法力也会变成消耗一点法力，所以法力也是减二。

观察可以发现，这样做是最优的。额外去攒法力或浪费法力都不会更优。

注意取整问题。时间复杂度除了排序的 $O(n \log n)$ 外，是线性的，空间也是线性的。

K 斐波那契

做法一

这里有个很经典的问题，给定一张有向图，求恰好包含k条边的路径的权值和，路径的权值定义为k条边的边权乘积，由离散数学和线性代数的知识可知，我们可以构造图的邻接矩阵A,$A^k[i][j]$为起点为i，终点为j的恰好包含k条边的路径的权值和，即为所求。

这里我们考虑$fib(i)$的信息可以用第i个斐波那契矩阵F^i维护，$F = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$。

$fib(i+j)$的信息等价于$F^{i+j} = F^i * F^j$。等价于矩阵乘法。

$fib(\sum_{i=1}^{k} w_{(v_{i-1},v_i)})$的信息等价于$\prod_{i=1}^{k} F^{w_{(v_{i-1},v_i)}}$。等价于矩阵乘法。

同上述经典做法，我们把邻接矩阵的每个元素替换成一个$2*2$大小的矩阵，这样可以通过矩阵维护斐波那契数列的信息，做分块矩阵乘法

对于边权为i的边的信息用第i个斐波那契矩阵F^i维护。

我们可以维护一个大小为$2n \times 2n$分块矩阵$A = \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \dots & \dots & \dots & \dots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{bmatrix}$

对于任意的$A_{i,j} = \sum_{(u_k=i,v_k=j) \in E, 1 \le k \le m} F^{w_{(u_k,v_k)}}$可以表示成若干个斐波那契矩阵之和。

其中$w_{(u_k,v_k)}$表示$u_k = i, v_k = j$之间的一条边的权值。

最后的答案显然是A^k。

求解i,j的答案，即为$A^k[2*i-1][2*j]$。

分块矩阵乘法和一般矩阵乘法求出的结果是一样的，采用普通矩阵乘法即可。

对于F^p和A^k显然可以用矩阵快速幂求解。

总复杂度为$O((2n)^3 logk + 8mlogp)$。

做法二

斐波那契的第n项$fib(n) = \frac{1}{\sqrt{5}}((\frac{1+\sqrt{5}}{2})^n - (\frac{1-\sqrt{5}}{2})^n)$，显然是个整数。

同上述经典做法，我们把邻接矩阵的每个元素替换成一个$a + b\sqrt{5}$的类，这样可以通过$Q(\sqrt{5})$维护斐波那契数列的信息。

所以$(\frac{1+\sqrt{5}}{2})^n$和$(\frac{1-\sqrt{5}}{2})^n$在$Q(\sqrt{5})$上共轭的，对于$a + b\sqrt{5}$中，两数的a相同，b相反。

因此我们采用扩域的方式，每条边用$(\frac{1+\sqrt{5}}{2})^n$表示，矩阵每个元素上维护$a + b\sqrt{5}$。

A^k取出$A^k_{i,j}$的$\sqrt{5}$部分系数乘以2即为答案。

总复杂度为$O(4n^3 logk + 4mlogp)$。

L 启航者

做法1：树形dp

考虑一条路径的走向，一定是从先某一个点开始一直向上走，直到走到一个“拐点”，接下来一直向下走。

所以对于每个点，我们维护三种值，在维护最大值的同时维护一下起点：

$f[x][0]$：由下至上最大的的和，**并且还能接着向上**；

$f[x][1]$：由上至下的和（只有唯一一种走法）；

$f[x][2]$：作为拐点最大的和。

设当前节点为 now，初始化 $f[now][0] = f[now][1] = f[now][2] = a[now]$。

定义一个函数 $getMax(now, from)$ 返回值为与 now 相邻的节点除了 $from$ 之外的最大值的点，也就是知道上一个从哪里来，想要知道往哪里去。

如果 $nxt = getMax(now, fa[now]) \neq 0$，则 $f[now][1].sum+ = f[nxt][1].sum$；

如果 $getMax(now, 0) \neq fa[now]$，则 $f[now][0]$ 作为起点不能继续向上了。

枚举 now 的儿子，设为 $from$，计算从 $from$ 走到 now 的值：

令 $nxt = getMax(now, from)$；

如果 $nxt \neq fa[now]$，代表这个点为拐点，则 $f[now][2] = max(f[now][2], f[from][0] + a[now] + f[nxt][1])$；

如果 $nxt = fa[now]$，代表这个点还可以继续向上走，则 $f[now][0] = max(f[now][0], f[from][0] + a[now])$；

最后答案为每个点 $max(f[x][0], f[x][1], f[x][2])$ 的最大值。

时间复杂度为 $O(n)$。

做法2：记忆化

记一下每个点向最大/次大走的答案。

dfs时如果是从最大的走进来，那下一个就只能向次大走，否则下一个向最大走。

因为记忆化了一下，所以每个点只会被计算一遍，时间复杂度也是 $O(n)$。

M 拉格朗日插值

容易得到取最大值时，$\forall i \to x_i \neq 0$ 。

因此由拉格朗日乘子法，记函数：

$$L(x_1, x_2, \cdots, x_k, \lambda) = \lambda(\frac{x_1^2}{a_1^2} + \frac{x_2^2}{a_2^2} + \cdots + \frac{x_k^2}{a_k^2} - 1) + \prod_{i=1}^{k} x_i$$

对函数求偏导得到：

$$\begin{cases} \dfrac{\partial L}{\partial \lambda} = \dfrac{x_1^2}{a_1^2} + \dfrac{x_2^2}{a_2^2} + \cdots + \dfrac{x_k^2}{a_k^2} - 1 \\ \dfrac{\partial L}{\partial x_i} = \dfrac{2\lambda}{a_i^2} x_i + \prod_{j \neq i} x_j & (i = 1, 2, \cdots, k) \end{cases}$$

令偏导数均为 0 得到：

$$-\lambda = \frac{a_i^2}{2x_i^2} \cdot \prod_{j=1}^{k} x_j \qquad (i = 1, 2, 3, \cdots, k)$$

$$\frac{a_i^2}{2x_i^2} \cdot \prod_{j=1}^{k} x_j = \frac{a_p^2}{2x_p^2} \cdot \prod_{j=1}^{k} x_j \qquad (i, p = 1, 2, 3, \cdots, k)$$

$$\frac{x_i}{a_i} = \frac{x_p}{x_p}$$

再由 $\sum_{i=1}^{k} (\frac{x_i}{a_i})^2 = 1$ 得到 $\frac{x_i}{a_i} = \pm\sqrt{\frac{1}{k}}, (i = 1, 2, 3 \cdots, k)$

因此有 $\prod_{i=1}^{k} x_i = \prod_{i=1}^{k} \pm\sqrt{\frac{1}{k}} a_i = k^{-\frac{k}{2}} \prod_{i=1}^{k} a_i$ （由于 k 是偶数，因此正负号运算后一定为正号）。

或根据均值不等式，平方平均数大于等于几何平均数，有：

$$\sqrt{\frac{1}{k}(\sum_{i=1}^{k} \frac{x_i^2}{a_i^2})} \geq \sqrt[k]{\prod_{i=1}^{k} \frac{x_i}{a_i}}$$

$$\sqrt{\frac{1}{k}} \geq (\prod_{i=1}^{k} x_i)^{\frac{1}{k}} \cdot (\prod_{i=1}^{k} a_i)^{-\frac{1}{k}}$$

$$\prod_{i=1}^{k} x_i \leq k^{-\frac{k}{2}} \cdot \prod_{i=1}^{k} a_i$$

当且仅当 $\frac{x_1}{a_1} = \frac{x_2}{a_2} = \cdots = \frac{x_k}{a_k}$ 时取得。

综上，$\prod_{i=1}^{k} x_i$ 的最大值为 $k^{-\frac{k}{2}} \cdot \prod_{i=1}^{k} a_i$ 。而 $k^{-\frac{k}{2}}$ 由于保证了 k 是偶数，因此可以由快速幂求解。

现在的问题是求解从 $b_1, b_2, b_3, \cdots, b_m$ 中任选 k 个数，他们乘积的期望是多少。

构造生成函数 $F(x)=\prod_{i=1}^{m}(1+b_ix)$，则其 x^k 系数 $[x^k]F(x)=\sum_{c_1+c_2+\cdots+c_m=k}\prod_{i=1}^{m}b_i^{c_i}$ ，即 $b_1,b_2,\cdots,b_m$ 中恰好挑选 k 个数的乘积在所有方案中的求和。

由此我们得知，从 $b_1,b_2,b_3,\cdots,b_m$ 中任选 k 个数的所有方案，他们乘积的求和即为 $[x^k]F(x)$ 。而方案的总数为 $\binom{m}{k}$ ，故任取 k 个数的乘积期望为 $\frac{[x^k]F(x)}{\binom{m}{k}}$ 。

$\binom{m}{k}$ 可以通过求阶乘与阶乘逆元的方式求得，$[x^k]F(x)$ 可以通过分治 FFT 或启发式合并 FFT 得到。

求解 $k^{-\frac{k}{2}},\binom{m}{k},[x^k]F(x)$ 的时间复杂度分别为 $O(\log n),O(n)+O(\log n),O(n\log^2 n)$ ，故总复杂度为 $O(n\log^2 n)$ 。