A.克苏鲁的呼唤

贪心

把n种算法按s_i值排序后从小往大选即可

时间复杂度$O(nlog_2n)$

B.魔道中人

组合数学

从总方案数中减去不合法方案数即为答案

首先考虑满足$aa, bb, cc \geq 0, aa+bb+cc \leq d$的三元组$(aa, bb, cc)$的个数

令$dd = d-aa-bb-cc$，则问题转化为求满足$aa, bb, cc, dd \geq 0, aa+bb+cc+dd = d$的四元组$(aa, bb, cc, dd)$的个数，由插板法知方案数为$C_{d+3}^3$

然后考虑所有三元组(aa, bb, cc)中不能使$a+aa, b+bb, c+cc$构成三角形三边的方案数，即：

1.$a+aa \geq b+bb+c+cc$

2.$b+bb \geq a+aa+c+cc$

3.$c+cc \geq a+aa+b+bb$

三种情况类似，以求第一种情况的方案数为例，该种情况等价于$bb+cc \leq a-b-c+aa$，枚举aa，则问题转化为求满足$bb, cc \geq 0, bb+cc \leq min(a-b-c+aa, d-aa)$的二元组$(bb, cc)$的个数，同理可得方案数为$C_{t+2}^2$，其中$t = min(a-b-c+aa, d-aa)$

时间复杂度$O(d)$

C.手掌模拟器

博弈论

由于每次操作后小正方体数目一定减少，故每个状态必分胜败，没有平局

若$a=b=c=1$，即只有一个小正方体，先手只能选择该正方体为A，后手也只能选择该正方体，进而后手操作结束后所有正方体被拿完，先手胜

若$a \cdot b \cdot c > 1$，将所有正方体分成两种：A和非A，先手确定一个角为A后，如果所有非A块都是必败态，则后手取走A使得先手下一轮必然选择一个必败态进而失败，此时后手必胜；如果存在非A块是必胜态，则后手取该非A块进而胜利

D.峰峰学长的碎碎念

树链剖分+线段树+在线倍增LCA

求出树上每点的dfs序后，任意一棵子树的dfs序是一段连续的区间，子树查询操作等价于区间求和

树链剖分后一条树上路径转化为至多log_2n段dfs序连续的区间，路径更新操作等价于对若干区间的更新操作

由于该树上路径是有向路径，更新操作分两种：

1.从下往上，即从dfs序为R的点到dfs序为L的点依次进行加$C_x^3, C_{x+1}^3, \ldots, C_{x+R-L}^3$的操作，其中$R \geq L$

2.从上往下，即从dfs序为L的点到dfs序为R的点依次进行加$C_x^3, C_{x+1}^3, \ldots, C_{x+R-L}^3$的操作，其中$R \geq L$

注意到对于一个dfs序为pos的点可能要进行若干次更新操作，每次更新操作给该点权值加上C_{val}^3，而每次的更新值val可能不同，考虑把每次的更新值转化为一个依赖于pos的值便于操作，对于两种更新操作有：

1.更新值与dfs序的和为定值，均为$x+R$，令$d=x+R$，则对于dfs序为$pos \in [L,R]$的点加的值即为C_{d-pos}^3

2.更新值与dfs序的差为定值，均为$x-L$，令$d=x-L$，则对于dfs序为$pos \in [L,R]$的点加的值即为C_{d+pos}^3

故每次操作对于每个点的更新值都是一个关于该点dfs序的三次多项式，且对于每次操作的区间中的每一点，该三次多项式的系数固定，与这些点的dfs序无关，这样一来每次更新就不用对于每个点单独维护修改值，只需要维护每个点更新值关于该点dfs序的三次多项式的四个系数，这样更新操作就变成区间的系数更新

用线段树维护四个系数的更新以及每个点权值之和，那么查询操作即为线段树的区间查询操作，更新操作即为线段树的区间更新操作，且由于$\sum_{i=1}^{x} i = \frac{x(x+1)}{2}, \sum_{i=1}^{x} i^2 = \frac{x(x+1)(2x+1)}{6}, \sum_{i=1}^{x} = \frac{x^2(x+1)^2}{4}$，故对区间$[L,R]$加上一个三次多项式$p(x)=c_0+c_1x+c_2x^2+c_3x^3$的操作变成对区间$[L,R]$的四个系数值分别加上$c_0, c_1, c_2, c_3$的操作，以及对区间$[L,R]$的权值和$sum$加上

$$c_0 \cdot (R-L+1) + c_1 \cdot (\frac{R(R+1)}{2} - \frac{L(L-1)}{2}) + c_2 \cdot (\frac{R(R+1)(2R+1)}{6} - \frac{L(L-1)(2L-1)}{6}) + c_3 \cdot (\frac{R^2(R+1)^2}{4} - \frac{L^2(L-1)^2}{4})$$

在求路径长度时需要知道两个点的LCA，而每次对路径的前两个点的更新值都是1，不能按照前面的分析来更新，此时只需加一个单点更新即可，这两步在树上路径的移动会需要求一个点的某级祖先，这两步操作可以通过在线倍增LCA来解决

时间复杂度$O(nlog_2^2n)$

E.老哥，稳

数学+dp

1.考虑这种特殊排列的性质：

证明一个结论：1必然自己构成一个循环或在循环(21)里

如果$p_1 \neq 1$，考虑1所在循环的最大值x，由条件，$p_1 = x$且x作为1所在循环的最大值，在所有循环的最大值里是最小的，由于1所在循环经过三步操作后在排列最前端，故$p_1, p_2, \ldots, p_x$都应该在1所在循环里，进而得到循环长度为x且$p_x = 1$，即1所在循环长度为2，且由前面的分析轻易得到$p_2 = 1$，即$x=2$，结论成立

故特殊排列必为若干不动点和若干相邻点的对换

2.考虑长度为n的特殊排列的个数$f(n)$

若$p_1 = 1$，问题转化为长度为$n-1$的特殊序列个数；若$p_1 = 2, p_2 = 1$，问题转化为长度为$n-2$的特殊序列个数。进而有转移$f(n) = f(n-1) + f(n-2)$，其中$f(1) = 1, f(2) = 2$，即为斐波那契数列

3.考虑原问题

从前到后逐个求出答案排列的值，满足$p_1=1$的特殊排列有$f(n-1)$个，如果$k\le f(n-1)$，说明答案排列第一位是1，问题转化为求长度为$n-1$的特殊排列中的第k个排列；如果$k>f(n-1)$，说明答案排列的前两位是2 1，问题转化为求长度为$n-2$的特殊排列中的第$k-f(n-1)$个排列。以此类推即可确定答案

时间复杂度$O(Tn)$

F.图灵测试

机智

一方面，由于S只能变成$aSbS$或$bSaS$或空串，故每次操作后a,b的数量要么都不变要么都加一，故操作结束后a,b的数量必然相同；

另一方面，若a,b数量相同，用数学归纳法证明：a,b数量均为n时必然合法

$n=1$时，ab或ba显然合法

假设$k<n$时结论成立，下证$k=n$时结论也成立

若第一个字符和最后一个字符不同，即$s_1=a,s_{2n}=b$或$s_1=b,s_{2n}=a$，那么s_2~s_{2n-1}是一个a,b数量均为$n-1$的字符串，由假设必然可以由一个S生成，故原串等价于aSb或bSa，这两种情况显然合法

若第一个字符串和最后一个字符串相同，即$s_1=s_{2n}$，不妨设$s_1=s_{2n}=a$，那么由于整个串a,b数量相同，必然存在一个位置x使得$s_x=b$，且$s_1,\dots,s_x$这x个字符中a,b数量一致，且由于$s_{2n}=a$，故$x\le 2n-2$，那么整个字符串被分成四部分：$s_1=a$，s_2~s_{x-1}，$s_x=b$，s_{x+1}~s_{2n}，显然第二部分和第四部分为两个a,b数量相同且小于n的字符串，由假设知这两部分可以由S生成，故原串等价于$aSbS$，显然合法

由数学归纳法，a,b数量相同的字符串合法

进而一个串合法等价于该串a,b数量相同

时间复杂度$O(|s|)$

G.入门数学题I

单调栈

考虑找出所有满足$index(z)<index(s)<index(q)$且$z<q<s$的三元组，维护和的最小值即为答案

注意到z,s相当于是原序列某个递增子序列的两端，而q相当于作为编号最大的元素新加入该递增子序列，由于$q<s$，为求得以q为末端的递增子序列，需要把s从该递增子序列中删除，该过程即为单调栈维护原序列的一个严格递增列时，新加入元素不大于栈顶元素时的退栈操作，更准确的说，假设栈内元素为$s_1,\dots,s_p$，新加入元素为a_i，若$a_i>s_p$则a_i进栈，否则一直退栈到栈顶元素严格小于a_i然后a_i进栈，在这个退栈过程中把栈底元素s_1看作z，把要退栈的栈顶元素s_p看作s，把准备入栈的元素a_i看作q，只要这三者满足$z<q<s$则为一个合法的三元组，用他们的和更新最小值即可

时间复杂度$O(n)$

H.入门数学题II

字符串最小表示法，模版题

时间复杂度$O(|s|)$

I.P5天下第一

最小费用最大流

首先不考虑每行1的数量占整体1的数量的比例，枚举每行1的数量最大值x，考虑先把所有的0变成1，那么问题等价于在保证每行每列的1的数量相同且不超过x的前提下，不变成1的0的数量最少

考虑两排点，第一排第i个点为第i行，第二排第i个点为第i列，设$row[i]$为第i行0和1的数量，$col[i]$为第i列0和1的数量，建图如下：

1.源点向第i行连容量为$row[i]$，花费为0的边；

2.第i列向汇点连容量为$col[i]$，花费为0的边；

3.第i行向第i列连容量为x，花费为0的边

4.若$A[i][j] =' 0'$，则从第i行向第j列连容量为1，花费为1的边

考虑建图的合理性，第i行有容量为$row[i]$的流，这些流分两部分流出，一部分通过第i行到第i列的容量为x的边流，表示这些$0, 1$全部变成1，体现出第i行的1的数量，另一部分通过若干花费为1的边流到其他列，表示这些0不变成1；而对于第i列，流向第i列的流分为两部分，一部分流的花费为1的表示有若干第i列的0不变成1，依旧作为第i列的0，另一部分的流花费为0则表示某行的$0, 1$变成了第i列的1，且花费为0的流只能通过第i行到第i列的边流入，那么整个网络满流保证了流向第i列的花费为1的流就是第i列的1的数量，进而第i行第i列的1的数量相同且不超过x，而最小费用保证了满足之前条件的情况下，不变成1的0的数量最少，也即1的数量最多

假设满流的最小费用为$cost$，总的$0, 1$数量为sum，那么总的1的数量即为$sum - cost$，而每行1的数量不超过x，若$\frac{x}{sum-cost} \leq \frac{A}{B}$则说明该方案满足条件，更新$sum - cost$的最大值即可

时间复杂度$O(N^4)$

J.填坑

dp

以$dp[i]$表示使得前i个位置的地面满足条件的最少花费，枚举第i个位置所处平整地面长度有转移：

$dp[i] = min\{dp[j] + cost(j+1, i), i - j \geq k\}$

其中$cost(i, j)$表示把第i个位置到第j个位置变成平整地面的最小花费，$dp[n]$即为答案，故只要求出$cost$即可

固定i，j从i到n，维护这段区间高度的最大值mx，若$b_j \leq mx$，那么把第j个位置加进去的代价为$a_j \cdot (mx - b_j)$，若$b_j > mx$，那么区间$[i, j-1]$的高度要从mx变到b_j，花费为$(b_j - mx) \cdot (s_{j-1} - s_{i-1})$，其中$s_i = \sum_{j=1}^{i} a_j$表示前$i$个位置高度增加1的花费，以此即有转移：

$cost(i, j) = cost(i, j-1) + a_j \cdot (mx - b_j), b_j \leq mx$

$cost(i, j) = cost(i, j-1) + (b_j - mx) \cdot (s_{j-1} - s_{i-1}), b_j > mx$

时间复杂度$O(n^2)$