Notes

A. Today's Word

字符串倍增模拟.

最多倍增 $\log 2m$ 次, 直接暴力模拟, 接着只需要一个快速幂判断平移多少次即可.

B. Honkai in TAIKULA

奇数长度最小环.

首先可以想到跑 scc 之后每一个强连通分量单独考虑.
对于一个强连通分量而言, 首先分为两种情况:

1. 如果内部是存在负环的, 接着又分为两种
    1. 对于点 u 而言, 存在一个长度为奇数的环, 那么答案一定是 $-\infty$,
    2. 否则答案一定是不存在, 也就是 $+\infty$.
2. 内部不存在负环就可以考虑怎么计算了, 可以想到分层图最短路, 把点 u 拆成 u 和 $u+n$, 分别表示到当前点时路径长度为奇数或偶数, 接着只需要从 $u+n$ 开始跑单源最短路, 答案就是 $dist_u$.

接着算一下时间复杂度, 跑 n 次 spfa 是不正确的, 这里就可以考虑 Johnson 全源最短路.

再从头整理一遍, 可以再建图的时候就进行分层, 接着对这 $2n$ 个点跑 scc, 如果 u 与 $u+n$ 不在一个强连通分量内, 答案自然是 $+\infty$. 接着对每一个 scc 判断有没有负环, 没有负环的 scc 跑 Johnson 即可.

E. LCM Plus GCD

k 个不同数的 $\gcd+\operatorname{lcm}$ 为 x 的方案数.

首先有 gcd 整除 lcm, 如果设 $\gcd\{x_1,x_2,\cdots,x_k\}=g$, 自然首先要满足 $g\mid x$, 可以把所有的 x_i 都除以 g, 这样得到的 $\gcd\{x'_1,x'_2,\cdots,x'_k\}=1$, 且 $\operatorname{lcm}\{x'_1,x'_2,\cdots,x'_k\}=\frac{x}{g}-1:=L$.

设 $L=\prod_{i=1}^{s}p_i^{\alpha_i}, x'=x/g=\prod_{i=1}^{s}p_i^{\beta_i}$, 那么就有 $i=1,2,\cdots,s$,

$$\max\{v_{p_i}x'_i\}=\alpha_i \text{ and } \min\{v_{p_i}x'_i\}=\beta_i,$$

这里一共 $2s$ 个限制, 可以考虑容斥, 每一个 i 对应的两个限制根据是否触屏上界或者下界分为 4 种. 对于每一种, 将 $\prod_{i=1}^{s}(r_i-l_i+1)$ 作为总方案数, 直接组合数计数即可.

F. Timaeus

期望 dp.

每次有两种选择, 一种是 p 的概率多获得一个, 另一种是 q 的概率少花费一个.
很显然可以考虑 dp, 转移式如下

$$\begin{aligned}
&dp_i->\max dp_i, dp_{i-b}+1+p, \text{ 这是选择第一种 ,}\\
&dp_i->\max dp_i, dp_{i-b}*(1-q)+dp_{i-b+1}*q+1, \text{ 这是选择第二种 ,}
\end{aligned}$$

但是根据第二个式子的表达式可以发现要保证 $b > 1$, 再单独考虑 $b = 1$ 的情形, 可以直接计算, 得到 $\max\{1 + p, 1/(1 - q) * a\}$.

H. Neil's Machine

S 串 rshift 得到 T 串.

其实就是看 $s[i] - t[i]$ 是否等于 $s[i - 1] = t[i - 1]$, 如果不相等就需要在 i 位操作一次, 否则不需要. 注意单独讨论第 0 位.

I. Elevator

签.

直接 $n - m + 1$.

J. Similarity (Easy Version)

任意两个串的最长公共子串的长度最大值.

暴力枚举两个不同串跑 dp 计算最长公共子串.

K. Similarity (Hard Version)

构造 n 个长度为 k 的串使得任意两个串的最长公共子串的长度最大值为 m.

其实更像是分类讨论.

1. $m \geqslant k$ 显然无解.
2. $k = 1$ 的时候至多 26 个, 从 a 到 z 输出即可.
3. $m = 0$ 也是至多 26 个, 从 a 到 z 输出即可.
4. $m = 1$ 的时候只要循环输出所有形如 $baba\cdots, caca\cdots, cbcb\cdots, \cdots, zyzy\cdots$ 即可, 一共 $C_{26}^2 > 300$ 个.
5. 剩下的情形就是 $2 \leqslant m < k$ 的情形, 类似上一种情形的操作, 只需要在前两个串最前面加上 m 个 a 并截断后面 m 个字符即可.

L. Architect

判断 n 个长方体是否不交且并为给定长宽高的大长方体.

可以想到用三维差分. 先添加一个反的大长方体, 接着加入 n 个长方体, $O(n \log n)$ 计算每一个关键点的三维前缀, 判断是否都为 0.

另一个做法. MAOoo 给的做法, 大长方体的八个顶点先记为 1, 此后每一个小正方体的八个顶点的值异或上 1, 最后要满足每一个点的值都是 0, 但是还要再判断体积和是否为大长方体的体积.