A

签到。（有操作次数比网格要大的情况）

示例代码

B

第一问：$g=\sum_{i=1}^{n}\sum_{j=1}^{n}i\cdot j=\sum_{i=1}^{n}i\cdot\sum_{j=1}^{n}j$，相当于两个前缀和乘积。

第二问：$f=\prod_{i=1}^{n}\prod_{j=1}^{n}i\cdot j$，考虑每个数字 x 出现的次数，当 $i=x$，内层式子$\prod_{j=1}^{n}x\cdot j$ 中 x 共出现 $n+1$ 次，当$i\neq x$ 时，每个 $\prod_{j=1}^{n}i\cdot j$ 出现 1 次 x，这样的 i 共有 $n-1$ 个，所以每个数字的出现次数都为 $2\cdot n$ 次，答案为 $(n!)^{2\cdot n}$

示例代码

C

容易发现 $F(x)=x+lowbit(x)$，所以整个森林构成的是一个树状数组，add 操作等价于树状数组常见的 log 个节点增加 w 的操作，sum 操作是一个前缀和的形式，可以想到维护一个树状数组 a，sum 操作为一个 log，考虑add，每次 add 相当于在 a 上执行 log 次单点加法：

```
for ( ; x <= n; x += x & -x)
    for (LL y = x; y <= n; y += y & -y)
        a[y] += w;
```

其中 a 是一个哈希表（map），注意到两个地方都是增加一个 $lowbit$，所以可以转化为以下代码

```
for (LL d = w; x <= n x += x & -x, d += w)
    a[x] += d;
```

这样整个 add 操作复杂度只有一个 log，使用 $unordered\_map$ 两个 log 复杂度即可通过本题，由于比赛方选择了卡常，本题需要更快的离线离散化代替 map 。

示例代码 map

示例代码 离线

D

套路题

先考虑 2 操作，函数 g 用自然语言描述即当遇到字符 A，$a=a+b$，当遇到字符 B，$b=a+b$。

注意到两个矩阵 $\begin{bmatrix}1&0\\1&1\end{bmatrix}_{(1)}$ $\begin{bmatrix}1&1\\0&1\end{bmatrix}_{(2)}$ 有

$$\begin{bmatrix}a&b\end{bmatrix}\cdot\begin{bmatrix}1&0\\1&1\end{bmatrix}=\begin{bmatrix}a+b&b\end{bmatrix}$$

$$\begin{bmatrix}a&b\end{bmatrix}\cdot\begin{bmatrix}1&1\\0&1\end{bmatrix}=\begin{bmatrix}a&a+b\end{bmatrix}$$

所以问题变为遇到字符 A，乘上第一个矩阵，遇到字符 B，乘上第二个矩阵，原序列就变成了一个矩阵序列，使用线段树维护矩阵序列的乘积，就可一个 log 的复杂度拿到区间矩阵乘积结果 T，答案即为 $\begin{bmatrix} a & b \end{bmatrix} \cdot T$。

再来考虑 1 操作将区间 $[L, R]$ 中的 A 变为 B，B 变为 A，对于每个位置来说就是交换矩 $(1)(2)$，可以在建线段树的时候把两种情况都维护起来，则该操作相当于交换一段区间的矩阵，区间修改，对应维护一个 $lazy$ 即可。

复杂度 $O(nlogn)$。

示例代码 codeforces链接

$$E$$

模拟。

示例代码

$$F$$

考虑动态规划，$dp[x]$ 表示前缀长度为 x 中最多可以分出多少份能被 3 整除的段，我们记 k 为前缀和对 3 取模的值。

$dp[x] = \max_{i=1}^{x-1} [k_x \ equal \ k_i] \cdot (dp[i] + 1)$

表达式内的 $[condition]$ 表示当 $condition$ 为真时取 1，否则取 0 的函数。维护前缀模 3 的每个 $dp[i]$ 最大值即可 $O(1)$ 转移。

示例代码