# Solutions

## A. Gudako and Ritsuka

期望难度：Medium

考虑从后往前进行博弈动态规划，在这一过程中维护所有的先手必胜区间。区间不妨采用左开右闭，方便转移。

考虑一次转移，如果当前Servant的后一个位置属于对手，则当前Servant的必胜区间可以通过将后一个Servant的每个必败区间的左端点+1、右端点+x得到；如果后一个位置属于自己，则可以通过将后一个Servant的必胜区间做同样的操作得到。不妨分别对必胜区间左右端点维护一个偏移量，需要从对手进行转移时只需修改偏移量后交换左右端点的集合，然后再在左端点的集合里插入一个0即可。

需要注意的是，这样得到的必胜区间会有重叠，可能导致对下一个对手的必胜区间的统计出错。考虑到每次转移时所有的同类区间的长度的变化量都相同，可以分别用两个优先队列维护这两类区间，每次转移后暴力合并重叠的区间即可。

复杂度$O((A+B)\log(A+B))$

## B. Call of Accepted

期望难度：easy

表达式求值问题可以将中缀表达式转换为后缀表达式，其中转换步骤使用调度场算法 与后缀表达式的求值 均在维基百科中有详细的介绍。

根据d运算的描述，$x \mathrm{~d~} y$本质上是一个定义在整数集的幂集上的运算，即
对于任意 $S_1, S_2 \in 2^{\mathbb{Z}}$且 $\min(S_1) \geq 0$且 $\min(S_2) \geq 1$，定义$S_1 \mathrm{~d~} S_2 = \{x \in \mathbb{Z} \mid \exists a \in S_1, \exists b \in S_2, a \leq x \leq ab\}$. 则对于任意一次有定义的d运算，都有$\min(S_1 \mathrm{~d~} S_2) = \min(S_1), \max(S_1 \mathrm{~d~} S_2) = \max(S_1) \cdot \max(S_2)$. 其中$\min(S)$和$\max(S)$分别表示集合$S$中的最小值和最大值。

再将其余三种运算扩展到$2^{\mathbb{Z}}$上，可得

$\min(S_1+S_2) = \min(S_1) + \min(S_2), \max(S_1+S_2) = \max(S_1) + \max(S_2)$

$\min(S_1-S_2) = \min(S_1) - \max(S_2), \max(S_1-S_2) = \max(S_1) - \min(S_2)$

$\min(S_1 * S_2) = \min\{\min(S_1) * \min(S_2),\ \min(S_1) * \max(S_2),\ \max(S_1) * \min(S_2),\ \max(S_1) * \max(S_2)\}$

$\max(S_1 * S_2) = \max\{\min(S_1) * \min(S_2),\ \min(S_1) * \max(S_2),\ \max(S_1) * \min(S_2),\ \max(S_1) * \max(S_2)\}$

由于我们只关注最大和最小的结果，所以可以直接用二元组$<\min(S), \max(S)>$来表示一个子表达式的运算结果，按照上述扩展定义进行运算即可。

*p.s.虽然从实际意义来看d运算不满足结合律，但如果只考虑二元组$<\min(S), \max(S)>$的话，有* $<\min(S_1 \mathrm{~d~} S_2 \mathrm{~d~} \cdots \mathrm{~d~} S_n),\ \max(S_1 \mathrm{~d~} S_2 \mathrm{~d~} \cdots \mathrm{~d~} S_n)> \ = \ <\min(S_1),\ \max(S_1)\max(S_2)\cdots\max(S_n)>$*，是无需考虑d运算的结合顺序的。*

## C. Convex Hull

期望难度：medium

## Solution 1

不考虑外层循环的情况，那么答案显然是：

$$ans=\sum_{i=1}^{\sqrt{x}}\mu(i)*\frac{1}{6}(\frac{x}{i^2}+1)*(\frac{x}{i^2})*(2(\frac{x}{i^2}))+1)*i^4$$

在加了外层循环的情况下，考虑计算$\mu(i)*i^4$的系数 令$sum(i)=\sum_{j=1}^{i}j^2$ 对于每一个$\mu(i)*i^4$，它在全部的答案中出现次数为$n-i^2+1$次，可以推出系数为$\sum_{j=i^2}^{n}sum(\frac{j}{i^2})$ 令$Max=\frac{n}{i^2}$ 考虑sum括号中的取值，可以发现，一定有$i*i$个 1，$i*i$个 2...$i*i$个 $Max-1$，$(n-i*i+11-(Max-1)*i*i)$个 $Max$ 所以，最终的系数为

$$\begin{aligned}&i*i*(\sum_{j=1}^{Max-1}sum(j))+(n-i*i+1-(Max-1)*i*i)*sum(Max)\\=&Max*Max*(Max+1)*(Max-1)/12+(n-i*i+1-(Max-1)*i*i)*sum(Max)\end{aligned}$$

考虑到模数很大，计算过程中需要用类似于分治乘法的思路或int128。

## Solution2

答案可转化为

$$\sum_{i=1}^{n}gay(i)\cdot(n+1-i)\mod p$$

在$\sum_{i=1}^{n}gay(i)(n+1-i)$中，$i^2\cdot(n+1-i)$被计入答案当且仅当i不含有平方因子。不妨考虑对所有i的因子进行容斥，即

$$\begin{aligned}Ans&=\sum_{x=1}^{[\sqrt{n}]}\mu(x)\sum_{k=1}^{[\frac{n}{x^2}]}(kx^2)^2*(n+1-kx^2)\mod p\\&=\sum_{x=1}^{[\sqrt{n}]}\mu(x)\left((n+1)x^4\sum_{k=1}^{[\frac{n}{x^2}]}k^2-x^6\sum_{k=1}^{[\frac{n}{x^2}]}k^3\right)\mod p\end{aligned}$$

其中，平方和与立方和为

$$\begin{aligned}\sum_{i=1}^{n}i^2&=\frac{n(n+1)(2n+1)}{6}\\\sum_{i=1}^{n}i^3&=\left(\frac{n(n+1)}{2}\right)^2\end{aligned}$$

直接枚举x后代入计算即可。

在计算$x\times y\mod p$时，由于p的最大值为$10^{11}$，可能会超过long long的表示范围，可以将x拆成$(a\cdot2^{20}+b)$，将y拆成$(c\cdot2^{20}+d)$，将每次乘法的运算数范围限制在$2^{20}$以内，可以直接用((((((a * c) << 20) + (a * d + b * c)) % mod) << 20) + b * d) % mod计算出结果。

时间复杂度$O(\sqrt{N})$

# D. Made In Heaven

期望难度：easy

K短路模板题。由于数据均为随机生成，直接预处理出每个点到终点的最短路后A*搜索即可。

## E. The Cake Is A Lie

期望难度：easy

二分答案，那么每次check相当于是给定一个半径的圆，然后问这个圆最多覆盖多少个点，我们可以枚举一个点，然后再枚举每个与他距离<=2r的点，就可以求出所有的相交弧，离散化之后，求出覆盖最多次的弧，就是答案了。复杂度$O(n^2\log(n)\cdot\log(30000))$。

## F. Fantastic Graph

期望难度：easy

添加源点s,汇点t。 对于原图的边，定义流量为[0,1]，s对于N个点都连边，流量为[L,R],M个点对t都连边，流量为[L,R]。那么就变成了有源汇上下界可行流问题。根据相关方法建图即可。

## G. Spare Tire

期望难度：medium

观察递推方程，不难看出通项公式的形式：

$$a_n = kp^n + an^2 + bn + c$$

代入后解得$p=1, a=1, b=1, c=-k$ 即$a_n = n^2+n$

则答案为

$$\begin{aligned}
&\sum_{i=1}^{n}[\gcd(i,m)=1](i^2+i)\\
=&\sum_{d|m\wedge d\le n}\mu(d)\cdot\sum_{t=1}^{[\frac{n}{d}]}((td)^2+td)\\
=&\sum_{d|m\wedge d\le n}\mu(d)\cdot\left(d^2\cdot\sum_{t=1}^{[\frac{n}{d}]}t^2+d\cdot\sum_{t=1}^{[\frac{n}{d}]}t\right)
\end{aligned}$$

对m分解质因数后dfs枚举所有满足条件且$\mu(d)$不为0的d，然后用求和公式计算后半部分的贡献。

## H. Hamming Weight

期望难度：medium+

将N表示为

$$N=\sum_{i=0}^{n-1}A_i 2^i,\ A_i\in\{0,1\}$$

由于位与运算每一位是独立的，不妨对每一位单独考虑它对答案的贡献：

$$Ans(N) = \sum_{i=0}^{n-1} \left[ A_i \cdot (1 + \sum_{j=0}^{i-1} A_j \cdot 2^j) + 2^i \cdot \sum_{j=i+1}^{n-1} A_j \cdot 2^{j-i-1} \right]^2$$

如果直接用FFT计算每次平方，时间复杂度为$O(n^2 \log n)$，难以接受。

考虑在N的某个区间$[L, R)$上定义答案，并将答案写成多项式的形式，即

$$Ans_{[L,R)}(x) = \sum_{i=L}^{R-1} \left[ A_i \cdot (1 + \sum_{j=L}^{i-1} A_j \cdot x^{j-L}) + x^{i-L} \cdot \sum_{j=i+1}^{R-1} A_j \cdot x^{j-i-1} \right]^2$$

其中有一项常数项，不方便合并，因此先将答案拆开：

$$\begin{aligned} & Ans_{[L,R)}(x) \\ =& \sum_{i=L}^{R-1} \Bigg[ A_i^2 + 2A_i(A_i \cdot \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + x^{i-L} \cdot \sum_{j=i+1}^{R-1} A_j \cdot x^{j-i-1}) \\ & + (A_i \cdot \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + x^{i-L} \cdot \sum_{j=i+1}^{R-1} A_j \cdot x^{j-i-1})^2 \Bigg] \end{aligned}$$

考虑到$A_i$的取值范围为0或1，即$A_i^2 = A_i$，可将答案转化为

$$\begin{aligned} & Ans_{[L,R)}(x) \\ =& \sum_{i=L}^{R-1} A_i + 2 \sum_{i=L}^{R-1} A_i \cdot \left( \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1} \right) + \sum_{i=L}^{R-1} \left( A_i \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1} \right)^2 \end{aligned}$$

令

$$\begin{aligned} S(i) &= \sum_{j=i}^{n-1} A_i \\ F_{[L,R)}(x) &= \sum_{i=L}^{R-1} A_i x^i \\ Ans1_{[L,R)}(x) &= \sum_{i=L}^{R-1} A_i \cdot \left( \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1} \right) \\ Ans2_{[L,R)}(x) &= \sum_{i=L}^{R-1} \left( A_i \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1} \right)^2 \end{aligned}$$

则$Ans_{[L,R)}(x) = S(L) - S(R) + 2Ans1_{[L,R)}(x) + Ans2_{[L,R)}(x)$。

考虑如何合并两个相邻区间$[L, mid)$、$[mid, R)$的答案。

$$
\begin{aligned}
Ans1_{[L,R)}(x) =& \sum_{i=L}^{R-1} A_i \cdot \left(\sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1}\right) \\
=& \sum_{i=L}^{mid-1} A_i \cdot \left(\sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{mid-1} A_j \cdot x^{j-L-1} + \sum_{j=mid}^{R-1} A_j \cdot x^{j-L-1}\right) \\
& + \sum_{i=mid}^{R-1} A_i \cdot \left(\sum_{j=mid}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1} + \sum_{j=L}^{mid-1} x^{j-L}\right) \\
=& Ans1_{[L,mid)}(x) + Ans1_{[mid,R)}(x) \cdot x^{mid-L} \\
& + \left(\sum_{i=L}^{mid-1} A_i\right) \cdot \left(\sum_{j=mid}^{R-1} A_j \cdot x^{j-L-1}\right) \cdot + \left(\sum_{i=mid}^{R-1} A_i\right) \cdot \left(\sum_{j=L}^{mid-1} A_j \cdot x^{j-L}\right) \\
=& Ans1_{[L,mid)}(x) + Ans1_{[mid,R)}(x) \cdot x^{mid-L} \\
& + \sum_{j=mid}^{R-1} (S(L) - S(mid)) \cdot A_j x^{j-L-1} + \sum_{j=L}^{mid-1} (S(mid) - S(R)) \cdot A_j x^{j-L}
\end{aligned}
$$

$$
\begin{aligned}
Ans2_{[L,R)}(x) =& \sum_{i=L}^{R-1} \left(A_i \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1}\right)^2 \\
=& \sum_{i=L}^{mid-1} \left(A_i \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{mid-1} A_j \cdot x^{j-L-1} + \sum_{j=mid}^{R-1} A_j \cdot x^{j-L-1}\right)^2 \\
& + \sum_{i=mid}^{R-1} \left(A_i \sum_{j=mid}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-L-1} + A_i \sum_{j=L}^{mid-1} A_j \cdot x^{j-L}\right)^2 \\
=& Ans2_{[L,mid)}(x) + Ans2_{[mid,R)}(x) \cdot x^{2(mid-L)} \\
& + 2 \sum_{i=L}^{mid-1} \left(A_i \sum_{j=L}^{i-1} A_j \cdot x^{j-L} + \sum_{j=i+1}^{mid-1} A_j \cdot x^{j-L-1}\right) \cdot F_{[mid,R)}(x) x^{mid-L-1} \\
& + 2 \sum_{i=mid}^{R-1} A_i \left(\sum_{j=mid}^{i-1} A_j \cdot x^{j-mid} + \sum_{j=i+1}^{R-1} A_j \cdot x^{j-mid-1}\right) \cdot x^{mid-L} \cdot F_{[L,mid)}(x) \\
& + F_{[mid,R)}(x)^2 \cdot (mid - L) \cdot x^{2(mid-L-1)} + F_{[L,mid)}(x)^2 \cdot [S(mid) - S(R)]
\end{aligned}
$$

化简到这里，就可以通过两次长度为$(mid-L)$和$(R-mid)$的FFT运算和若干次多项式加法、数乘和移位，由$[L,mid)$和$[mid,R)$在$O((R-L)\log(R-L))$的时间内求出$Ans1_{[L,R)}(x), Ans2_{[L,R)}(x)$。由于待求的相当于x=2时的点值，所以每次用FFT进行乘法后都可以对结果进行一次进位，从而确保在相乘的过程中系数不会溢出。

对$Ans_{[0,n)}(2)$分治求解，总复杂度$O(n\log^2 n)$.

# I. Lattice's basics in digital electronics

期望难度：easy

签到题。直接根据题意模拟即可，可以采用map来减少编码难度。

# J. Ka Chang

期望难度：medium-

按每一层的结点个数分类讨论，设阈值为$T$。

当第$L$层的结点个数$Size_L < T$时，每次1 $L$ $X$操作只需枚举这一层的所有结点，维护它们对每个结点的答案产生的贡献即可。

当第$L$层的结点个数$Size_L >= T$时，这样的层不超过$\frac{N}{T}$个，对于操作1可以直接对每个这样的层维护增加了多少point，对于每次询问直接枚举一遍即可。

对于第一种情况，可以用树状数组维护dfs序列上的区间和，时间复杂度$O(Q(T \cdot \log N))$; 对于第二种情况，时间复杂度$O(Q \cdot \frac{N}{T})$

则总时间复杂度为$O(Q \cdot (T \log N + \frac{N}{T}))$，取$T = \sqrt{\frac{N}{\log N}}$ 最优。

时间复杂度$O(Q \cdot \sqrt{N \log N})$ 。

## K. Supreme Number

期望难度：easy-

考虑到答案中任意一位都必须是1或质数，可知答案只可能由1、2、3、5、7构成。由于任意两个不为1的数字构成的两位数一定可以被11整除，所以答案中除1外的数字只能出现一次；1最多出现2次，因为111可以被3整除；而2、5、7三者一定不会有两者同时出现。因此满足条件的整数不会超过四位，全部预处理出来即可。