第二十届电子科技大学程序设计竞赛（初赛）解题报告

A. 二维格雷码

通过观察，由于对于每行或每列，相邻的两个格雷码只有一位不同，那么总存在一种分解方式，可以让每行和每列分别分解为一维格雷码。又根据调整法，可知如果要让最大格雷码值最小，两维格雷码的取值范围分别应该为 $[0, n-1]$ 和 $[0, m-1]$。这里的最大格雷码值应该是由 $n-1$ 和 $m-1$ 的二进制组合而成。因此 n 和 m 首先自减 1。

考虑 DP，设 $f_{i,j}$ 为使用 n 的前 i 位，m 的前 j 位拼出的格雷码最小值。每次转移考虑使用 n 的第 i 位还是 m 的第 j 位即可，取最小值即为答案。

单组数据时间复杂度 $\mathcal{O}(\log^2 n)$，空间复杂度 $\mathcal{O}(\log^2 n)$。

B. 修复木琴

我们所求的值为 $\operatorname{mex}(S \cup \{\operatorname{mex}(S)\})$，其中 S 指 $a_l, a_{l+1}, \ldots, a_r$ 组成的数集。考虑 $\operatorname{mex}(S)$ 就是不在 S 中的最小非负整数，而 a 是一个排列，$\operatorname{mex}(S)$ 不在 S 中，则一定在 S 外。也就是说 $\operatorname{mex}(S)$ 就是 $a_0, a_1, \ldots, a_{l-1}, a_{r+1}, a_{r+2}, \ldots, a_{n-1}, a_n$ 的最小值。根据这一层考虑，那么我们所求的其实是除 a_l 到 a_r（包括两端）外所有元素的次小值。

观察性质，我们实际上取的是一个前缀和一个后缀的次小值，即可使用类似前缀和的方法维护前缀最小值，前缀次小值，后缀最小值和后缀次小值。使用结构体和函数等编程方法将大大降低编程难度。

时间复杂度 $\mathcal{O}(n)$，空间复杂度 $\mathcal{O}(n)$。

C. Bounce!

我们考虑维护每条单位线段的高度，但是我们要查询的是过一点与它高度相同的最长序列的左右端点。这里考虑差分，即维护这个位置与前一个位置的高度差。由于线段的连续性，在改变一个区间的线段高度时，我们只需考虑最左边和最右边两端线段的合并，根据差分值即可简单判断是否需要合并到左边或右边了。使用 `set` 即可维护每个连续段。

由于本题可以离线，因此离线后可以使用线段树代替 `set`，复杂度不变。

时间复杂度 $\mathcal{O}(T \log n)$，空间复杂度 $\mathcal{O}(n)$。

D. Kanade Hates 3

考虑将每个数不断除 3 后会变成的数以及这些数的个数都用数据记录下来。这个过程的时间复杂度实际上为 $\mathcal{O}(n \log n)$，可以接受。在查询最后整个数列时，我们只需要知道查询的位置是由哪个数生成的，输出这个数最终将变为什么即可。我们可以通过求个数数组的前缀和，然后对这个数组二分查找位置即可。

由于查询的位置是递增给出的，因此我们也可以在个数数组上使用指针查询即可，每次查询判断是否需要移动指针。

需要注意数据范围与数据类型的使用。

时间复杂度 $\mathcal{O}(n \log n)$，空间复杂度 $\mathcal{O}(n)$。

E. 丁香树

对于所有无标号有根树，一条边对答案的贡献最后都会直接上传到根节点，中间没有任何加成，因此最终答案为所有边的贡献和。而对于一条边两端节点的四种属性，它们互相独立，因此对于每个属性考虑即可，最后将四个属性的贡献加和即为答案。由于对于贡献来说，只有相同和不同两种可能，因此一条边一种贡献的期望就是 $\frac{a+b}{2}$，因此由期望的可加性，对于所有边，所有属性对答案的贡献就是 $\frac{(n-1)\sum(a_i+b_i)}{2}$，也就是答案。

因此只需计算 2 在模 998244353 意义下的逆元即可，为 $\frac{998244353+1}{2}=499122177$。

单组数据时间复杂度 $\mathcal{O}(1)$，空间复杂度 $\mathcal{O}(1)$。

F. 鬼畜区的计数菌

考虑匹配过程使用 KMP 算法即可，但最终串可能很长，因此当遇到引用其他人说的话时就递归匹配。记 $f_{i,j}$ 表示串 i，开始时 KMP 指针位置为 j 时匹配的匹配个数，$p_{i,j}$ 表示匹配结束后 KMP 指针的位置。依次递归匹配即可。

时间复杂度 $\mathcal{O}(nL^2)$，空间复杂度 $\mathcal{O}(nL)$。

G. 东风谷早苗 · 改

考虑用线段树维护每个位置的行进步长，初始行进步长都为 1，由于一次修改只增加某种方向的行进步长，因此最初需要统计每个线段树节点表示的区间中每种类型的指令个数，在打标记时即可简单计算出对于该区间该种类型的指令带来的移动影响了。

对于查询 t 步后位置，由于 t 步都是从 l 开始，区间 $[l,r]$ 指令的循环，因此可以先计算一个循环后机器人的位置，之后再计算从 l 开始要走几步到最终位置，求出剩余部分的影响即可。因此如果我们把一次移动看做向量的话，最终答案其实就是一些移动向量的和。我们可以增加在线段树上维护这个向量即可，根据维护的行进步长即可维护这个向量。

由于指令个数是常数 4，因此复杂度可以接受。

时间复杂度 $\mathcal{O}(m\log n)$，空间复杂度 $\mathcal{O}(n)$。

H. 高校夏令营

我们把题意抽象出来，在一维数轴上有 n 条线段，每条线段有一个权值 w_i 和属性 a_i。需要选出一个线段的集合，在满足所有属性 $a_i=0$ 不和其它任何线段相交的情况下，w_i 的和最大，求这个最大的权值和。

考虑 DP，对 $a_i=0$ 的线段进行 DP 更新最大权，用线段树更新 $a_i=1$ 线段的贡献。将线段端点离散化后，从左到右处理每个端点，如果是一个 $a_i=0$ 线段的左端点，我们更新这条线段右端点的 DP 值，如果是一个 $a_i=1$ 线段的右端点，我们将从开始到区间左端点（不包含区间左端点）的 DP 值统一加上这条线段的权值。使用线段树加快这个维护操作。

时间复杂度 $\mathcal{O}(n\log n)$，空间复杂度 $\mathcal{O}(n)$。

I. 运动会

考虑对于所有交换操作，将形成一个置换，考虑每个置换环，对于一轮动作，每个人手中的花都将移动一位。考虑每个人可以增加一朵花，或者将手中的花清空，因此维护最后的状态，如果最后的环上没有人放下过花，那么一次交换每个人手中的花都会增多，考虑断环成链并延长一倍后维护环上区间和，通过计算循环节计算答案即可。如果有人放下过花，那么环上就会出现一系列断点，在经过这些断点后，花数就会置为 0，因此查询状态和第一次状态是相同的。这里需要小心处理前缀和求出答案。

更简单的方式是在环上倍增维护，倍增维护中不需要维护更多断点细节，也可以通过本题。但时间复杂度较直接维护差。

时间复杂度 $\mathcal{O}(n+k)$ 或 $\mathcal{O}(n\log n+k)$，空间复杂度 $\mathcal{O}(n)$ 或 $\mathcal{O}(n\log n)$。

J. 电子科技大学校园 VPN 系统

根据题意模拟即可，注意 `https` 和有端口号的 URL 的处理。

单组数据时间复杂度 $\mathcal{O}(n)$，空间复杂度 $\mathcal{O}(n)$。

K. 幸运之环

我们只需找到原图的一棵 DFS 树，维护出根到每个节点的边权异或和即可。对于原图不在 DFS 树的每条边，如果连接的两边节点的权值异或和异或这条边的权值不为 0，即可判断为出现了一个异或和不为 0 的环。

单组数据时间复杂度 $\mathcal{O}(n+m)$，空间复杂度 $\mathcal{O}(n+m)$。

L. Wordle Plus

根据题意模拟即可，注意游戏规则和 Wordle 的规则不同。

单组数据时间复杂度 $\mathcal{O}(1)$，空间复杂度 $\mathcal{O}(1)$。

M. 爱生活

考虑处理出这棵树的 DFS 序。将权值离散化，并从小到大加入树状数组中统计。首先将等于这个权值的节点在树状数组中维护值加 1，之后对于这些节点，查询它的子树中权值比它小和等于它的个数，大于的个数通过统计子树大小，作差即可得出，这些都使用 DFS 序区间查询即可。最后持久化维护小于的数组，并清空维护等于的数组。

时间复杂度 $\mathcal{O}(n\log n)$，空间复杂度 $\mathcal{O}(n)$。

N. 简单算术

构造多项式 $f(x)=\prod_{i=1}^{n}(x+a_i)$，答案就相当于对这个多项式代入 $a_1,a_2,\ldots,a_n$ 多点求值后将结果乘起来。关于 $f(x)$ 的展开形式，可以使用分治方法将所有多项式相乘，然后使用多点求值即可。

时间复杂度 $\mathcal{O}(n\log^2 n)$，空间复杂度 $\mathcal{O}(n)$。

O. Easy Math

The hardness of this problem is only about to solve the discrete logarithm under the multiplication group of finite field $\mathbb{F}_p[\sqrt{w}]$.

Find arbitrary primitive root g in $\mathbb{F}_p[\sqrt{w}]^\times$ then use Pohlig-Hellman algorithm to solve $g^z=a+b\sqrt{w}$.

Then find $z*k^{-1}$ under $\mathbb{Z}_{|F|-1}$, and output $g^{z*k^{-1}}$.

The number of such roots can be found by solving a linear modulo equation, which is also simple.

P. 公主连结

注意到要求 $8x+9y+10z=n$ 的一组非负整数解，由于 n 并不大，因此考虑预处理，枚举 x,y,z，更新对应的 n 的答案即可。实际预处理过程十分快，完全可以在时间限制内计算出。

单组时间复杂度 $\mathcal{O}(1)$，空间复杂度 $\mathcal{O}(N)$。