ICPC 2022 网络赛 (I) 讲题

北京大学 吉如一

A. 01 Sequence

- 给一个环状 01 序列，求至少修改多少个数，可以通过下面的操作删完：
  - 删去相邻的三个数，中间的一个是 1。
- q 次区间询问。
- $3 \leq n \leq 10^6, 1 \leq q \leq 10^6$。

A. 01 Sequence

- 考虑最多可以进行多少次删除操作。
- 对于连续 k 个 1，可以进行 $\left\lceil\frac{k}{2}\right\rceil$ 次操作。
- 优先进行不会导致 1 的连续段合并的操作。
- 如果所有删除都会导致 1 的连续段合并，则说明序列中没有相邻的 0，显然能够删完。
- 而每次修改可以增加 1 次操作次数。
- 因此答案为 $\max\left\{0, \frac{r-l+1}{3} - \sum\left\lceil\frac{k_i}{2}\right\rceil\right\}$。
- 预处理左右连续段长度和前缀和，即可 $O(1)$ 回答询问。

B

- 有 n 个电站，每个电站有参数 a[i],b[i], 同时需要花费一单位的材料进行建造。建造电站不花费时间。
- 假定你在时刻 t 知道了编号为 i 的电站，则在时刻 t+a[i],t+2a[i],t+3a[i]… 该电站都能够产出恰好 b[i] 单位的材料。
- 初始时刻你有一个单位的材料，询问最少需要多少时间才能将所有电站建设完毕。
- n <= 16, a[i] <= 10^6, b[i] <= n

B

- 对于每个电站 i，我们记录用于建设其的材料是由哪个电站生产的。假定电站 j 生产了电站 i 建设需要的材料，则我们设 fa[i]=j，据此我们可以得到一个树形结构，根节点为初始建设的电站。

- 据此我们可以设如下状态：
  - F[i][j]: 建设完了集合 i 的电站，根节点为 j，最少花费的时间
  - G[i][j]: 建设完了集合 i 的电站，初始总共有 k 单位材料，最少花费的时间。
- 两者状态的转移都只需要枚举其中的一颗/若干个子树对应的点集合即可，因此上述算法总复杂度 O(3^n*n)

B

- 当然，由于 n 仅有 16，选手完全可以采用爆搜加上剪枝的方式通过本题。也可以利用奇怪的乱搞来帮助骗分（笑）。
- --大家各凭本事，不管算法复杂度分析不分析的出来，能在时限里跑出来的就是好算法！

C

- 给定一棵 n 个点的树，Alice可以进行两种操作：
  - 删除：把这个点和与之相邻的边删掉
  - 收缩：把一个二度点缩掉（删掉这个点和相邻的边，再把该点原本连着的另两个点连起来）
- 我们希望把这棵树删干净，问最少需要多少个“删除”操作。多组数据。
- $\sum n <= 10^6$，时限1s

C

- 容易发现如果能“收缩”那一定会选择“收缩”，且该操作除了把缩掉的那个点删了之外，其余所有点的度数不变。
- 在能“收缩”就“收缩”的情况下：对于一个“删除”操作，如果删的是一个>2度点，那么整张图里的0/1度点总数量一定不变；如果删的是一个0/1度点，那么整张图里的0/1度点总数量一定减一。
- 而把整棵树删完等价于整张图0/1度点总数量为0。
- 故每次操作我们一定删一个0/1度点，然后能缩就缩。答案即为整棵树中0/1度点的数量。
- 时间复杂度 O(n)。

D

- T 次询问，每次询问在一个区间 [l, r] 里找到一个数 x，满足 popcount(x)=ctz(x)，或者声明没有。
- popcount是二进制位 1 的个数，ctz是末尾 0 的个数。
- $1 \le T \le 10^5, 1 \le l \le r \le 10^9$

D

- 首先有个数位做法，考虑枚举 1 的个数/末尾 0 的个数，然后想怎样找一个在 [l, r] 内的这样的数。
- 可以发现 $\le r$ 的最大的数是很好求的，只要高位到低位贪心放 1，能放就放，最后留一个 1 保证末尾 0 的个数就可以了。
- 于是直接做就好。单组复杂度 $O(\log n)$，实现的不好可能是 $O(\log^2 n)$，但常数良好应该也能通过。

D

- 本题还有一种偏暴力的做法。事实上范围内合法的数大约只有 50 万种，因此用各种方法搜出来，然后询问时候直接在里面二分就可以了。

E

- 给定点集 S，每次操作可以选择 S 中的两个点，并把它们之间线段上的某一个点加入到 S 内
- 令 f(S, p) 表示最少操作多少次才能把 p 加入到 S 内，如果无法加入则定义为 10^9
- 给定初始点集 S，m 组询问，每次给出点 p，你需要计算
- $\sum_{T\subseteq S} f(\mathrm{T},p)$
- |S|, p ≤1000，保证没有重点

E

- 当 p 位于 T 某两个点之间的线段上时，$f(T,p)=1$
- 当 p 位于 T 的凸包内时，$f(T, p) = 2$
- 其他情况 $f(T, p) = 10^9$

- 问题转化成统计 S 有多少个子集 T 满足前两种情况
- 极角排序

F

- 给定n,k，求有多少个a[1…k]满足a[i]|a[i+1](i<k)，a[k]<=n。
- n,k <= 10^9

F

- 令f(x)表示a[k]=x时的方案数。
- 不难发现f是积性函数，f(p^e)=C(e+k-1,e)。
- 使用积性函数前缀和的算法即可。
- 时间复杂度O(n^(3/4)/logn)。

G. Read the Documentation

- n 秒钟，每秒钟读文档会有一个收益，同时根据连续读的时间增加愤怒值。连续读 5 秒文档或者愤怒值超过 T 就会被禁赛。问最大收益。
- $1 \le n \le 100$。

G. Read the Documentation

- DP。
- 状态只需要记录长度为 1,2,3,4 的连续段数，设其分别为 a, b, c, d，则状态数为 $2a + 3b + 4c + 5d + e \leq n + 1$ 的方案数，约 $\frac{n^5}{1200}$。
- 时间复杂度 $O(n^5)$，空间复杂度 $O(n^4)$，但常数极小。
- 也可以使用哈希表或 std::map 储存状态，均可通过。

H

- 给出一个 C– 程序，你需要计算它运行时会调用 library 多少次

- `arithmetic` performs some arithmetic calculations.
- `library` invokes a function in the standard library of C--.
- For any C-- expression e and integer $w \in [1, 100]$, `repeat` e `for` w `times` repeats expression e for w times.
- For any two C-- expressions e_1, e_2, $e_1\ e_2$ runs expressions e_1, e_2 in order.

- 写一个简单的 parser 把程序结构还原出来即可直接计算

- 给定一个长度为 n 的排列，将这个排列shift（把排列的第一项扔到最后一项）n 次能依次得到 n 个不同的排列。
- 求这 n 个的排名之和。
- 一个长度为 n 的排列的排名定义为将所有长为 n 的排列按字典序排序之后的排名。
- $n <= 2.5\times10^5$，时限7.5s

- 考虑如何计算一个排列的排名。容易发现对于一个长度为 n 的排列 $p_1, p_2, \ldots, p_n$，它的排名等于这个式子：

$$1+\sum_{i=1}^{n}(p_i-1-\sum_{j=1}^{i-1}[p_i>p_j])(n-i)!$$

- 其中 $[p_i>p_j]$ 在 $p_i>p_j$ 时为 1，否则为 0。
- 相当于对于每一位来说，假设这一位之前已经确定了，那么在这一位上还有多少比它小的选择，乘上后续的排列总数。

- 容易发现这个式子里只有 $\sum_i \sum_j$ 同时存在的那部分是难算的（包括考虑所有 n 个排列的排名之和）。故我们只关心这部分。
- 对于其中一个排列来说，一对 (i,j) 对答案产生的贡献是$-[p_i>p_j](n-i)!$。而我们把 n 个排列综合考虑进来，就会发现，若 i+n-j mod n=k，那么它们对答案产生的贡献是 $-[p_i>p_j]\sum_{m=0..n-k-1}m!$。
- 换句话说，设 $f[k]=\sum_{m=0..n-k-1}m!$，那么它们对答案的贡献就是 $-[p_i>p_j]$ f[i+n-j mod n]。再设 P 是 p 的逆排列，那么对答案的贡献就是 [i>j] f[P[i]+n-P[j] mod n]。

- 那么现在我们只需要对于每个 k，统计所有二元组 (I,J) 中有多少个二元组满足 P[I]+n-P[j] mod n=k。（其中 I>J）
- 考虑把排列分块，每块大小为 S，块数为 n/S。对于每个块内部直接用 O(S^2) 暴力统计，对于不同块之间的元素用单个块 O(nlogn) 的复杂度卷积统计。
- 故总复杂度为 O(S^2*n/S+nlogn*n/S)=O(nS+n^2logn/S)。取 S=sqrt(nlogn)，则复杂度为 O(nsqrt(nlogn))。足以通过本题。

- 但实际上由于常数等原因，将 S 取为 7000 往往程序运行得更快。如果你被卡常了，可以试试更改 S 的大小。（否则的话可以考虑带个常数更好的 DFT 板子）

J

- 设计如下抽卡系统，满足如下条件:
  - 假定上一次抽出 SSR 后，已经 p 抽没有抽出 SSR，则下一抽抽出 SSR 的概率为 a[p+1]
  - 保证 X 抽抽出一个 SSR，也就是 a[X]=1；但是不保证 X-1 抽抽出一个 SSR, 也就是对于所有 x < X,a[x] < 1.
  - 保证每一抽抽出 SSR 的概率不为零，也就是 a[x]>0 恒成立
  - 在上述规则下，抽出一个 SSR 的期望恰好为 Y.
  - 你需要额外保证 a[p] 可以写成分数的形势，且分子分母绝对值不超过 10000.设计如下抽卡系统，满足如下条件:
- 保证 2 <= Y < X <= 100，数据保证有解。

J

- 考虑 b[i] 表示抽出上一次 SSR 后，恰好花费 i 抽抽出 SSR 的概率。根据 b[i] 不难得出 a[i] 的分数表示。
- 根据题目约束知道有：\sum b[i] = 1, \sum ib[i] = Y。
- 我们做如下最简单的假设：b[1]=…=b[i],b[i+1]=…=b[X]，且据此解方程即可。
- 在取 i=1 或者 i=k-1 的时候，不难发现我们可以让 b[i] 的分母全部取 X(X-1)/2，且满足上述条件。因而我们得到的 a[i] 也满足题目中所描述的条件。

K

- 你有一个属性值为 1 的随从，游戏会进行 n 个回合。每个回合你可以使用一次英雄技能，使用结束后会进行一次战斗。
- 第 i 轮战斗需要属性值不低于 x_i 才能获胜。
- 英雄技能有一个参数 w，初值为 1。每一次使用有两种选项。
  - 将参数 w 的值永久加一。
  - 将随从的属性永久加 w。
- $n, x_i \leqslant 5000$

K

- 令 $m = (\max x_i)^{0.5}$。
- 结论1：最多只会输 $O(\mathrm{m})$ 场
- 简单策略：先进行 m 次操作 1，之后全是操作 2。
- 结论2：最多只会进行 O(m) 次操作 1。
- 简单替换：考虑把最后一次操作1替换成操作 2。

- 令 dp[i][j][k] 表示前 i 场选择了 j 次操作 2，输 k 场的情况下，最大的属性值

L

- 给定字符串 s,t。
- 询问 s 的最长的子序列 s’，使得 s’和t的最长公共子序列长度至多为 1.
- |s|,|t| <= 500000

L

- 假定 f[i][j] 代表在前 i 个字符中，子序列s ‘最后一个字符为 j，且满足条件的情况下，最长可能的长度。
- 在进行末尾添加字符转移的时候，我们只需要检查 j+s[i] 是否作为 t 的子串，如果不是即可从 f[i-1][j] 转移到 f[i][s[i]]。否则该转移不合法。
- 因而上述算法时间复杂度为 O(26n)，足以通过本题。

L

- 正确性证明：
  - 如果 j+s[i] 没有出现在 t 的子序列中，但是之前选择的 k+s[i] 出现在了 t 的子序列中。我们尝试通过反证法证明上述情况不可能发生。
  - 如果发生上述情况，则必然 j 在 t 中第一次出现的位置在 k 之后，因此 t 必然有子序列 k+j。同时由于 j 是我们选择的 s’ 的最后一个字符，因此 k+j 也是我们选择的 s’ 的子序列。这同 s’ 和 t 最长公共子序列长度不超过 1 矛盾。