2019牛客暑期多校训练营 第六场

北京航空航天大学

A – Garbage Classification

- 垃圾分类题
- 按题意模拟即可

B – Shorten IPv6 Address

- 模拟题

- 首先把前导 0 去掉
- 每次一定会删去一段连续且尽可能长的 0
- 有一个坑点是首尾和中间同样长度的 0 对答案的贡献是不一样的
- 如果不想讨论可以直接枚举删去哪一段 0，暴力计算答案
- 推荐学习所有提交中最短的几份代码的写法。

C – Palindrome Mouse

- 以下做法仅供参考，说不定有更厉害的做法
- 题解的做法需要一些Palindrome Series的知识，下文中link[u]指节点u的后缀链接，diff[u]=len[u]-len[link[u]]，slink[u]指节点u在link链上最近的diff与u不同的节点，slink链的长度是O(log(n))级别的。
- 首先本质不同回文子串的数量是容易求得的，考虑计算存在包含关系的pair数量。
- 用回文树容易求得每种回文子串第一次出现的前缀，假如我们可以维护在这个前缀中每种回文子串最后出现的位置，就可以通过区间询问求出该第一次出现的回文子串的本质不同回文子串个数。
- 设 last_i 为前缀 i 中出现的每种回文子串最后出现位置（以字符串开头位置为准）的集合（若两个不同的回文子串u和v在前缀i中的最后出现位置都是left，设|u|>|v|，由对称性left+|u|-|v| 也是v的出现位置，与left是v的最后出现位置矛盾）。

C – Palindrome Mouse

- 考虑last_i-1到last_i的变化，增量维护这个集合
- 对于前缀i的最长回文后缀u，u的所有回文后缀v的最后出现位置都会变为i-|v|+1，对于这些位置，不属于last_i-1当且仅当link[v]=slink[v]成立（一样可以用对称性证明），因此增加的位置数量是O(log(n))的。
- 从增加的位置不难发现，减少的位置一定是由新出现的位置替换了，因此减少的位置数量也是O(log(n))的
- 定义st[u]为link[u]链上最近的v满足link[v]=slink[v]，将series的信息保存在st[u]处。每个这样的节点维护一个二元组(left,maxl)的单调栈，保证left严格单调增，maxl严格单调减。每个二元组表示series中|v|<=maxl的回文串v的最后出现位置不小于left+maxl-|v|.
- 最后的区间询问可以用树状数组来维护last_i即可。
- 时间复杂度O(nlog^2(n))，空间复杂度O(nlog(n)).

D – Move

- 首先这道题箱子容积和答案没有单调性
- 比如：
  - 15 5
  - 39 39 39 39 39 60 60 60 60 60 100 100 100 100 100
  - 199 为一个合法的答案，但 200 不是，201 也不是。

D – Move

- 考虑到答案下界显然为 **ceil(sum / k)**
- 上界为 **ceil(sum / k) + maxV**
  - 假设某个答案 ans 装不下，那么每个箱子的剩余空间都 < maxV
  - 此时 k * (ans - maxV + 1) <= sum
  - ans <= sum/k + maxV - 1
- **check** 一个答案的复杂度为 **O(nlogn)**
- 所以我们直接在这个范围内枚举答案，复杂度为 **O(maxV * nlogn)**

D – Move

- 或者你注意到 f(x) , f(x + maxV) , f(x + 2maxV) … , f(x + k * maxV) 是单调的！
- 那么就枚举 x 之后二分，复杂度为 O(maxV * nlog^2(n)

E – Androgynos

- 以下做法仅供参考，说不定有更厉害的做法
- 首先两个图同构有个必要条件是边数要相同，也就是说n阶完全图的边数得是偶数
- 若 n = 4k：
  - 先考虑一个更简单的情况，n=4，相信大家都会手构（如果你不会，还可以枚举映射关系跑个2sat）；观察n=4的情况，容易发现，可以将顶点分为两组，使得i和fi都来自不同组；容易想到：原图中的团在补图中中会变成独立集。
  - 对于n=4k(k>1)的情况，可以先把顶点分成4块，2块内部连成团，2块内部不连组成独立集，然后按n=4的情况连块之间的边即可
- 若n = 4k + 1：多出的一个点向所有团连边即可
- 若n = 4k + 2 或 4k + 3：无解

F – K-ary Heap

- 求 **Rank** 通用做法，固定一个前缀，算后面的合法排列数量。
- 首先 **dp** 出一个子树内的合法排列数量 **F[]**。

- 如果给定了一个排列的前缀，剩下没有确定的位置构成了若干个空子树。
- 每个空子树的限制只有内部所有点的权值不小于根的父亲的权值。

- 从限制条件紧（根的父亲的权值大）的子树开始选，对答案的贡献是一堆组合数的乘积。

F – K-ary Heap

- 如果计算比给定排列小的排列数量，需要枚举第几位开始变小以及变小成什么，一共有 O(n^2) 种前缀，每个前缀计算复杂度为O(n), 总复杂度为 O(n^3)。

- 反过来计算比给定排列大的排列数量，可以发现当枚举第几位开始变大，考虑这一位是多少的时候，可以通过一个限制条件来计算方案数，而不用再枚举具体是多少。每个前缀计算复杂度为O(n), 总复杂度为 O(n^2)。
- 预处理叶子节点，计算时只考虑n/K个非叶子节点，可以将复杂度降为 O(n^2 / K)。

G – Is Today Friday?

- 搞题
- date -> week - 预处理/蔡勒公式

- 合法日期 + 周五的限制很紧
  - 对于任何一个加密后的日期，只有几万个排列能将其映射到合法日期。
  - 只对前若干个日期求出所有合法的排列后，再依次检查是否对所有日期合法。
  - 坑：有重复的日期，需要去重。

H – Train Driver

- 以下做法仅供参考，说不定有更厉害的做法

- 先(SA+SB)次bfs预处理出A中每个点和B中每个点到所有点的距离
- 考虑枚举了一个A中的点a和一个B中的点b，如何很快的计算出所有点作为第三个点的答案
- 首先初始化每个点i的dis[i]=dis[a][i]+dis[b][i]，然后跑一遍最短路即可。
- 这个最短路可以用bfs来计算
  - 首先将所有点按初始dis基数排序放在队列1，之后入队的点都放进队列2
  - 每次扩展选两个队首dis较小的元素即可
- 时间复杂度O(SA*SB*n).

I – Can They Go to Galar?

- 概率+图论题

- 根据期望的线性性，答案就是每个点被选取的概率的和
- 不妨先考虑一下树上的情况
- 树上的每个点只能被它的父亲带走
- 因此对于点 u，$prob_u=prob_{fa[u]}*e_{fa[u],u}$

I – Can They Go to Galar?

- 回到仙人掌上，我们求出所有的点双连通分量（一定是一个单点或一个简单环）
- 可以发现，每个环上恰好有一个点或者是被父亲带走，或者是在上一个环中（割点）
- 环上其它的点都是从该点出发逐个被带走的

I – Can They Go to Galar?

- 为了讨论方便，我们设环是 1-2-...-n-1，1 号点是初始点，它被带走的概率是 1
- i 号点被带走可以通过 1-2-...-i 这条路径，也可以通过 1-n-(n-1)-...-i 这条路径
- 它被带走的概率是 1 减去上述两条路径都没有带走它的概率，即
- $1-(1-e_{1,2}*e_{2,3}*...*e_{i-1,i})(1-e_{1,n}*e_{n,n-1}*...*e_{i+1,i})$
- 可以用前缀积预处理，从而 O(n) 计算一个环的答案
- 总复杂度 O(n)

J – Upgrading Technology

- 一道有点坑的题，做法挺多

- 一种错误的想法是枚举有 j 个 level 升满，然后对第 i 种科技从 pre[i][j] 到 pre[i][m] 中选择最小的那个作为第 i 种科技的最终 level（这里 pre[i][j] 表示第 i 种科技升前 j 个 level 的代价和）

- 这种做法的问题在于 d_j 可能是负的，贪心选取最小的 pre 可能导致 j+1 之后某些负的 level 使答案变差

J – Upgrading Technology

- 将刚刚的想法稍微修改一下即可得到一种正解

- 枚举一种科技 i 作为 level 最小的科技，并枚举它的 level j，那么其它科技的 level 就可以在 j 到 m 之间任取了

J – Upgrading Technology

- 另一种想法是 dp，设 dp[i][j] 表示前 i 种科技最小 level 为 j 时，最小的 cost
- 转移方程为 dp[i+1][j]=min(dp[i][k]+pre[i+1][u])，其中 k≥j，u≥j，且它们中至少有一个为 j
- 用一些前后缀加速转移

Thanks