Comet OJ - Contest #8 题解

A.杀手皇后

按题目中的方法比较两个串的字典序。

即从左边第一位开始，如果两个串这一位上的字符不一样，则结束比较，这一位上字符小的串字典序更小，否则继续比较下一位。特殊地，如果比较至某一位时某个串比另一个串长度短提前结束，则长度短的串字典序更小。

c++选手也可以直接利用stl中的string直接比较两个串的大小。找出字典序最小的串即可。时间复杂度$O(n*|s|)$

B.支援城市

直接计算时间复杂度为$O(n^2)$

$$\sum_{i=1}^{n}(w_i-w_x)^2=\sum_{i=1}^{n}w_i^2+n\cdot w_x^2-2\times w_x\times\sum_{i=1}^{n}w_i$$

预处理出$\sum_{i=1}^{n}w_i^2$和$\sum_{i=1}^{n}w_i$即可在$O(n)$的时间复杂度内计算出答案。注意开longlong

C.符文能量

注意到精炼操作后答案为$\sum_{i=1}^{n-1}b_i*a_{i+1}$，与合并的顺序无关，所以关键在于如何选取精炼的区间。

注意只能选取一段连续区间进行精炼，考虑dp：

$f[i][0]$表示前i个数，没有进行精炼的价值和 $f[i][1]$表示前i个数中，有一段后缀进行了精炼，即精炼区间包含i的最大价值和 $f[i][2]$表示前i个数中，精炼已经结束，即精炼区间在i左侧的最大价值和

则有转移

$$\begin{aligned}
&f[i][0]=f[i-1][0]+b[i-1]\cdot a[i]\\
&f[i][1]=\max(f[i-1][1]+b[i-1]\cdot a[i]\cdot k,f[i-1][0]+b[i-1]\cdot a[i]\cdot k)\\
&f[i][2]=\max(f[i-1][2]+b[i-1]\cdot a[i],f[i-1][1]+b[i-1]\cdot k\cdot a[i])
\end{aligned}$$

时间复杂度$O(n)$

D.菜菜种菜

把所有土地排成一行，从左到右第 i 个编号为 i 。容易发现，一块土地 i 能不能作为家，只跟他能直接到达的土地中，在它左边最靠右的一块（记作 x_i，不存在则为 0）和在它右面最靠左的一块（记作 y_i，不存在则为 $n+1$）有关。先预处理出 x_i 和 y_i，则对于一次询问 $[L,R]$，问题就是计算满足 $L \le i \le R, x_i < L, y_i > R$ 的所有土地菜值之和。

直观上看是一个三维数点的问题，在规定的时空限制内难以解决，需要进一步分析。形式化地记 $s(a,b,c,d,e,f)$ 为满足 $a \le i \le b, c \le x_i \le d, e \le y_i \le f$ 的土地的菜值之和，则答案为 $s(L,R,0,L-1,R+1,n+1)$。

根据容斥原理，对 x_i和 y_i 的限制进行转化，有：

$$\begin{aligned} s(L,R,0,L-1,R+1,n+1) =& s(L,R,0,n+1,0,n+1) - s(L,R,L,n+1,0,n+1) \\ & - s(L,R,0,n+1,0,R) + (L,R,L,n+1,0,R) \end{aligned}$$

这个式子有四项，第一项是一个简单的区间和，第二项只需要满足 $i \le R, x_i \ge L$ 的限制（由于 $x_i < i$，这样一定能保证 $i \ge L$），同理第三项只需要满足 $i \ge L, y_i \le R$，最后一项，如果一个 i 满足了$x_i \ge L, y_i \le R$，那么 i 一定在区间 $[L,R]$ 内，所以对于 i 的限制就取消了。如此一来，转化为了三个二维数点的问题（第二项限制 i 和 x_i，第三项限制 i 和 y_i，第四项限制 x_i 和 y_i）。

二维数点有经典做法，将所有询问离线，点和询问按第一维排序，第二维作为下标，顺次扫描，遇到一个点将它第二维对应的位置加上它的权值，遇到一个询问就查询对应第二维的对应的区间和，只需一个支持单点加，区间求和的数据结构，树状数组即可胜任。

时间复杂度 $O(n \log n)$，空间复杂度 $O(n)$。（这题本来放在C，是不是很简单？）

E.神奇函数

设 x 的唯一分解为 $x = \prod_{i=1}^{k} p_i^{e_i}$，则有：

$$\begin{aligned} f(x) &= \prod_{i=1}^{k} p_i^{\lfloor \frac{e_i}{2} \rfloor} = \frac{n}{\prod_{i=1}^{k} p_i^{\lceil \frac{e_i}{2} \rceil}} \\ &= \sum_{y=1}^{n} \left[\left(\prod_{i=1}^{k} p_i^{\lceil \frac{e_i}{2} \rceil} \right) | y \right] \\ &= \sum_{y=1}^{n} [x | y^2] \end{aligned}$$

则答案为：

$$\begin{aligned}\sum_{i=1}^{n} f(i) &= \sum_{i=1}^{n}\sum_{j=1}^{i}[i|j^2] = \sum_{j=1}^{n}\sum_{i=1}^{n}\sum_{k=1}^{i}[ik=j^2] \\ &= \sum_{d=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n}\sum_{k=1}^{i}[\gcd(i,k)=d][ik=j^2] \\ &= \sum_{d=1}^{n}\sum_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum_{k=1}^{i}[\gcd(i,k)=1][i,k\text{都是完全平方数}] \\ &= \sum_{d=1}^{n}\sum_{i=1}^{\sqrt{\lfloor\frac{n}{d}\rfloor}}\sum_{k=1}^{i}[\gcd(i^2,k^2)=1] = \sum_{d=1}^{n}\sum_{i=1}^{\sqrt{\lfloor\frac{n}{d}\rfloor}}\varphi(i) \\ &= \sum_{i=n}^{\sqrt{n}}\varphi(i)\left\lfloor\frac{n}{i^2}\right\rfloor\end{aligned}$$

线性筛即可，时间复杂度$O(T\sqrt{n})$。（本来这题放在D的，std直接就是根号的了。。。）

F.黄金体验

首先考虑没有修改操作的情况。

由于点权始终是正的，可以看出选择的点必然都是树的某个叶子。当$k=2$的时候，一定会选两个路径上点权和最大的叶子，即选择这棵树的带权直径。

考虑$k>2$时的情况，可以发现一个很好的性质：选k个点时的最优解一定是在选$k-1$个点时的最优解的基础上再多选一个点。知道这个性质后，就有了一个贪心的做法，即先找出树的带权直径，以直径的一端为根，每次选一个到跟的路径的点权和最大的点作为这次选的点，并且把这一段路径上的点权都设为零。这样选出的前$k-1$个点即为题目要求的答案。可以通过简单归纳整明这个贪心的正确性。设前2到$k-1$个点的答案都满足上述性质，可以发现如果此时要多加一个点进来，将之前已选的点换成其他的点收益都是负的，而不做改动直接加入一个新点时收益是最大的。

简单分析这个贪心做法，发现的实际上是对树做了一个带权的长链剖分，而选k个点的答案即为权值前$k-1$大的链的权值和。

有了上述做法，再去考虑修改操作。如果能在修改时维护这个带权的长链剖分的结构，询问时还是可以直接输出前$k-1$大的链。修改操作只有加点权，这会造成的影响是让点权增加的点所在的链向上延伸，这个操作可以用lct来实现，即对这个点进行access操作，如果父亲节点的重儿子由原本的点变为这个点，则将这个点和他的父亲连起来，继续向上access，否则再往上的点重儿子都不会有改变，直接结束操作即可。这样做的复杂度与lct的access操作复杂度相同，均为$O(\log n)$，而维护长链剖分的同时还需要维护一棵每条链权值的平衡树，所以单次修改复杂度为$O(\log^2 n)$。此时注意到，长链剖分做法的前提是以直径的一个端点为根，而直径在操作过程中有可能改变，这时就需要给整个长链剖分的结构换根。如果随意指定一个点为根，新的结构可能会和原本的

结构有一些不同，但是由于我们只需要以直径的任意一个端点为根，换根操作实际只需要将原本直径的端点中不是根的点换上来，此时很容易发现，长链剖分的结构是不会有任何改变的，于是可以直接用lct里的换根操作来实现。

至此，能在时间复杂度$O(q\log^2 n)$，空间复杂度$O(n)$内解决这个问题。