南昌网络赛题解

PERFECT NUMBER PROBLEM

```c
#include <stdio.h>
int main(){
    int x[5] = {1, 2, 4, 6, 12};
    int i;
    for(i = 0; i < 5; i++) {
        printf("%d\n", (1 << x[i]) * ((1 << x[i] + 1) - 1));
    }
    return 0;
}
```

Greedy HOUHOU

考虑用线段树维护这个序列，对于线段树的一个节点 k，我们用 $f[k]$ 表示当选取的区间恰好是这个节点对应区间时的最优解，用 ls 和 rs 表示 k 节点的左儿子和右儿子标号，显然，$f[k] = f[ls] + v$，v 是个未知数，现在考虑如何求 v，我们设节点 ls 的最小值为 w，再设节点 rs 的左儿子为 tl，右儿子为 tr，显然， w 一定会被 HOUHOU 吃掉，然后考虑节点 rs，如果节点 tl 的最小值不小于 w，那么，节点 tl 对应的整个区间 HOUHOU 都会忽略掉，这个时候我们只需要考虑节点 tr，反之，节点 tl 的最小值一定会被吃掉，那么节点 tr 对 v 的贡献一定为 $f[rs] - f[tl]$（原因留给读者思考）。根据上述过程，在计算 $f[k]$ 的时候，我们需要递归搜索节点 rs 及它的后代，但是对于任意一个节点，我们要么前往它的左儿子，要么前往它的右儿子，换句话说，我们在给一个节点计算 v 的值时，至多需要搜索 $\log n$ 个节点，那么计算/更新一个节点信息的时间复杂度为 $\mathcal{O}(\log n)$，由于线段树只有 n 个节点，因此建树的复杂度为 $\mathcal{O}(n \log n)$。有了以上的分析，查询和更新操作就很容易了，对于查询，线段树的区间查询至多需要查 $\log n$ 个节点，如果我们把这些节点按他们维护的区间的左端点升序排序，那么影响第 i 个节点答案的是第 $i-1$ 个区间的最小值，那么我们在查询第 i 个节点的时候，将第 $i-1$ 个节点的最小值记录下，然后利用上述过程递归搜索第 i 个节点对应的子树就可以得到这个节点贡献的答案，因此，单次查询的时间复杂度为 $\mathcal{O}((\log n)^2)$，更新操作同理，单次更新需要更新 $\log n$ 个节点，每个节点更新信息的时间复杂度为 $\mathcal{O}(\log n)$，因此单次更新的时间复杂度也是 $\mathcal{O}((\log n)^2)$。

Angry FFF Party

因为斐波那契的增长是指数级别的，所以两层斐波那契增长的很快。只需要矩阵乘暴力处理前 28 项。然后对于W反着算，能减就减，

Match Stick Game

动态规划。注意到式子的长度不超过 100，且一个数字或者符号至多用 7 根火柴棒，因此总的火柴棒数量不会超过 700，所以我们可以用 $dp[i][j][k]$ 表示仅考虑前 i 个位置，剩余 j 根火柴棒没用，前 i 个位置最后一个符号位 k ($k=0$ 或者 1 分别代表加号和减号)时的最大值，由于题目要求了符号数量和数字数量不变且每个数字的数位个数也不变，因此在转移时我们没必要枚举某个符号后面完整的数字是多少，而可以只单独考虑每个数位的情况，那么每个状态至多只会转移向 10 个后继状态，转移的时候稍微注意下前导零问题即可，时间复杂度为 $\mathcal{O}(140n^2)$。

Card Game

这道题可以分为两部分。

首先，组合游戏部分：

这是一个由多个 有卡牌的位置 组合起来的游戏，组合局面的 sg 值就是初始每个位置 sg 值的异或，因此首先要求出每个位置有卡牌时的 sg 值。

搞清楚什么情况下先手必败是解题的关键。这个卡牌游戏的原型就是翻一个、两个或三个硬币的游戏，做过 HDU-3537 的应该对这个模型比较熟悉。

不熟悉模型的也可以通过打表找规律推出 sg 值，最后可以发现，对于某个编号 x 的位置，它的 sg 值要么是 $2\times x$，要么是 $2\times x+1$，继续观察，可以发现每个 sg 值的二进制形式中 1 的个数都为奇数，显然 $2\times x$ 和 $2\times x+1$ 中只会有一个数满足，因此判断其中一个是否满足即可。

求出每个位置的 sg 值以后，组合局面就是每个有卡牌位置的 sg 值的异或和。第一部分就完成了。

第二部分：

我们需要的其实是初始局面中最长的异或和为 0 的连续区间，可以先求出初始局面的前缀异或和。

已知前缀异或和，假设只要求不包括空位在内的最长的异或和为 0 的连续区间，我们可以通过维护所有前缀异或和值出现的最小的位置和最大的位置来维护出最大长度。

如：$1\ 2\ 1\ 2\ 1$ (编号 $0-4$)。 这个序列中前缀异或和为 1 的最小位置 0，最大位置 4，长度为 4，即区间编号 $1-4$ 异或和为 0。2 的最小位置为 1 ，最大位置为 3，长度为 2，即区间编号 $2-3$ 异或和为 0。取其中长度最大的，故为 4。

因为前缀异或和可能比较大，可以采用 unordered_map 等方法来维护。

对于本题，要求最长的连续区间应该还包括没有卡牌的空位，因此维护最小位置和最大位置时需要注意，最大位置应该是下一个有卡牌位置的前一个位置，当然，对于最后一个有卡牌的位置来说，则是最后一个位置。

假设 0 代表位置没有卡牌：$1\ 2\ 1\ 2\ 1\ 0\ 0\ 2\ 0$ (编号 $0-8$)。则此时 1 的最大位置应为 6，2 的最大位置为 8。

Information Transmitting

首先考虑单组询问的情况，可以先跑一遍 dfs 算出信息上传到根的概率，再跑一遍 dfs 算出信息从根传递到子树上所有节点的概率。

在考虑 2 个信息源传递到同一个点的概率时（不妨设其概率分别为 x 和 y ），该点能收到信息的概率可以使用容斥原理计算，为 $x+y-xy$。

再考虑多组询问的情况，由于总点数不超过 5×10^5，因此我们对于当前询问的点把关键点拿出来 构建虚树 ,这样保证了多次询问的点不超过 5×10^5。

tsy's number

根据欧拉函数计算公式可化简分式成 $j\times k^2$ 再令 $d=gcd(i,j,k)$ 则可以化简成

$\sum_{d=1}^{n}\varphi(d)\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}jk^2[gcd(i,j,k)==d]$ 对后面的式子进行反演

得 $\sum_{d=1}^{n}\varphi(d)\sum_{i=1}^{\frac{n}{d}}\mu(i)\lfloor\frac{n}{id}\rfloor sum(\lfloor\frac{n}{id}\rfloor)sum2(\lfloor\frac{n}{id}\rfloor)(i,j,k)i^3$ 但是显然这个式子不能简单计算令 $T=id$ 对公式整理得 $\sum_{T=1}^{n}\lfloor\frac{n}{T}\rfloor sum(\lfloor\frac{n}{T}\rfloor)sum2(\lfloor\frac{n}{T}\rfloor)T^3u\times\varphi(T)$ 显然前面多项可以分块计算

我们可以预处理一下 $T^3u \times \varphi(T)$ 的前缀和 $(u \times \varphi(T))$ 显然是积 性函数可以用线性筛出来)就可以再根号时间内算出来了 总复杂度 $\mathcal{O}(n + T \times n)$ (如果有兴趣的同学可以思考一下后面的式子的杜教筛方法)

Coloring Game

对于 2 到 $n-1$ 列每列有 3 种涂法，第 1 列和第 n 列只有 2 种，答案就是 $4 \times 3^(n-2)$ 需特判 $n = 1$ 的情况

Max answer

先用单调栈判断以每个值为最小值的最大左边界和右边界。后对对每个值分成负数和正数讨论取可行区间内的最小或最大值，方法为求前缀和和后缀和，然后用线段树求区间最值。

Distance on the tree

Solution1：（二分+主席树）

将边权离散化，按链建主席树，每次都是建的当前节点到根节点的主席树

查询u到v的时候，只需处理出u到lca和lca到v两条链即可

这两条链对应的主席树为 u 的主席树 +v 的主席树 $-2 \times lca$ 的主席树

二分 k 在线段树里的 rank，统计两条链对应的主席树中不超过 rank 的值的个数即可

这个题解写的树剖好像只用来查询 LCA 了

Solution2:（离线+树剖）

将边权离散化，把树形先建好，

然后把加的边权 w 和读入的边权 k 都搞离线

按边权增序排序，如果是询问则回答询问，

如果是加边则在 dfs 序线段树里单点修改，把边权加到远离树根的那个的点上，

用树剖在查询 lca 过程中统计重链和轻边的贡献，最后减去 lca 单点的贡献（ lca 的贡献来自于 lca 与其父亲相连的边）
边权树剖

MORE XOR

对于每次询问区间 $[i,j]$，考虑对区间里每个数算贡献，设该数位置为 pos，则该数参与总异或次数为 $(pos-i+1)\times(j-pos+1)$ 次，且当仅当 $(pos-i+1)\times(j-pos+1)\%2==1$ 该数有真贡献，观察发现 $pos-i+1$ 随 pos 增减奇偶性交替，$j-pos+1$ 同理，那么就是说 pos 位置 和 $pos-4$ 位置的异或次数的奇偶性相同，即循环节为 4 (打表也能发现)，所以只需考虑询问区间前四个数参与异或次数的奇偶性就能得到整个询问区间所有数异或次数的奇偶性，$\mathcal{O}(n)$ 预处理前缀异或和，每次询问 $\mathcal{O}(1)$ 复杂度。

qiqi'tree

因为数据量量给的比比较小小，所以我们可以直接搜索得到我们想要的答案。也就是说在初始的第一一年年，

我们特判掉是否第一一个枝干干就已经被剪掉了了，然后每次我们增广广每个方方向，分别求答案。一一个简单

的优化是:因为要求的精度是 10^{-6}，而而且每年年长长度都是前一一年年的 $1/4$，所以长长度的减少很快，很快就 都能达到 10^{-8} 的长长度。

Subsequence

直接暴力模拟就行了，遍历 T 串的每个字符，在 S 串中寻找有没有，如果没有的话就返回 false.