GAME

题目来源：CODE FESTIVAL 2016 Final H Tokaido

简略题意

有两个人和排成一行的 n 只鸽子，每只鸽子都有一个非负权值，人在鸽子的左边。 每一轮最左边的人会选一只右边的还没有选过的鸽子，然后移动到鸽子的位置，如果这样的鸽子不存在则游戏结束。 每个人的得分是自己选的鸽子的权值和与对方选的鸽子的权值和之差，且每个人都想最大化自己的得分。 前 $n-1$ 只鸽子的权值已经确定，现在有 q 个询问，每次给出第 n 只鸽子的权值，要求计算出第一个人的得分。

$2 \leq n \leq 200000$，$q \leq 200000$, 前 $n-1$ 只鸽子的权值之和不超过 10^6，第 n 只鸽子的权值不超过 10^9。

子任务 1

咕咕咕

子任务 2

在这个子任务中，$q=1$。

可以发现一些性质（假设没有人正在移动）：

1. 两个人之间的鸽子都是还没有选过的
2. 权值非负，所以如果两个人之间还存在鸽子，那么最左边的人肯定会把这些鸽子全部选完

所以可以发现，游戏规则等价于每一轮最左边的人移动到右边的某个位置，然后另外一个人把他们之间的所有鸽子全部选完。 这样，每一轮结束后两个人肯定在相邻的位置上，并且两个人的左右关系和原来相反。

设 a_i 为第 i 只鸽子的权值，f_i 为只考虑第 $i+1$ 到 第 n 只鸽子时第一个人的得分，可以得到方程

$$f_i = \max_{j=i+1}^{n} \left(a_j - f_j - \sum_{k=i+1}^{j-1} a_k \right)$$

边界条件是

$$f_n = 0$$

那么答案就是 f_0。

这样直接转移是 $O\left(n^3\right)$ 或 $O\left(n^2\right)$ 的。

注意到当 $i < n$ 时可以把 $\max\limits_{j=i+1}^{n}$ 拆开，得到

$$f_i=\max\left(a_{i+1}-f_{i+1},\max_{j=i+2}^{n} a_j-f_j-\sum_{k=i+1}^{j-1} a_k\right)=\max(a_{i+1}-f_{i+1},f_{i+1}-a_{i+1})$$

所以我们就得到了一个 $O(n)$ 做法。

子任务 3

在这个子任务中，a_n 在 $\left[0,10^9\right]$ 中等概率随机生成。

注意到当 $f_{i+1}\geq a_{i+1}$ 时 $f_i=f_{i+1}-a_{i+1}$，且 $f_{n-1}=a_n$，所以如果 $a_n\geq\sum_{i=1}^{n-1} a_i$ 则答案为 $a_n-\sum_{i=1}^{n-1} a_i$。因为 $\sum_{i=1}^{n-1} a_i\leq 10^6$，所以 $a_n<\sum_{i=1}^{n-1} a_i$ 的概率不超过 10^{-3}。而 $10^{-3}qn\leq 40000000$，可以接受。这样，只要特判一下就可以通过子任务 3。

子任务 4

在这个子任务中，a_1 到 a_{n-1} 的生成方式为：首先全部初始化为 0，然后重复 10^6 次，每次等概率挑选一个 a 并加上 1。

注意到这样生成的数通常较小。考虑一个动态规划：设 d_{ij} 为当 $f_i=j$ 时 f_0 的值，那么可以得到

$$d_{ij}=d_{i-1,|j-a_i|}$$

边界条件是

$$d_{0j}=j$$

我们先对于 $0\leq i\leq n$，$0\leq j<\max_{i=1}^{n-1} a_i$ 的 i，j 计算出 d_{ij}。然后对每个询问，首先特判掉子任务 3 中提到的情况，然后找到最小的满足 $a_n\geq\sum_{i=k}^{n-1} a_i$ 的 k（$k=n$ 满足条件，$k=1$ 不满足条件，所以最小的 k 一定存在）。这时，我们有

$$a_n-\sum_{i=k-1}^{n-1} a_i<0\leq a_n-\sum_{i=k}^{n-1} a_i$$

所以

$$0\leq f_{k-1}=a_n-\sum_{i=k}^{n-1} a_i<a_{k-1}$$

这时只要利用预处理好的 d 即可。

如果找最小的 k 时使用二分查找，时间复杂度可以为 $O\left(n\max_{i=1}^{n-1} a_i+q\log n\right)$。如果对于所有满足 $a_n<\sum_{i=1}^{n-1} a_i$ 的 a_n 预处理出 k，时间复杂度可以是 $O\left(n\max_{i=1}^{n-1} a_i+q\right)$。

子任务 5

这个子任务没有任何附加约束。

考虑子任务 4 中的动态规划：当 $j \ge a_i$ 时 $d_{ij} = d_{i-1,j-a_i}$，否则 $d_{ij} = d_{i-1,a_i-j}$。可以发现，d_i 实际上是将 d_{i-1} 平移了 a_i 的距离，然后再在最前面对称地补上 a_i 个数。这样只要用一个支持前端插入的数据结构（deque 或者逆序的 vector）维护即可。这样我们就可以在 $O\left(\sum_{i=1}^{n-1} a_i\right)$ 的时间里对于所有满足 $0 \le j < \sum_{i=1}^{n-1} a_i$ 的 j 求出 $d_{n-1,j}$。总的时间复杂度为 $O\left(\sum_{i=1}^{n-1} a_i + q\right)$

另一个做法

也可以把子任务 4 中的动态规划的 d_i 的状态数从 $\max_{i=1}^{n-1} a_i$ 优化到 a_i 然后利用二分/其他方法来转移，这里就不写了。预处理时间复杂度为 $O\left(\sum_{i=1}^{n-1} a_i\right)$ 或 $O\left(\sum_{i=1}^{n-1} a_i \log n\right)$，带 $\log$ 的版本不一定可以通过。

总结

题出的好！覆盖知识点广，题目有着切合实际的背景，解法比较自然。