NOIP 模拟赛 2

数字重组 (digit)

暴力：搜索，或者 dp（虽然没给 dp 分数，但毕竟 T1 不该考 dp）。

正解：发现 $a_i \geq 1$，同时 X 是 $B-1$ 的倍数当且仅当 X 在 B 进制下数位和是 $B-1$ 的倍数，所以计算给出所有数字的和，设为 s，则去掉一个 $s \bmod (B-1)$，剩下的数贪心地从大到小放即可，对于每个询问还需要二分查询。时间复杂度：$O(B+q\log B)$。

收集灵水 (collect)

暴力：模拟，$O(nm)$，可以直接得 60pts。

$a_i \le 10$：只需要考虑 $1 \sim 6$ 和 $n-5 \sim n$，其他的每次去一定都收集满了，这样你就 70pts 了。

正解：每次如果在 $2n-1$ 中的前 n 天喝到灵水，那么称为 A类；后 $n-1$ 天称为 B 类。

可以发现所有盆子会分成以下三类

- 一直是 A 类；
- 第一次是 A 类，接下来一直是 B 类；
- 一直是 A 类然后 B 类循环往复。

再对于同一类的盆子，按照 $\lfloor \frac{a_i}{2n-1} \rfloor$ 放在一块做（这些盆子对答案的贡献相同），直接暴力找到所有需要修改的答案贡献上去即可。

时间复杂度：$O(n \ln n)$。

商店购物 (shop)

算法零

爆搜，时间复杂度：$O(\text{poly}(nm))$。期望得分：5~15pts。

算法一

一些奇奇怪怪的 $O(\text{poly}(n,m))$ 的做法或常数较大的其他做法，期望得分：20~60pts

算法二

现在可以先让小 A 选出他依次准备要买的物品给小 B，接着小 B 找一个最优的结束时刻让小 A 收益最小，具体地：

- 小 A 选取 $m+1$ 个物品 $i_1,i_2,\cdots,i_{m+1}$。
- 小 B 求出最小收益：

$$\min_{j=1}^{m+1}\{B_{i_j}-\sum_{k=1}^{j}A_{i_k}\}$$

容易发现，这样与原来的结果没有区别。

显然，对于 $x<y$，如果 $B_{i_x}>B_{i_y}$，交换他们一定更优，所以 B_{i_j} 是不降的。

考虑将原来的物品按照 B_i 升序排序。

接着二分答案，现在只需检验小 A 能否到达某个收益 mid。

设 $f_{p,q}$ 表示考虑了前 p 个物品，选中的有 q 个，每个 $j\le q$ 都满足 $B_{i_j}-\sum_{k=1}^{j}A_{i_k}\ge mid$，此时最小的 $\sum_{j=1}^{q}A_{i_q}$。

转移如下：

$$\begin{aligned}
&f_{0,0}=0\\
&f_{p,q}\leftarrow f_{p-1,q}\\
&f_{p,q}\leftarrow f_{p-1,q-1}+A_p \qquad (B_p-A_p-f_{p-1,q-1}\ge mid)
\end{aligned}$$

时间复杂度：$O(nm\log V)$。期望得分：85pts。

算法三

算法二的 dp 可以使用数据结构优化，例如线段树之类的，但常数较大，可能无法通过。

算法四

接着算法二的 dp 之前的思路。

考虑反悔贪心，p 从 1 到 n 遍历，维护前 p 个物品想要有 $\geq mid$ 的收益，最多能抗下小 B 几次摧毁。

每次尝试加入一个物品，如果加入不了，则可以把之前的某个物品的 A_x 替换为当前的 A_i（如果 $A_x > A_i$ 的话）。

使用优先级队列维护选中的物品的 A 的集合即可。

时间复杂度：$O(n \log n \log V)$。

跳跃游戏 (jumpgame)

直接写出 dp 转移（和题中给的略有不同，结果不变）：

$$f_i = \max(f_{i-1}, f_{i-k} + a_i)$$

暴力做有 17pts，再写一个 $k = 1$ 的输出 $\sum_{i=1}^{n} a_i$ 的东西，可以获得 29pts。

正解做法：先考虑 $k \le 2.5 \times 10^5$ 怎么做，发现可以考虑用一个 $\bmod k$ 的循环数组维护所有 f，同时，对于 $i \in [l, r]$，若 a_i 相同，考虑如何快速整体转移。发现用线段树就能够维护了。

> *Q：为什么没有 $k \le 2.5 \times 10^5$ 的部分分?*
>
> *A：搬的数据，懒得造了。*

然后优化到更大 k 的情况，你当然可以选择用动态开点线段树做，但是不太清楚能不能过。

想要做到空间线性，加一个离散化就行了，对于所有可能出现的转移区间 $[l, r]$，在循环数组上标记断点 $l \bmod k, (r+1) \bmod k$，然后，你会发现同一段内部的 f 值，一定是相同的。时间复杂度：$O(q \log q)$，空间复杂度：$O(q)$。