2025-2026 学年大连理工大学开发区校区程序设计竞赛新生赛-题解

出题人赛前预测难度

- Easy：D，I，E
- Medium Easy：J，A，F
- Medium：G，C，B
- Medium Hard：K，H

A 集合栈计算机 II

本题要求构造一个集合 T，使得其括号数 $f(T)$ 等于给定的 x 。

一些关键性质：

- **空集的括号数为 1**：$f(\{\}) = 1$ 。
- **ADD 操作的加法性**：将集合 A 添加到集合 B 中，如果 A 不在 B 中，新集合的括号数等于两者之和：$f(B \cup \{A\}) = f(B) + f(A)$ 。
- **倍增技巧**：对集合 S 执行 `DUP` 然后 `ADD`，得到 $S \cup \{S\}$，括号数翻倍：$f(S \cup \{S\}) = 2 \times f(S)$ 。

解法一：递归构造

- $x = 1$
  - 执行 `PUSH`，得到空集。
- $x = 2k$
  - 调用 $\mathrm{solve}(k)$ 构造括号数为 k 的集合。
  - `DUP` 和 `ADD` 将括号数翻倍到 $2k$ 。
- $x = 2k + 1$
  - 先 `PUSH` 一个空集作为容器。
  - 递归调用 $\mathrm{solve}(2k)$ 构造括号数为 $2k$ 的集合。
  - 再执行 `ADD` 将它放入容器，括号数变为 $1 + 2k$ 。

每个二进制位最多用 4 条指令，最多需要 $1 + 4 \times 62 = 249$ 条。

解法二：预处理 + 二进制组合

这个是出题人想到的第一个解法。也许不容易想到，但是很优美。

先预处理出 2 的幂次集合，再按二进制位组合。

1. 预处理

- 执行 PUSH 得到 S_0，括号数为 1。
- 循环 62 次，每次 DUP, DUP, ADD，得到括号数为 2^i 的集合 S_i。
- 结束后栈中有 $S_0, S_1, \ldots, S_{62}$。

2. 关键性质

- $S_0 \subset S_1 \subset S_2 \subset \cdots \subset S_{62}$
- $S_j \in S_i \ (j < i)$
- $S_i \cap S_{i-1} = S_{i-1}$
- $S_i \cup S_{i-1} = S_i$

3. 组合

从高到低遍历 x 的二进制位，设栈顶为 T，次栈顶为 S_{i-1}

- 第 i 位为 0 且 $2^i > x$
  - 执行 `INTERSECT`，相当于丢弃 T，保留 S_{i-1}。
- 第 i 位为 1 且第 $i-1$ 位为 0
  - 执行 `UNION`，相当于丢弃 S_{i-1}，保留 T。
- 第 i 位为 1 且第 $i-1$ 位为 1
  - 执行 `ADD`，得到 $S_{i-1} \cup \{T\}$，括号数为 $f(T) + 2^{i-1}$。

预处理 $1 + 3 \times 62$ 条，组合 62 条，共 249 条。

B 机械猫的流浪

- 小猫在每个位置只有两种选择：等待或前进
- $\sum a_i$ 可能很大，但总路程限制了有效时间与能量值上界

状态设计

设 $f(i,t)$ 为在位置 i、时间 t 时的最大剩余能量。

初始状态：$f(1,0)=E$ 。

目标：找到最小的 t 使得 $f(n+1,t)>0$ 。

状态转移

在位置 i、时间 t 有两种决策：

- 原地等待：消耗 1 格能量
  - 若 $t = l_i$，可获得 a_i 格补给

$$f(i, t+1) \leftarrow \max(f(i, t+1),\ f(i, t) - 1 + a_i)$$

  - 否则无法获得补给

$$f(i, t+1) \leftarrow \max(f(i, t+1),\ f(i, t) - 1)$$

- 前进一步：消耗 2 格能量
  - 若 $t \in [l_{i+1}, r_{i+1}]$，可获得 a_{i+1} 格补给

    $$f(i+1, t+1) \leftarrow \max(f(i+1, t+1),\ f(i, t) - 2 + a_{i+1})$$

  - 否则无法获得补给

    $$f(i+1, t+1) \leftarrow \max(f(i+1, t+1),\ f(i, t) - 2)$$

注意：只有当 $f(i, t) > 0$ 时才能进行转移。

复杂度分析

小猫走 n 步至少需要时间 n，最坏情况下需要等待所有补给站开放。

由于 $r_i \le 10^4$，时间上界约为 $\max(r_i) + n$ 。

小猫最多走 n 步，消耗至多 $2n + 2$ 格能量。

因此能量值上界约为 $\min(a_i, 2n + 2)$ 。

设时间上界为 $T = \max(r_i) + n$，总时间复杂度 $O(n \cdot T)$，空间复杂度 $O(n \cdot T)$ 。

C 跟我的漫游者说去吧

本题给定期望暴击率 p，求暴击系数 c 。

设第 i 次攻击的暴击概率为 $r_i = \min(1, ic)$ 。

注意到 p 关于 c 单调递增，可以二分答案求解。

设暴击间隔为随机变量 X，表示两次暴击之间的攻击次数。由于稳态下暴击事件周期性出现，期望暴击率为

$$p = \frac{1}{EX}$$

设攻击前 $i-1$ 次均未暴击、第 i 次恰好暴击的概率为 $P\big(X=i\big)$，则

$$P(X=i) = r_i \cdot \prod_{j=1}^{i-1}(1-r_j)$$

期望暴击间隔

$$EX = \sum_{i=1}^{\infty} i \cdot P(X = i)$$

当 $ic \geq 1$ 时，$r_i = 1$，必定暴击，因此求和有限。

式子看起来很复杂，但是实际上代码非常简单。

二分次数约 $\log_2(10^{10}) \approx 34$ 次，并且可以适当增加次数以提高精度。每次求期望需 $O(\frac{1}{c})$ 次运算。

D 无人能证的猜想

题目保证答案存在且小于 1000，直接模拟即可，循环不会超过 1000 次。

E Excel 的列编号

初看像是 26 进制转换，但 Excel 列编号没有“零”这个概念：

- A 对应 1，而非 0
- Z 对应 26，下一个是 AA（27）而非 BA

这是一种“无零”的 26 进制，需要特殊处理。

每次处理时，先将 a 减 1，使其从 0 开始：

- 减 1 后，$a \bmod 26$ 范围的 $0 \sim 25$，恰好对应 A $\sim$ Z
- $\lfloor a/26 \rfloor$ 为剩余高位
- 可以递归处理，也可以循环处理后逆序输出

每次递归 a 减少为原来的 $1/26$，递归深度为 $O(\log_{26} a)$ 。

对于 $a \leq 10^{18}$，最多递归约 13 层。

总时间复杂度 $O(t \log a)$

F 送熏肉

注意到 k 的大小和 n 同阶，但是 k 全局不会变化。

于是我们可以利用 k 的全局不变性来重构序列——对每个长度为 k 的子区间求和，把求和结果作为新序列的项。显然，新序列总共有 $n-k+1$ 项。使用双指针或者前缀和预处理。

原序列上的单点修改等价于重构序列上的区间修改，具体来说是区间修改 $\left[\max(1,x-k+1),\min(n-k+1,x)\right]$ 区间的数，让它们全部加上 $y-a_x$，这个显然可以用线段树的区间修改来完成。

原序列上的区间询问等价于重构序列上的询问全局最大值。根据线段树的特性，我们维护一个区间 $\max$ 的属性，直接输出根节点的属性即可。

G 远交近攻

用并查集维护地块是否属于同一部落；对于相邻关系，并查集本身难以维护，需要记录更多的信息。

比如说，在并查集的祖先节点上记录其所有的地块编号。

并查集需要不断合并更新，对每个部落都记录所有的地块编号。

需要使用启发式合并。每次遍历地块数量较少的部落，将其合并到地块数量较多的部落中；

H Silly Tree

https://zhuanlan.zhihu.com/p/19934491253337307 29

I 奶龙塔

观察到如果有两只奶龙的体积相等，它们一定无法堆在一起；

如果所有奶龙的体积都不相同，则它们能在排序之后堆成一座奶龙塔。

检查序列中是否存在相同体积值即可。

J 小猫钓鱼

模拟题，需要理解规则后按流程实现。

一些核心观察如下：

- 在某一回合结束后，池塘中不会有两张相同点数的牌。
- 如果发生钓鱼，鱼的右端点一定是最右侧的牌。
- 池塘中最多有 $13+1$ 张牌。

因此，每次放入新牌后，扫描池塘，找到点数相同的牌即可确定“鱼”的范围。

池塘可以用数组模拟：由于钓鱼只会发生在末尾，不需要复杂的数据结构。

计分的逻辑非常好写：若“鱼”的长度为 l，则不同的点数一定有 $l-1$ 种，只需要统计不同的花色数即可。

K 用户名长度应在 5 位到 12 位之间

选取 s 的前缀 a 和后缀 b，满足 a 的末字符等于 b 的首字符，且 $|a| + |b| \in [l, r]$。

求本质不同的拼接结果 $t = a + b$ 的数量。

设 $|a| = i$，$|b| = j$，则 t 的长度为 $len = i + j$。不同 len 的字符串一定不可能相同。

对于固定的 len，可以枚举前缀长度 i，合法性是很容易验证的，剩下的就是去除重复字符串了。

对于固定的 len，从小到大枚举前缀 i 的长度，当前缀长度从 i 变成 $i+1$ 时，t 发生了什么变化?

答：只有 t 的第 $i+1$ 位从 $s[n-j+1]$ 变成了 $s[i+1]$，如果这两个字符相同，则字符串 t 不变。

如果字符串 t 不变，我们就可以一直 $i \leftarrow i+1$，在这个过程中顺便记录是否有至少一次合法拼接，如果有，答案加一。

复杂度 $O(n^2)$