Problem A. Find Your Miku III

根据题目，只需要输出一个符合条件的数字，所以考虑打表解决。首先观察到条件 1，$39^3 \approx 6 \times 10^4, 39^9 \approx 2 \times 10^{14}$，想要穷尽所有数字显然是不可能的。

但是观察到条件 4，要求最小的数，所以我们可以从左端点开始搜，按照条件 2 和条件 3 的要求用短除法和循环判断数字是否满足要求。实际上很快就能搜索到答案 142857，这是一个很有名的循环数。

时间复杂度 O(n)。

Problem B. The question posed by Hirasawa Yui

根据题意，数组中的所有数都是正数，而题目要求找长度不超过 m 的子数组，要求其加和最大，所以贪心易得就是要找 k 个长度为 m 的子数组。可以使用滑动窗口（也称尺取法）遍历所有长度为 m 的子数组，维护其加和，并设置变量（因为 k 最大为 2）来记录所有的子数组加和的最大值和次大值。

本题有一个坑点，就是当 m=n 且 k=2 时，需要取到的是一个长度为 n 的子数组，一个是长度为 n-1 的子数组，这里需要特殊处理一下。

当然也可以使用大小恒定为 k 的优先队列（大根堆）来保存最大和次大的子数组加和。

时间复杂度 O(n)。

Problem C. Name

这是一道模拟题，本身没有思维难度，但是需要细心 coding。

首先设置两个长度不小于 2000 的数组来接收读入，然后每 7 位转化为一个十进制的数，保存后以字符形式输出即可。

时间复杂度 O(n)。

Problem D && G. Black piano key(easy && hard version)

本场比赛压轴题，给出三种做法：暴力枚举/平衡树/线段树。

暴力枚举：把假设仅有这一组变成黑键时，两个黑键的位置放入数组中。然后对于每个查询，如果当前存放黑键位置的数组的黑键位置小于当前查询的位置，就将查询的目标位置减一，直到当前位置大于查询位置，然后计算这个查询位置的对应的组是第几组。

因为要维护存放黑键位置的数组的升序，所以每次插入以后都需要排序，这里使用插入排序的话，总体时间复杂度为 O(n^2)，标程使用的是 sort 函数，因此时间复杂度为 O($n^2 logn$)。

平衡树：建立一棵平衡树，每次插入的时候插入两个白键的真实位置，查询时二分查找对于询问第 j 个黑键，若其真实位置在 x 是否合法。对于每个 x，用平衡树查询在 x 前有几个白键已经变成了黑键，然后计算一波后返回对于真实位置 x，是第几个黑键。

时间复杂度 O($n(logn \times logASK)$)，其中 ASK 为询问的数字大小。

线段树：将所有查询离线后离散化，然后在离散化后的下标上建立**区间更改·单点查询**的线段树，接着遍历操作。对于某个询问 x，设其离散化后的值为 id，则在线段树上从 id 到最后的所有点都+2，代表白键的影响；对于查询 x，二分搜索它对应的询问的位置，然后通过公式得出当前查询的黑键的所在组。

时间复杂度 O(max(nlogn,mlogn))。

其中暴力枚举的方法可以通过 easy version，后两种方法可以通过 hard version。

Problem E. Enterdawn's hotpot

01 背包模板题，唯一需要改动的地方是最后统计答案，要在装的尽可能满的情况下的花费，时间复杂度 O(n^2)。

但是观察数据范围，n 最大只有 20，而每一种菜都有选或者不选两种可能，则 dfs 的时间复杂度为 O(2^n)，跑满大概是 10^6 级别，也能通过。

Problem F. Solitary number

"若一个数是大于 1 的正整数，且只能被 1 和它本身整除，则它就是一个'孤独的数'"。显然孤独的数就是质数。

对于一个数字 x，我们有一种 O($\sqrt{x}$) 的时间复杂度判断其是否为质数的算法，详情见标程。

也有其他更快的素数判定方法，但正确性不能保证，如米勒-罗宾素数判定算法。也可以使用各种筛来打表判断，都能通过。

标程的时间复杂度为 O($L\sqrt{L}$)。

Problem H. The easiest problem in this contest

签到题，通过第一个公式计算出项数后，带入第二个公式就能算得该等差数组的和，时间复杂度 O(1)。

Problem I. ii Genn ki

签到题，输出 a+b 即可，时间复杂度 O(1)。