

# C O D E F O R C E S   R E V I E W

# CONTENTS

2013D - 1900 . . . . .	I
2014H - 1900 . . . . .	I
2035D - 1800 . . . . .	I
2009GI - 1900 . . . . .	2
2002DI, 2021C2 - 1900 . . . . .	2

## .1 2013D - 1900

**Problem 1.** <sup>1</sup> Given an array  $A$  and an operator  $p$  that for each  $a_i$  and  $a_{i+1}$  where  $i \geq 1$ ,  $p$  may decrease  $|a_i - a_{i+1}|$ .  $p$  can be performed by infinite times and find the minimum  $|\max_{i=1}^n(a_i) - \min_{i=1}^n(a_i)|$

**Solution 1.** Suppose  $b_i$  is the result of performing  $p$  on  $a_i$  given the prefix of array  $A$   $\{a_1, a_2, \dots, a_i\}$ . Create a stack  $S$  to load these  $b_i$  and the count of it  $c_i$ . Then  $\{a_1, a_2, \dots, a_i\}$  is stored in  $S$  as  $(b_1, c_1), (b_2, c_2), \dots, (b_m, c_m)$ . And we keep the pairs in ascending order of  $b_i$ . For each new  $a_{i+1}$ , merge it to the top if  $a_{i+1} < b_m$  and then merge the top downwards until  $b_k > b_{k-1}$ . This way, each  $a_i$  is loaded in the array for once, the time of merging until  $a_i$  is at most  $i$ , so the time complexity is  $O(n)$ .

## .2 2014H - 1900

**Problem 2.** <sup>2</sup> Given an array  $A$ , check for each  $a_i$ , if there exists  $A_m = \{a_{m_1}, a_{m_2}, \dots, a_{m_j}\}$  s.t.  $a_i \in A_m$  and  $j$  is even.

**Solution 2.** If the size of  $A$  is small, for instance,  $1e6$ , then first hashing the  $A$  in a much bigger set, for instance,  $\{1, 2, \dots, 2^{64}\}$ , and check if the xor sum of the array is 0. The hash is very important, without which, several different numbers can also get xor sum 0 (e.g.,  $\{1, 2, 3\}$ ). The possibility of reaching such bad situation after hashing is  $\frac{1}{2^{64}}$ .

## .3 2035D - 1800

**Problem 3.** <sup>3</sup> Given an array  $A$  and an operator  $p$  that for each  $a_i$  and  $a_j$  that  $i < j$ ,  $p$  can update  $a_i$  to  $a_i \gg 1$  and  $a_j$  to  $a_j \ll 1$ .  $p$  can

<sup>1</sup><https://codeforces.com/problemset/problem/2013/D>

<sup>2</sup><https://codeforces.com/problemset/problem/2014/H>

<sup>3</sup><https://codeforces.com/problemset/problem/2035/D>

be performed by infinite times and find the maximum sum of all the prefixes of  $A$ .

**Solution 3.** A little similar to 2013D. In 2013D, we store  $b_i$  and  $c_i$ , the information of  $a_i$  after  $p$  in a stack  $S$ . This approach is applicable in this problem too. In this problem  $b_i = \min\{b_i^j | b_i^j \ll c = a_i\}$ , and  $c_i$  is the sum of all  $c_k$  that  $k < i$  that can reach the maximum prefix sum from 1 to  $i$ . This sum is obtained by merging from the top of the stack downwards. If  $c_k (k < i)$  is added to  $c_i$ , then  $b_k$  is popped out from the stack and added to the final sum. After the merging is terminated, we push  $(b_i, c_i)$  onto  $S$ . Each  $(b_i, c_i)$  pair is at most pushed to  $S$  by once and popped by once. So the time complexity is  $O(n)$

#### .4 2009G1 - 1900

**Problem 4.** <sup>4</sup> Given an array  $A$  and an operator  $op$  that can change any element  $a_i$  to any another value. Find the minimum step required to make  $A$  a consecutive array in which  $a_{i+1} - a_i = 1$  for any  $i$ .

**Solution 4.** Create an array  $B$  that  $b_i = a_i - i$ . Count the frequency  $f_i$  of each  $b_i$  and the answer is  $A.size() - \max(f_i)$ .

#### .5 2002D1, 2021C2 - 1900

**Problem 5.** <sup>56</sup> Continuously query about a property  $P$  with an update before each query.

**Solution 5.** Find another easy-to-maintain property  $P'$  s.t. if  $P'$  is satisfied then  $P$  is satisfied. The time complexity of checking for  $P'$  after each update is  $O(1)$  or  $O(\log n)$ .

---

<sup>4</sup><https://codeforces.com/problemset/problem/2009/G1>

<sup>5</sup><https://codeforces.com/problemset/problem/2002/D1>

<sup>6</sup><https://codeforces.com/problemset/problem/2021/C2>