计算学部寒假十大打卡活动——“龙舞编程新春会”编程打卡

> 2024年2月1日
>
> LCP 24. 数字游戏

一、题目

题目描述

小扣在秋日市集入口处发现了一个数字游戏。主办方共有 `N` 个计数器，计数器编号为 `0 ~ N-1`。每个计数器上分别显示了一个数字，小扣按计数器编号升序将所显示的数字记于数组 `nums`。每个计数器上有两个按钮，分别可以实现将显示数字加一或减一。小扣每一次操作可以选择一个计数器，按下加一或减一按钮。

主办方请小扣回答出一个长度为 `N` 的数组，第 `i` 个元素(0 <= i < N)表示将 `0~i` 号计数器 **初始** 所示数字操作成满足所有条件 `nums[a]+1 == nums[a+1],(0 <= a < i)` 的最小操作数。回答正确方可进入秋日市集。

由于答案可能很大，请将每个最小操作数对 `1,000,000,007` 取余。

样例 1：

输入：

```
[3,4,5,1,6,7]
```

输出

```
[0,0,0,5,6,7]
```

样例 2：

输入：

```
[1,2,3,4,5]
```

输出

```
[0,0,0,0,0]
```

样例 3：

输入：

```
[1,1,1,2,3,4]
```

输出

```
[0,1,2,3,3,3]
```

数据范围

- `1 <= nums.length <= 10^5`
- `1 <= nums[i] <= 10^3`

二、解答

思路：数学原理

设nums数组前m个数为$a_0, a_1, a_2, \ldots, a_m$,

令变换后的前m个数为$x, x+1, x+2, \ldots, x+m$,

则操作次数为

$$\begin{aligned} &|a_0 - x| + |a_1 - (x+1)| + |a_2 - (x+2)| + \ldots + |a_m - (x+m)| \\ =\ &|a_0 - x| + |(a_1 - 1) - x| + |(a_2 - 2) - x| + \ldots + |(a_m - m) - x| \end{aligned}$$

令$b_i = a_i - i$

则操作次数变成$|b_0 - x| + |b_1 - x| + |b_2 - x| + \ldots + |b_m - x|$

那么问题就变成了一个初中数学拓展题：求数轴上一个点到其他点的最小距离和

推理过程~

结论如下：

> 设数轴上有n个定点，当n为偶数时，
> 到这 n 个定点的距离之和最小的点在第$\frac{n}{2}\sim\frac{n}{2}+1$个点之间 (含两个端点);当n为奇数时，到这 n 个定点的距离之和最小的点在第$\frac{n+1}{2}$个点处.

总结：求中位数

```
class Solution {
public:
    vector<int> numsGame(vector<int>& nums) {
        vector<int>ans(nums.size());
        priority_queue<int>littles;//大根堆，维护较小的一半
        priority_queue<int,vector<int>,greater<int>> bigs;//小根堆，维护较大的一半
        int mod=1000000007,nlen=nums.size();
        long long lsum=0,bsum=0;
        for(int i=0;i<nlen;i++)
        {
            int x=nums[i]-i;
            if(i%2)//奇数个数，选中间那个数位中位数
            {
                if(!bigs.empty()&&x>bigs.top())
                {
                    bsum+=x;
                    bigs.push(x);
                    x=bigs.top();
                    bsum-=x;
                    bigs.pop();
                }
                littles.push(x);
                lsum+=x;
                ans[i]=(bsum-lsum)%mod;
```

```
            }
            else//偶数个数
            {
                if(!littles.empty()&&x<littles.top())
                {
                    lsum+=x;
                    littles.push(x);
                    x=littles.top();
                    lsum-=x;
                    littles.pop();
                }
                bigs.push(x);
                bsum+=x;
                ans[i]=(bsum-lsum-bigs.top())%mod;
            }
        }
        return ans;
    }

};
```

三、总结

1. 题目难点在于数学推理。代码实现的主体部分在于实时维护中位数。
2. 时间复杂度：$O(nlogn)$，其中 n为 $nums$ 的长度。每次操作堆的时间复杂度是 $O(logn)$。
   空间复杂度：$O(n)$。