滑动窗口【抓住本质】1838. 最高频元素的频数

题目描述：

1838. 最高频元素的频数

难度 中等 93

元素的 **频数** 是该元素在一个数组中出现的次数。

给你一个整数数组 `nums` 和一个整数 `k` 。在一步操作中，你可以选择 `nums` 的一个下标，并将该下标对应元素的值增加 `1` 。

执行最多 `k` 次操作后，返回数组中最高频元素的 **最大可能频数** 。

示例 1：

```
输入：nums = [1,2,4], k = 5
输出：3
解释：对第一个元素执行 3 次递增操作，对第二个元素执 2 次递增操作，此时 nums = [4,4,4] 。
4 是数组中最高频元素，频数是 3 。
```

我的解法：使用队列

```
class Solution {
public:
    int maxFrequency(vector<int>& nums, int k) {
        //采用队列的数据结构
        //先排个序
        sort(nums.begin(),nums.end());
        queue<int>que;
```

```
        long long int res=0,ans=0,num=0,sum=0;
        size_t i=0,n=nums.size();
        while(i!=n){
            sum+=nums[i];
            que.push(nums[i++]);
            ++num;
            if((num*que.back()-sum)<=k){
                ans++;
                //res=max(res,ans);
            }else{
                res=max(res,ans);
                while((num*que.back()-sum)>k&&!que.empty()){
                    sum-=que.front();
                    que.pop();
                    --num;
                }
                ans=que.size();
            }
        }
        res=max(res,ans);
        return res;
    }
};
```

滑动窗口简洁代码：

```
class Solution {
public:
    int maxFrequency(vector<int>& nums, int k) {
        sort(nums.begin(), nums.end());
        int n = nums.size();
        long long total = 0;
        int l = 0, res = 1;
        for (int r = 1; r < n; ++r) {
            total += (long long)(nums[r] - nums[r - 1]) * (r - l);
            while (total > k) {
                total -= nums[r] - nums[l];
                ++l;
            }
            res = max(res, r - l + 1);
        }
        return res;
    }
};
```