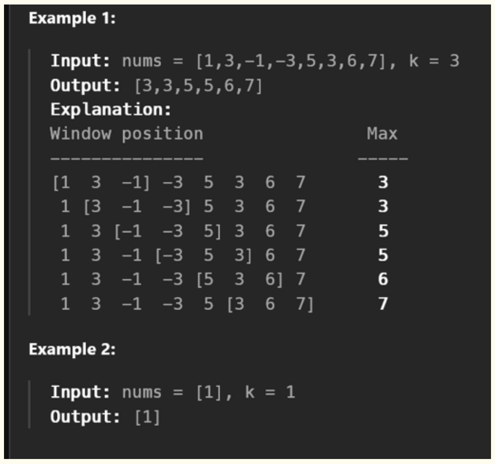
239. Sliding Window Maximum



Example 1:

```
Input: nums = [1,3,-1,-3,5,3,6,7], k = 3
Output: [3,3,5,5,6,7]
Explanation:
Window position                Max
---------------               -----
[1  3  -1] -3  5  3  6  7       3
 1 [3  -1  -3] 5  3  6  7       3
 1  3 [-1  -3  5] 3  6  7       5
 1  3  -1 [-3  5  3] 6  7       5
 1  3  -1  -3 [5  3  6] 7       6
 1  3  -1  -3  5 [3  6  7]      7
```

Example 2:

```
Input: nums = [1], k = 1
Output: [1]
```

1 -> 3 -> -1

3 -> -1

1 -> 3 -> -1 -> -3

× +

3 -> -1 -> -3

1 -> 3 -> -1 -> -3 -> 5

× +

3 -> ~~-1~~ -> ~~-3~~ -> 5

1 -> 3 -> -1 -> -3 -> 5 -> 3

× +

5 -> 3

1 -> 3 -> -1 -> -3 -> 5 -> 3 -> 6

× +

~~5~~ -> ~~3~~ -> 6

1 -> 3 -> -1 -> -3 -> 5 -> 3 -> 6 -> 7

× +

~~6~~ -> 7

Use a deque to find the maximum of a window in O(1)

add/remove - O(1)

remove()

If leftmost has same value popleft

add():

```
while q and rightmost_elem < val:
    q.popright()

q.appendleft(x)
```