**PROBLEM STATEMENT**:

Given an array *nums*, there is a sliding window of size *k* which is moving from the very left of the array to the very right. You can only see the *k* numbers in the window. Each time the sliding window moves right by one position. Return the max sliding window.

**Follow up:**  
Could you solve it in linear time?

**Example:**

**Input:** *nums* = [1,3,-1,-3,5,3,6,7], and *k* = 3

**Output:** [3,3,5,5,6,7]

**MY CODE:**

vector<int> maxSlidingWindow(vector<int>& nums, int k) {

int n=nums.size();

if(n<=1)

return nums;

vector<int> ans;

deque<int> q(k);

int i;

for(i=0;i<k;i++)

{

while(!q.empty() && nums[i]>=nums[q.back()])

q.pop\_back();

q.push\_back(i);

}

ans.push\_back(nums[q.front()]);

for(;i<n;i++)

{

while(!q.empty() && q.front()<=i-k)

q.pop\_front();

while(!q.empty() && nums[i]>=nums[q.back()])

q.pop\_back();

q.push\_back(i);

ans.push\_back(nums[q.front()]);

}

return ans;

}