Question: https://leetcode.com/problems/maximum-points-you-can-obtain-from-cards/

The first approach which came to my mind was backtracking where we make a decision at each step that is either to choose the first element of array r the last and like this we continue till kth level and return the maximum value, but this approach had a T.C of O(2^k), so I got a TLE(Time Limit Exceeded).

The next approach is that of a sliding widow.

See we can choose k elements, the combinations can be like k first elements, k last elements, or 1element from first k-1 elements from last, or 2 elements from first and k-2 elements from last, and so on.

So n being the total number of elements there will always be a sliding window of n-k elements.

So we need to find the max of total sum-sliding window sum.

Code:  
class Solution {

public int maxScore(int[] cardPoints, int k) {

int windowStart=0, l=cardPoints.length, windowEnd=l-k-1, sum=0;

for(int i=windowEnd+1; i<l; i++){

sum+=cardPoints[i];

}

int max=sum;

windowEnd++;

windowStart++;

while(windowEnd<l&&windowEnd>0){

sum-=cardPoints[windowEnd];

sum+=cardPoints[windowStart-1];

max=Math.max(max, sum);

windowEnd++;

windowStart++;

}

return max;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>