There are several cards arranged in a row, and each card has an associated number of points. The points are given in the integer array cardPoints.

In one step, you can take one card from the beginning or from the end of the row. You have to take exactly k cards.

Your score is the sum of the points of the cards you have taken.

Given the integer array cardPoints and the integer k, return the *maximum score* you can obtain.

Example 1:

Input: cardPoints = [1,2,3,4,5,6,1], k = 3
Output: 12
Explanation: After the first step, your score will always be 1. However, choosing the rightmost card first will maximize your total score. The optimal strategy is to take the three cards on the right, giving a final score of 1 + 6 + 5 = 12.

Example 2:

Input: cardPoints = [2,2,2], k = 2
Output: 4
Explanation: Regardless of which two cards you take, your score will always be 4.

Example 3:

Input: cardPoints = [9,7,7,9,7,7,9], k = 7
Output: 55
Explanation: You have to take all the cards. Your score is the sum of points of all cards.

Constraints:

- **1 <= cardPoints.length <= 10^5**
- **1 <= cardPoints[i] <= 10^4**
- **1 <= k <= cardPoints.length**

Solution:

```
class Solution {
    public int maxScore(int[] cardPoints, int k) {
        int total=0,n=cardPoints.length;
        for(int i=0;i<n;i++)    total+=cardPoints[i];
        int limit=n-k,sum=0;
        for(int i=0;i<limit;i++)    sum+=cardPoints[i];
        int answer=total-sum;
        for(int i=limit;i<n;i++){
            sum-=cardPoints[i-limit];
            sum+=cardPoints[i];
            answer=Math.max(answer,total-sum);
        }
        return answer;
    }
}
```