# Day 14 / 100 :

## Topic - Array

### 1 Problem statement: Put marbles in bag (Hard)

You have k bags. You are given a 0-indexed integer array of weights, where weights[i] is the weight of the ith marble. You are also given the integer k.

Divide the marbles into the k bags according to the following rules:
No bag is empty.
If the ith marble and jth marble are in a bag, then all marbles with an index between the ith and jth indices should also be in that same bag.
If a bag consists of all the marbles with an index from i to j inclusively, then the cost of the bag is weights[i] + weights[j].
The score after distributing the marbles is the sum of the costs of all the k bags.

Return the difference between the maximum and minimum scores among marble distributions.

Example 1:

Input: weights = [1,3,5,1], k = 2
Output: 4
Explanation:
The distribution [1], [3, 5, 1] results in the minimal score of (1+1) + (3+1) = 6.
The distribution [1,3],[5,1], results in the maximal score of (1+3) + (5+1) = 10.
Thus, we return their difference 10 - 6 = 4.
Example 2:

Input: weights = [1, 3], k = 2
Output: 0
Explanation: The only distribution possible is [1], [3].
Since both the maximal and minimal score are the same, we return 0.

## Solutions :

**Approach 1 - Sorting**

**Intuition:**
First and last one is always considered in final answer. At all the points of partition, we are considering two consecutive elements(marbles).

**Approach:**
To divide the array into k subarrays, we need k-1 partitions. Thus we will traverse the array and put sum of consecutive elements in another array. Then sort it. We will take sum of first k-1 elements of new array to get minimum score possible and similarly the last k-1 elements to get the maximum sum possible.
We can also use the priority_queue() to reduce the time complexity.

**Complexity**
Time complexity:O(n∗log(n))
Space complexity:O(n)

```cpp
#define ll long long

class Solution {
public:
    long long putMarbles(vector<int>& weights, int k) {
        ll n=weights.size(), mn=0, mx=0;
        if(k==1 || k==n){return 0;}
        vector<ll> v;
        for(ll i=0; i<n-1; i++){
            v.push_back(weights[i]+weights[i+1]);
        }
        sort(v.begin(), v.end());
        for(ll i=0; i<k-1; i++){
            mn += v[i];
            mx += v[v.size()-i-1];
        }
        return mx - mn;
    }
};
```

## 2 Problem statement: Palindrome Number (Easy)

Given an integer x, return true if x is a palindrome, and false otherwise.

Example 1:
Input: x = 121
Output: true
Explanation: 121 reads as 121 from left to right and from right to left.

Example 2:
Input: x = -121
Output: false
Explanation: From left to right, it reads -121. From right to left, it becomes 121-. Therefore it is not a palindrome.

Example 3:
Input: x = 10
Output: false
Explanation: Reads 01 from right to left. Therefore, it is not a palindrome.

## Solutions :

### Approach 1 - Maths Approach

**Approach:**

1. Initialize sum with 0.
2. Initiate temp = n; //I use t.
3. First, find the last digit of the number using (%).
4. After it stores last digit in sum. (sum has to multiply by 10 in every iteration because we have to reverse it).
5. After that, delete last digit using (/) and process again same procedure again until temp is not equal to 0.
6. If the sum (Reverse) is exactly equal to the given number, then it is a palindrome number. Return 1.
7. If not equal, return 0.

**Complexity:**
Time complexity: **O(n)**
because we have used a loop, and we don't know the iteration of the loop, so time complacency is O(n).
Space complexity: **O(1)**
We are not using extra space to run the program, so space complexity is constant.

```
class Solution {
public:
    bool isPalindrome(int x) {
        long long int t = x;
        long long int sum = 0;
        if( t < 0) //Number is negative.
        {
            t  =  t + (2 * t);
        }
        while(t!=0)
        {
            int rem = t % 10;
            sum = (10 * sum) + rem;
            t/=10;
        }
        if(x == sum)
        {
            return 1;
        }
        return 0;
    }
};
```