Max Subarray Sum

↳ Kadane's Algorithm

//Problem Statement

Calculate maximum subarray sum

arr[] : (3, 2, -6, 8, 2, 9, 4)

↳ ans = 23

arr[] : (-3, 2, 4, -1, 3, -4, 3)

↳ ans = 8

//idea 1 :

generate all subarray sum's and find out maximum out of them

T.C : $O(n^2)$, SC : $O(n)$

TLE will come bro... learn this algo, we don't have other option 😭

// Kadane's Algorithm

Observation

① (+ve, +ve, +ve, +ve, +ve, +ve)

Sum

Our ans will be sum of all

② (-ve, -ve, -ve, +ve, +ve, +ve, +ve)

ans = sum of all positives

① (+ve, -ve, +ve, -ve, +ve, +ve, +ve)

If these sum < 0 then this is our answer

Algo :-

if (any time Sum become < 0)

Update Sum with Current index value

Otherwise add arr[present_Index] to Sum

Update Max on every iteration

```java
class Solution {
    public int maxSubArray(int[] nums) {
        int n = nums.length;

        int sum = 0;
        int max = Integer.MIN_VALUE;
        for(int i = 0;i<n;i++){
            sum+=nums[i];
            max = Math.max(sum,max);
            if(sum<0)sum = 0;
        }
        return max;
    }
}
```

increase Sum → Update the max

So make Sum to 0 and go right

(Case 3)

whenever the Sum becomes negative our answer will be on right side of the index

① (+ve, -ve, +ve, -ve, +ve, +ve, +ve)

If these Sum < 0

then this is our answer