Maximum Sum SubArray

Kadane's Algorithm.

Helps to solve maximum sum subarray in linear time - O(n)

idea is as follows.

At every index, we ask what is the maximum sum subarray that ends at this index?

once we find this answer for every index, the final answer will be max of those values

This can be acheived in O(n) as follows:

```
maxSumArray[n]
maxSumArray[0] = a[0]   // single element, so will be max

for i=1 to n
{
   // max sum at index i will be either element at index i or
   // sum of a[i] & maxSum at index i-1
   if a[i] > maxSumArray[i-1] + a[i]
       maxSumArray[i] = a[i]
   else
      maxSumArray[i] = maxSumArray[i-1] + a[i]
}
return max(maxSumArray)
```

Same can be accomplished with O(1) Space Complexity

```
maxSumTillNow = a[0]   //holds maxSum till current index
maxSum = a[0]   //holds overall maxSum of Contiguous Subarrays

for i=1 to n
{
    if a[i] > (maxSumTillNow + a[i])
        maxSumTillNow = a[i]
    else
        maxSumTillNow = a[i] + maxSumTillnow

    if maxSumTillNow > maxSum
        maxSum = maxSumTillNo w
}
return maxSum.
```