Maximum Subarray (Kadane's Algorithm) - Notes

This problem is known as the **Maximum Subarray Problem** and is commonly solved using **Kadane's Algorithm**. You are given an integer array `nums`. The task is to find the contiguous subarray (containing at least one number) that has the largest sum and return its sum. ### Key Idea (Kadane's Algorithm): - Maintain a running sum of the subarray (`sum`). - At each step, add the current element to `sum`. - If `sum` becomes negative, reset it to `0` (since a negative sum would reduce the maximum for future subarrays). - Keep track of the maximum sum found so far (`maxSum`). ### Complexity: - Time Complexity: O(n), since we only traverse the array once. - Space Complexity: O(1), no extra array is needed. Below is the C++ code implementation:

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = INT_MIN;
        int sum = 0;

        for(int i = 0; i < nums.size(); i++){
            sum += nums[i];
            maxSum = max(sum, maxSum);
            if(sum < 0){
                sum = 0;
            }
        }

        return maxSum;
    }
};
```