1. Reverse String

Time Complexity: O(n)

Space Complexity: O(1)

class Solution {

public void getreverseString(char[] s, int left, int right){

while(left < right){

char temp = s[left];

s[left] = s[right];

s[right] = temp;

left++;

right--;

}

}

public void reverseString(char[] s) {

getreverseString(s, 0, s.length - 1);

}

}

2. Length of Last Word

Time Complexity: O(N)

Space Complexity: O(1)

class Solution {

public int lengthOfLastWord(String s) {

s = s.trim();

return s.length() - s.lastIndexOf(" ") - 1;

}

}

3. Reverse Words in a String

Time Complexity: O(N)

Space Complexity: O(N)

class Solution {

public String reverseWords(String s) {

s = s.trim();

List<String> wordList = Arrays.asList(s.split("\\s+"));

Collections.reverse(wordList);

return String.join(" ", wordList);

}

}

4. Maximum Subarray Sum

// Time Complexity: O(n)

// Space Complexity: O(1)

class Solution {

public int maxSubArray(int[] nums) {

// Kadane's Algorithm

int sum = 0;

int maxi = nums[0];

for(int i=0; i<nums.length; i++){

sum += nums[i];

if(sum > maxi){

maxi = sum;

}

if(sum < 0){

sum = 0;

}

}

return maxi;

}

}