1. Find Largest sum contiguous subarray?

i/p = Ex. -2, -3, 4, -1, -2, -1, 5, -3

Here in this array

o/p = 4, -1, -2, -1, 5 = 5

-2, -3, 4, -1, -2, -1, 5, -3

I

def solve(*arr*):

n = len(arr)

curSum = arr[0]

result = arr[0]

for i in range(1,n):

curSum = max(curSum+arr[i], arr[i])

result = max(result, curSum)

return result

print(solve([ -2, -3, 4, -1, -2, -1, 5, -3]))

*# Time complexity: O(n)*

*# Space complexity: O(1)*

Given a input string, write a program to produce the output in the below mentioned way and write the time and space complexity.

Example 1 : input ”AAABBCCADDDD” output -> A3B2C2A1D4

def solve2(*string*):

n = len(string)

count = 1

result = ""

for i in range(1,n):

if string[i]==string[i-1]:

count+=1

else:

result += string[i-1] + str(count)

count = 1

if count>0:

result += string[n-1]+str(count)

return result

print(solve2("AAABBCCADDDD")=="A3B2C2A1D4")

*# Time complexity: O(n)*

*# Space complexity: O(1)*

Given array, second largest

[ i j ]