Problem # 209 : Minimum Size Subarray Sum

<https://leetcode.com/problems/minimum-size-subarray-sum/>

Solution:

1. Initialize result to sys.maxsize, left pointer to 0 and running total to 0.

2. Iterate through the nums array and add the elements to the running total.

3. If the running total >= s, we remove the left most element from the total,

and the left pointer is incremented (i.e. it is moved to the next position).

The result is updated if the length of the subarray is less than the existing result.

The length of the subarray = index + 1 – left (we have index + 1 since we have

0-based index).

4. If the result is not sys.maxsize, return result. However, if result is still at initialized

value of maxsize, then result 0 since there is no subarray which has total >= s.

--------------------------------------

import sys

class Solution:

def minSubArrayLen(self, s: int, nums: List[int]) -> int:

n = len(nums)

result = sys.maxsize

left = 0

total = 0

for i in range(n):

total += nums[i]

while total >= s:

total -= nums[left]

result = min(result, i + 1 - left)

left += 1

if result != sys.maxsize:

return(result)

return(0)

-------------------------------------------------------