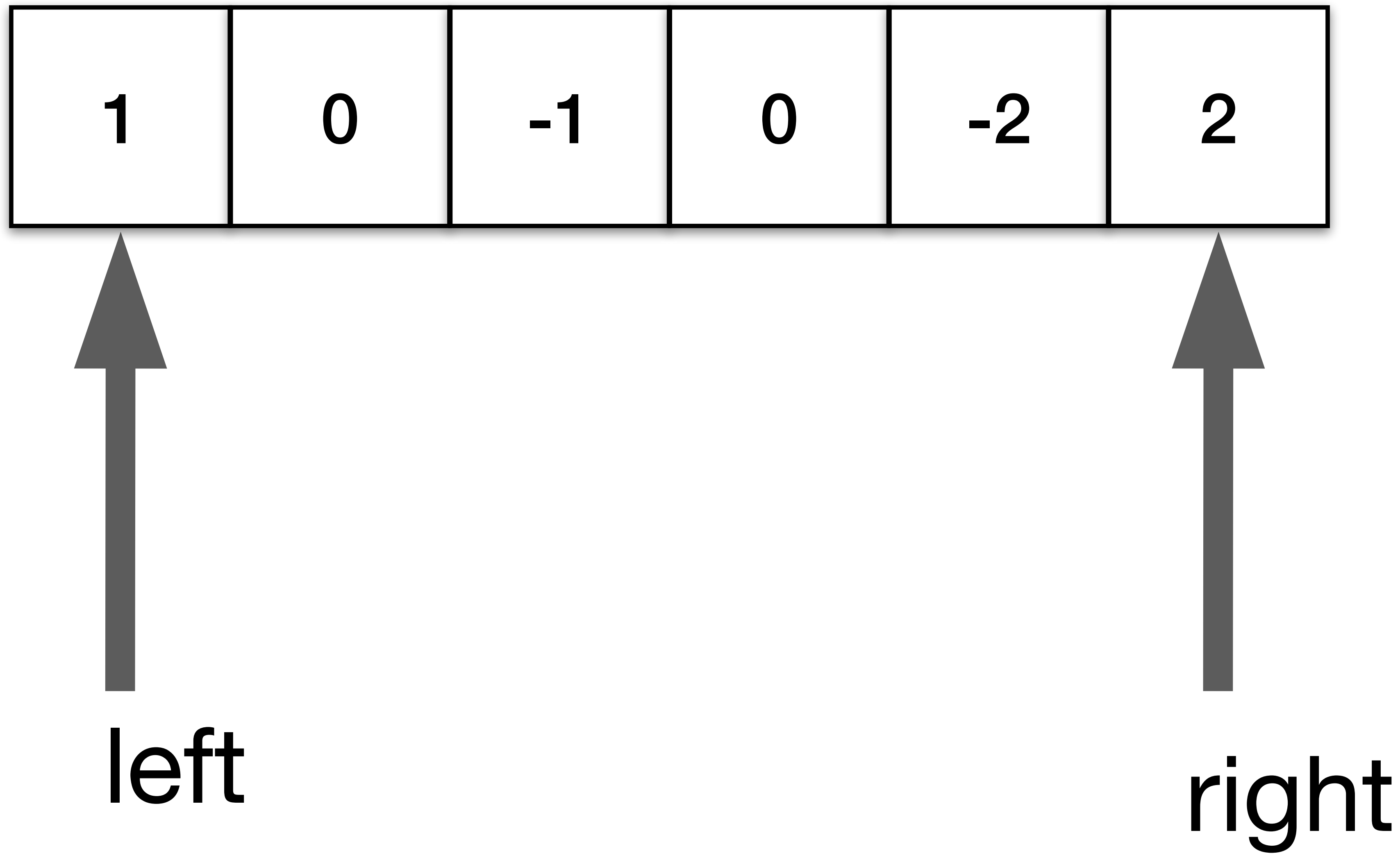
4 Sum - Test Case -1

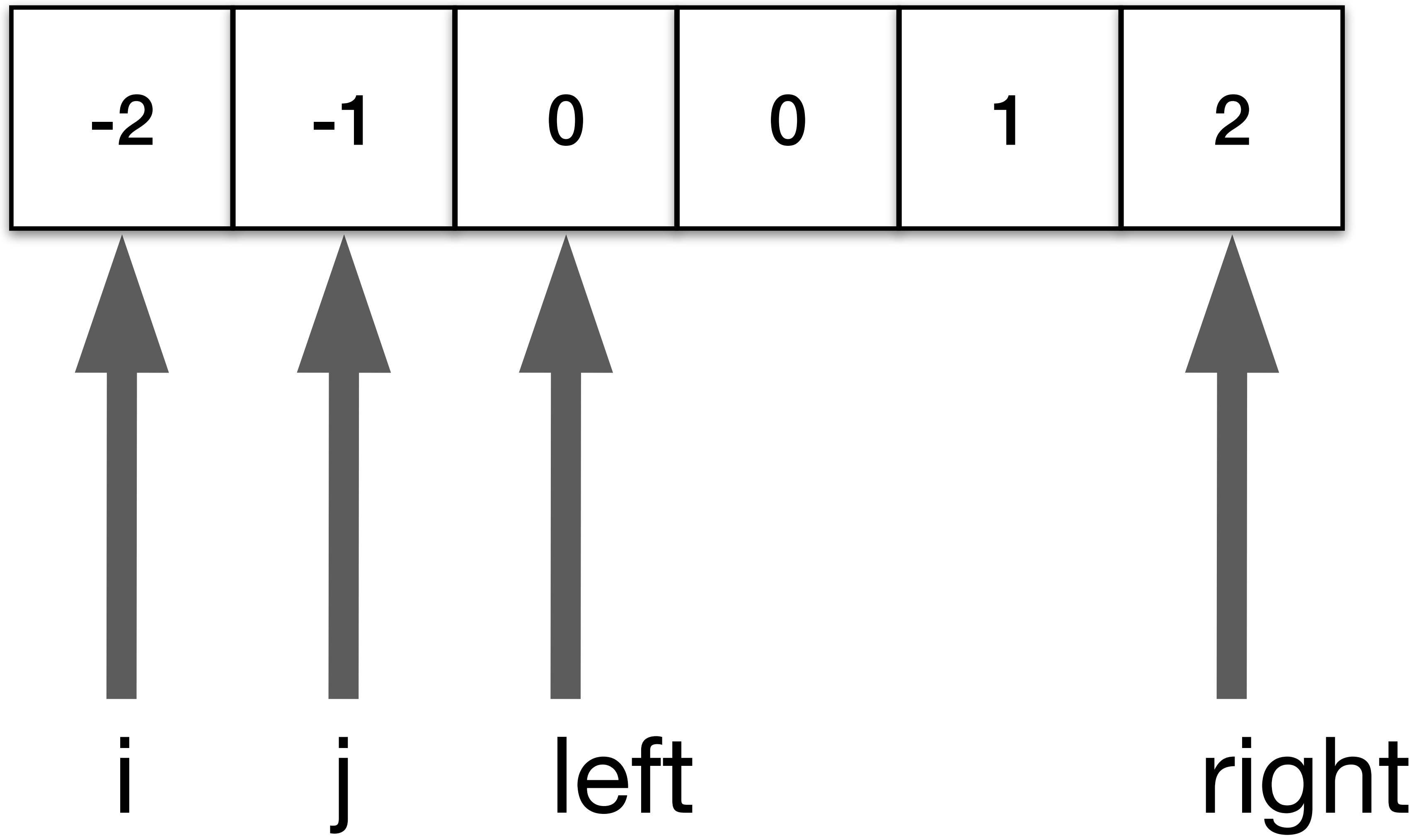


1	0	-1	0	-2	2

left

right

4 Sum - Test Case -1- Template Implementation

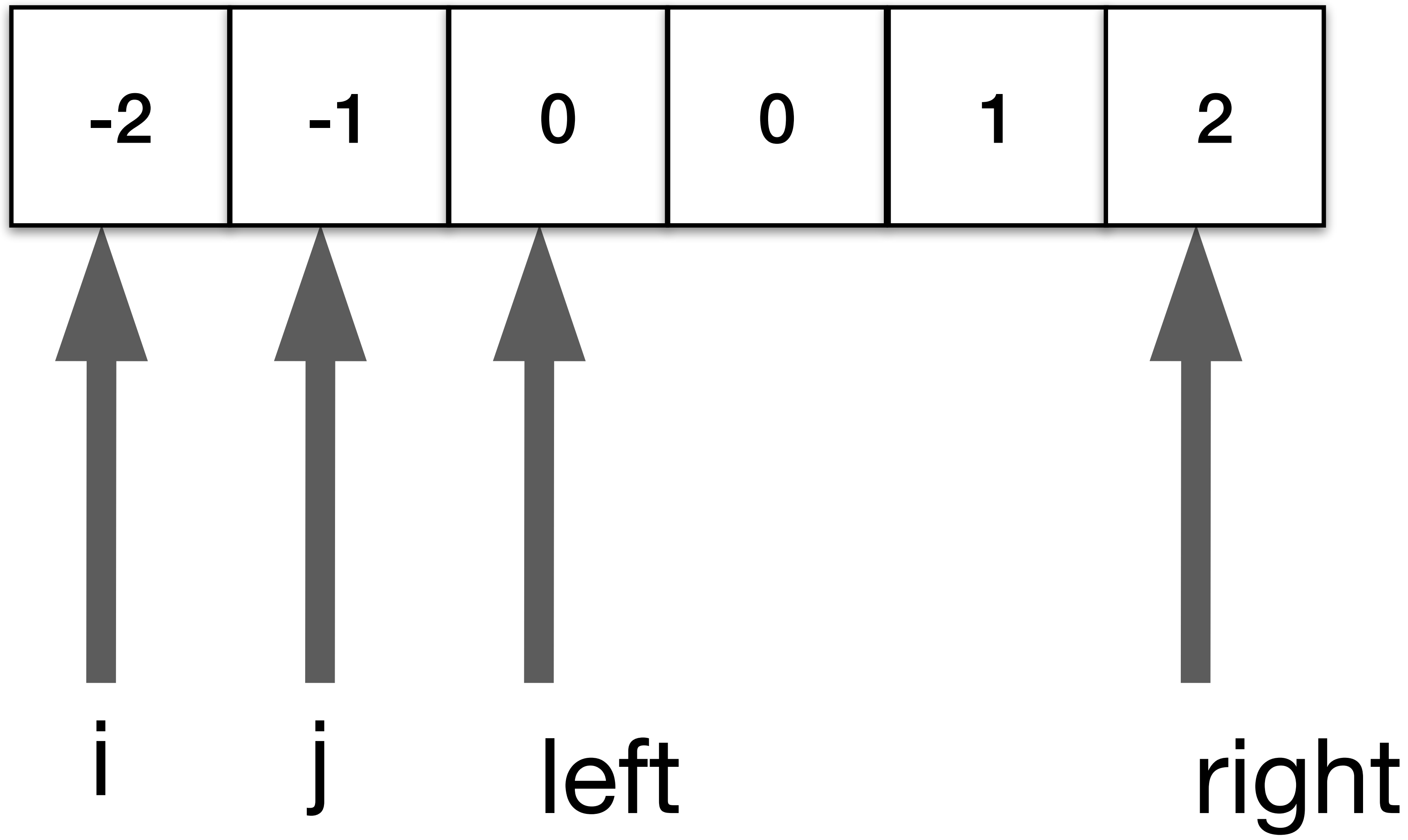


-2	-1	0	0	1	2

i j left right

Template :- Along with left and right pointers, We need to use two more pointers I and j to solve the problem and need to sort the array to get the maximum sum.

4 Sum - Test Case -1



-2	-1	0	0	1	2

CurrentSum = -2 + (-1) + 0 + 2 = -1

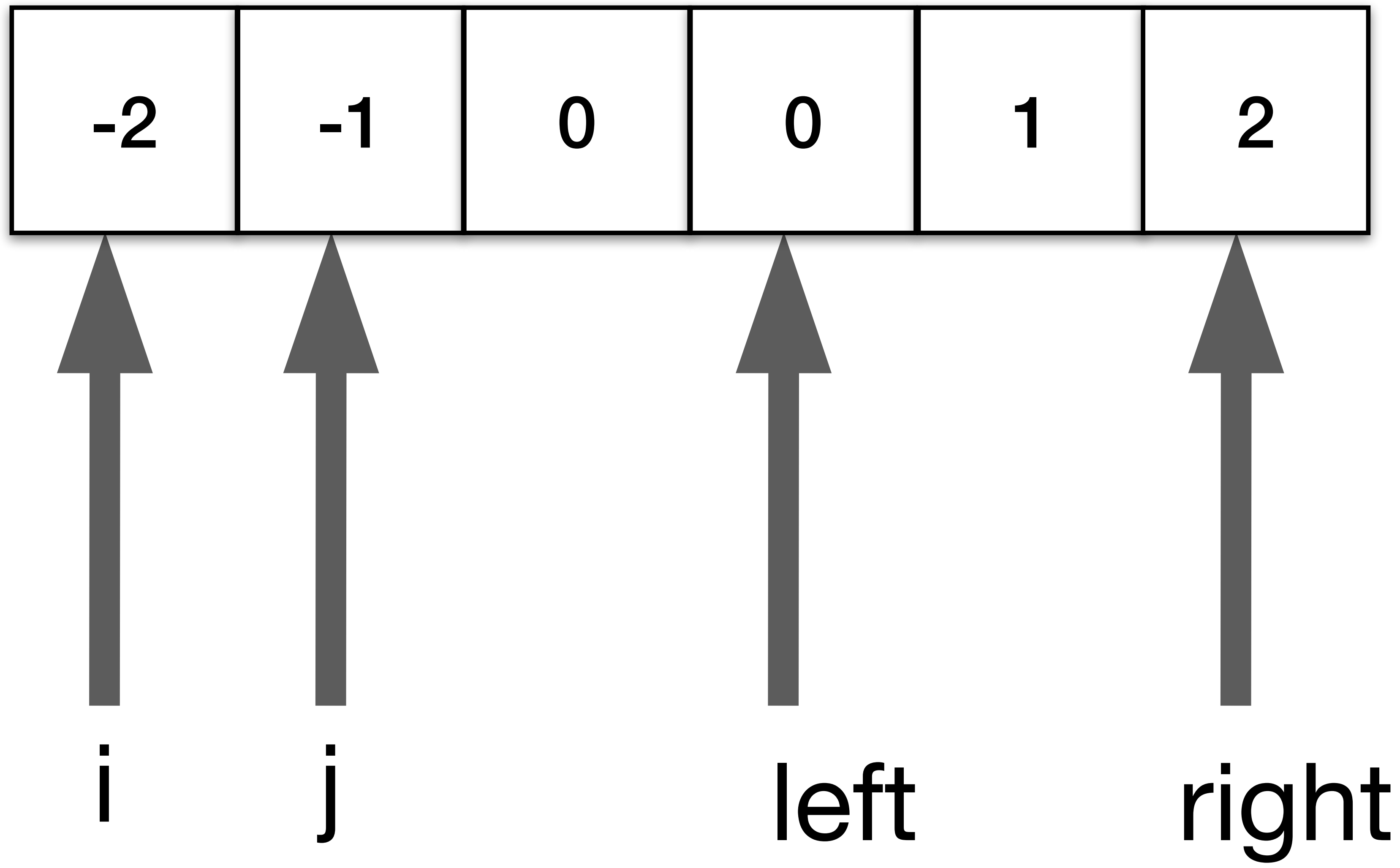
Target = 0

-1 < 0 OR currentSum < target

Output = []

Less than check scenario in opposite direction pointers template

4 Sum - Test Case -1



-2	-1	0	0	1	2

CurrentSum = -2 + (-1) + 0 + 2 = -1

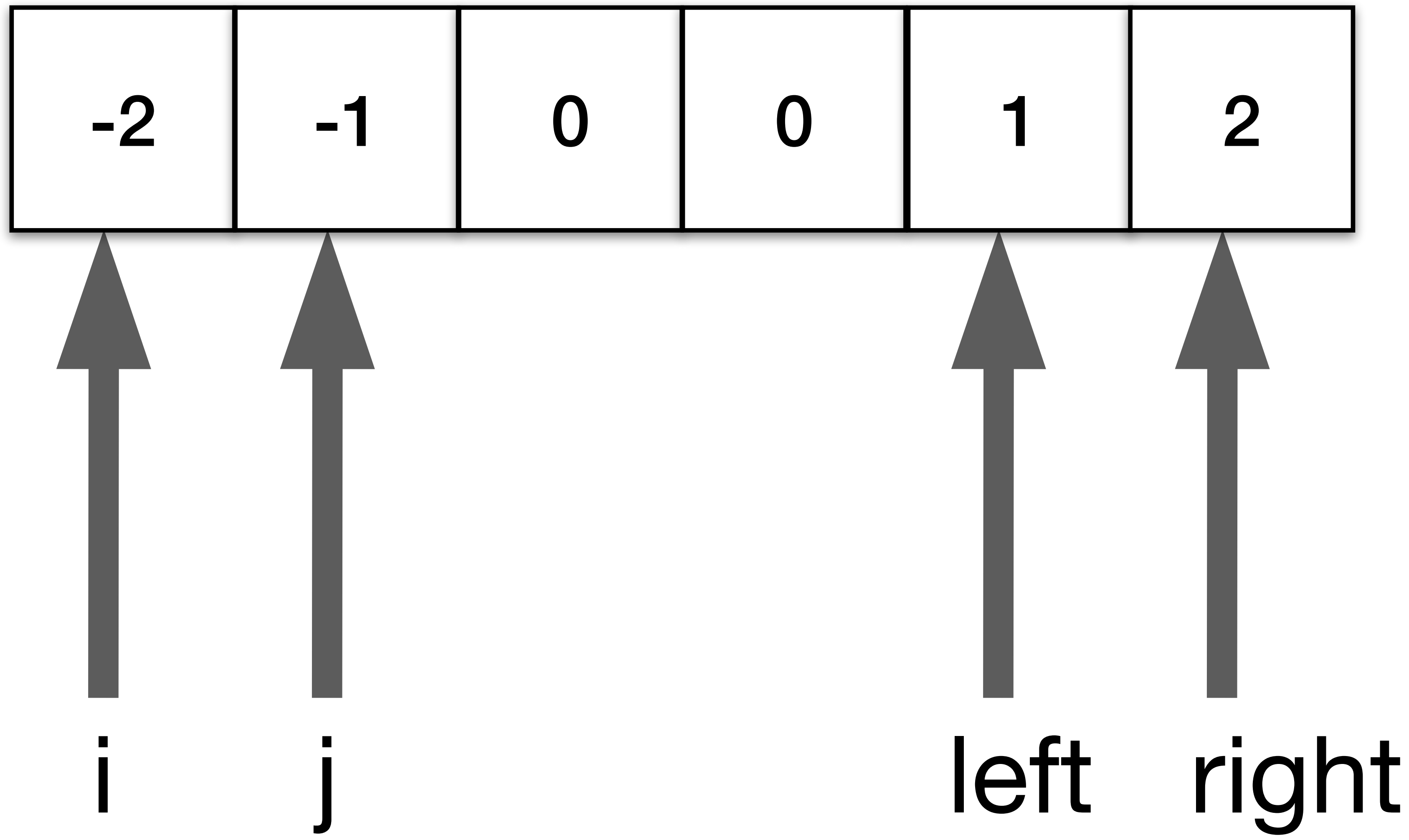
Target = 0

-1 < 0 OR currentSum < target

Output = []

Less than check scenario in opposite direction pointers template

4 Sum - Test Case -1



-2	-1	0	0	1	2
i	j			left	right

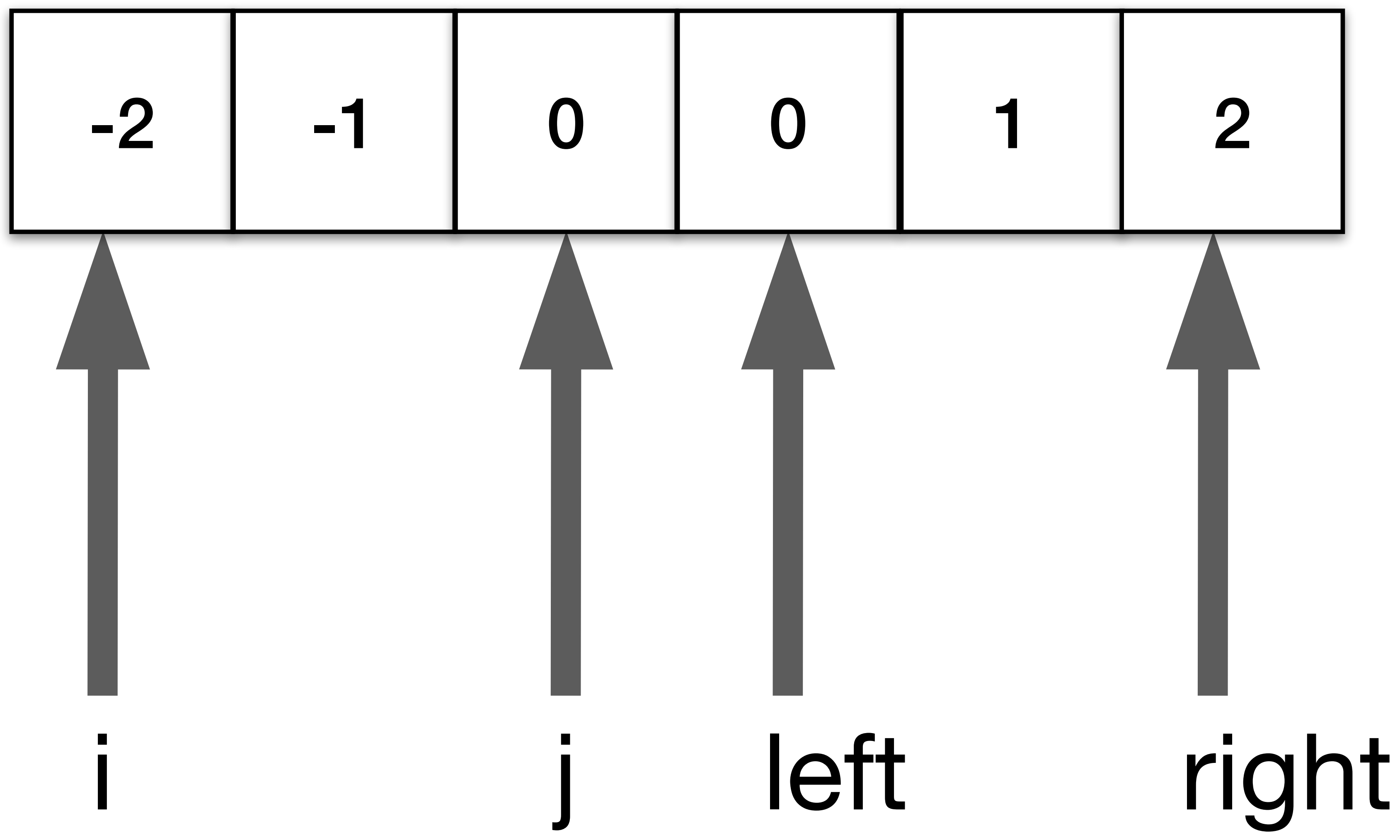
CurrentSum = -2 + (-1) + 1 + 2 = 0

Target = 0

0 = 0 OR currentSum == target

Output = [-2, -1, 1, 2]

4 Sum - Test Case -1



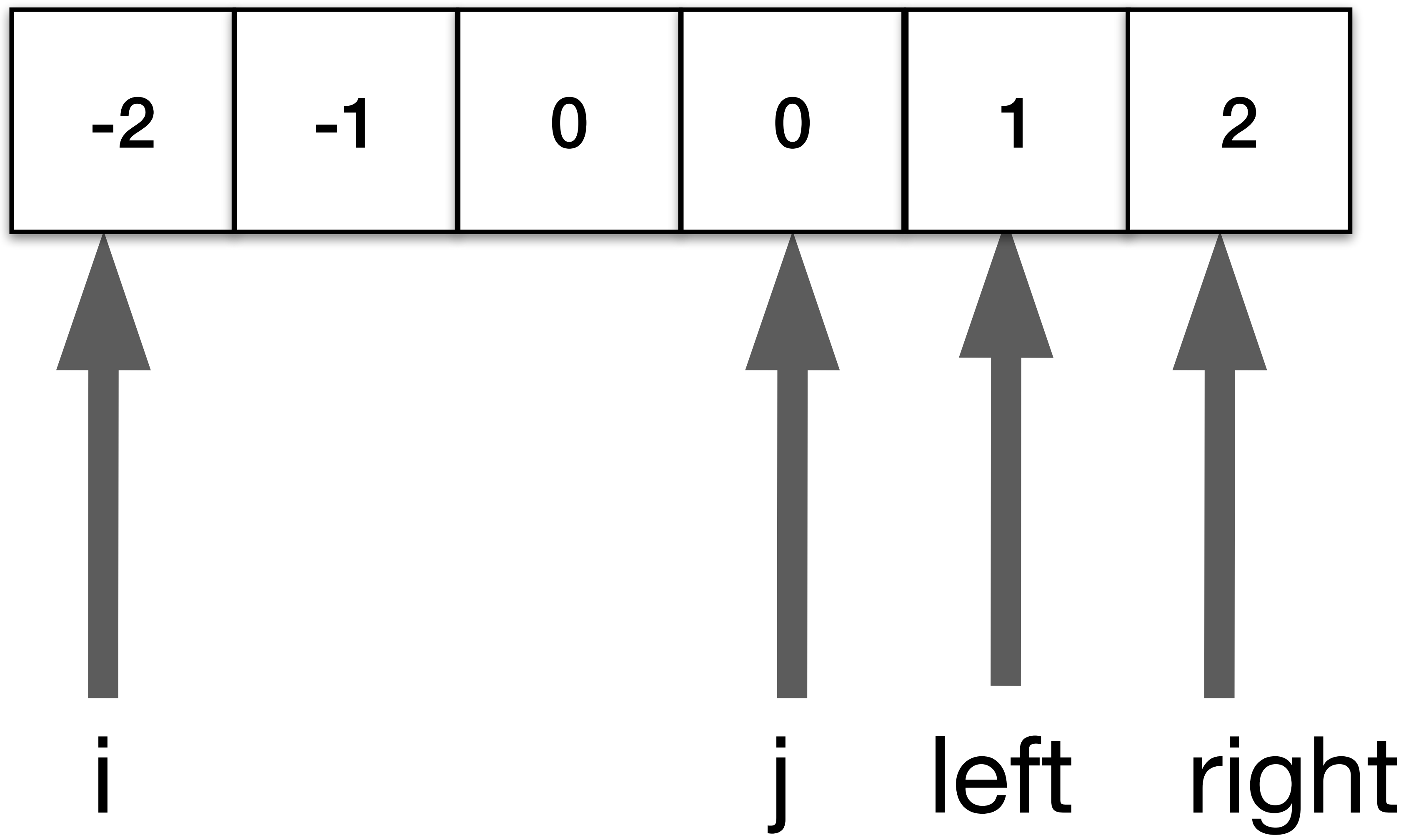
CurrentSum = -2 + 0 + 0 + 2 = 0

Target = 0

0 = 0 OR currentSum == target

Output = [-2, -1, 1, 2], [-2, 0, 0, 2]

4 Sum - Test Case -1



-2	-1	0	0	1	2

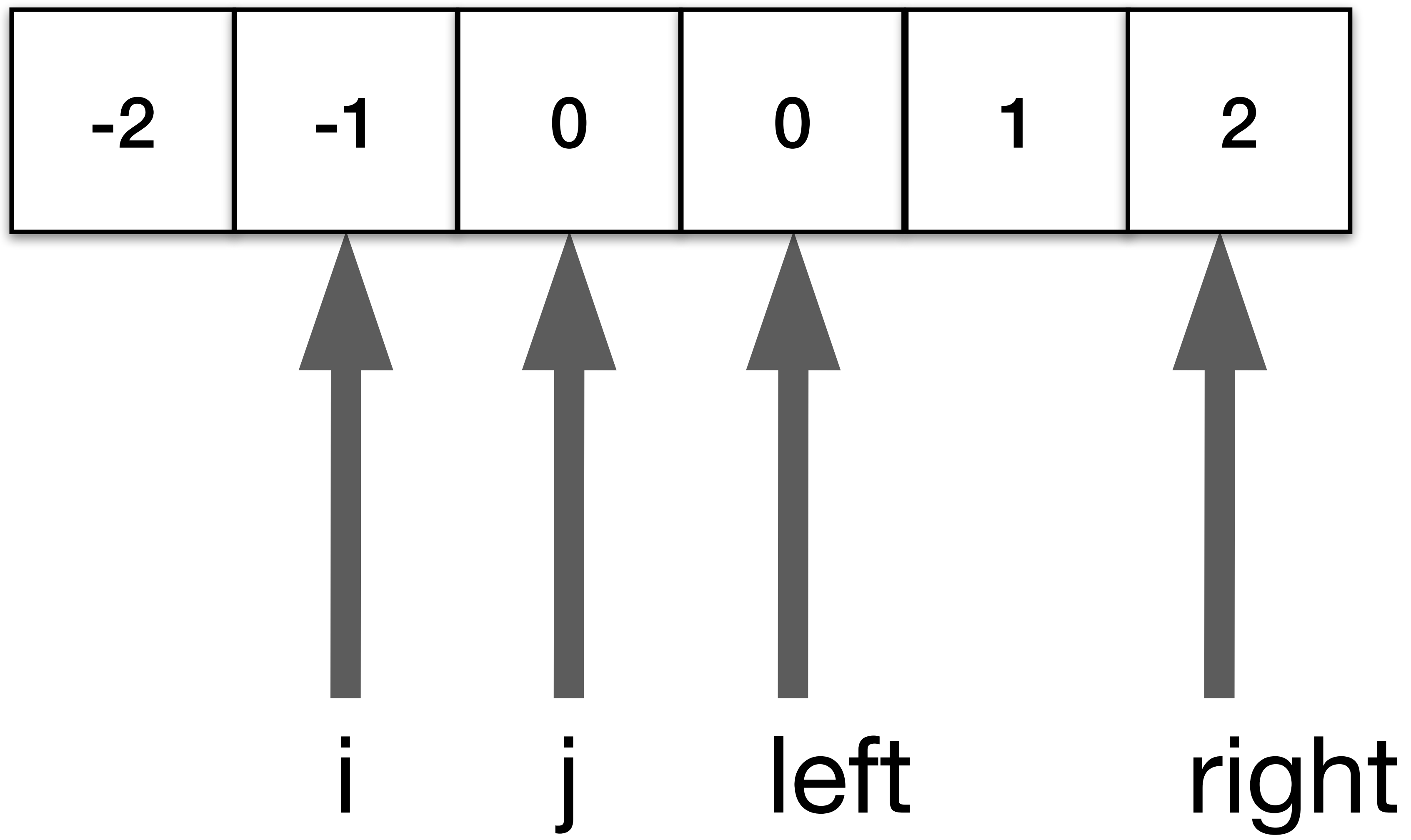
CurrentSum = -2 + 0 + 1 + 2 = 0

Target = 0

1 > 0 OR currentSum > target

Output = [-2, -1, 1, 2], [-2, 0, 0, 2]

4 Sum - Test Case -1



-2	-1	0	0	1	2
	i	j	left		right

CurrentSum = -1 + 0 + 0 + 2 = 1

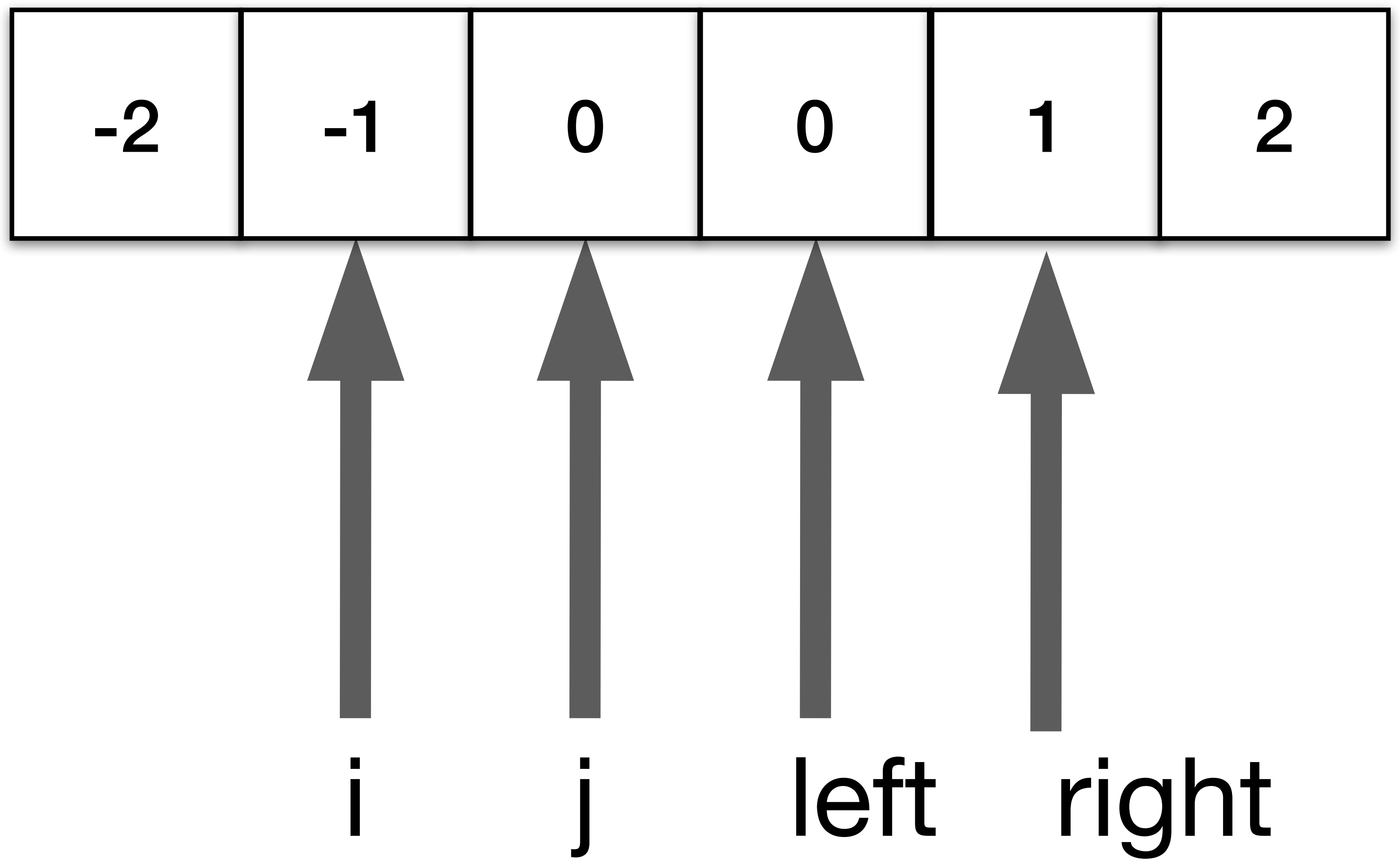
Target = 0

1 > 0 OR currentSum > target

Output = [-2, -1, 1, 2], [-2, 0, 0, 2]

Greater than check scenario in opposite direction pointers template

4 Sum - Test Case -1



-2	-1	0	0	1	2

i j left right

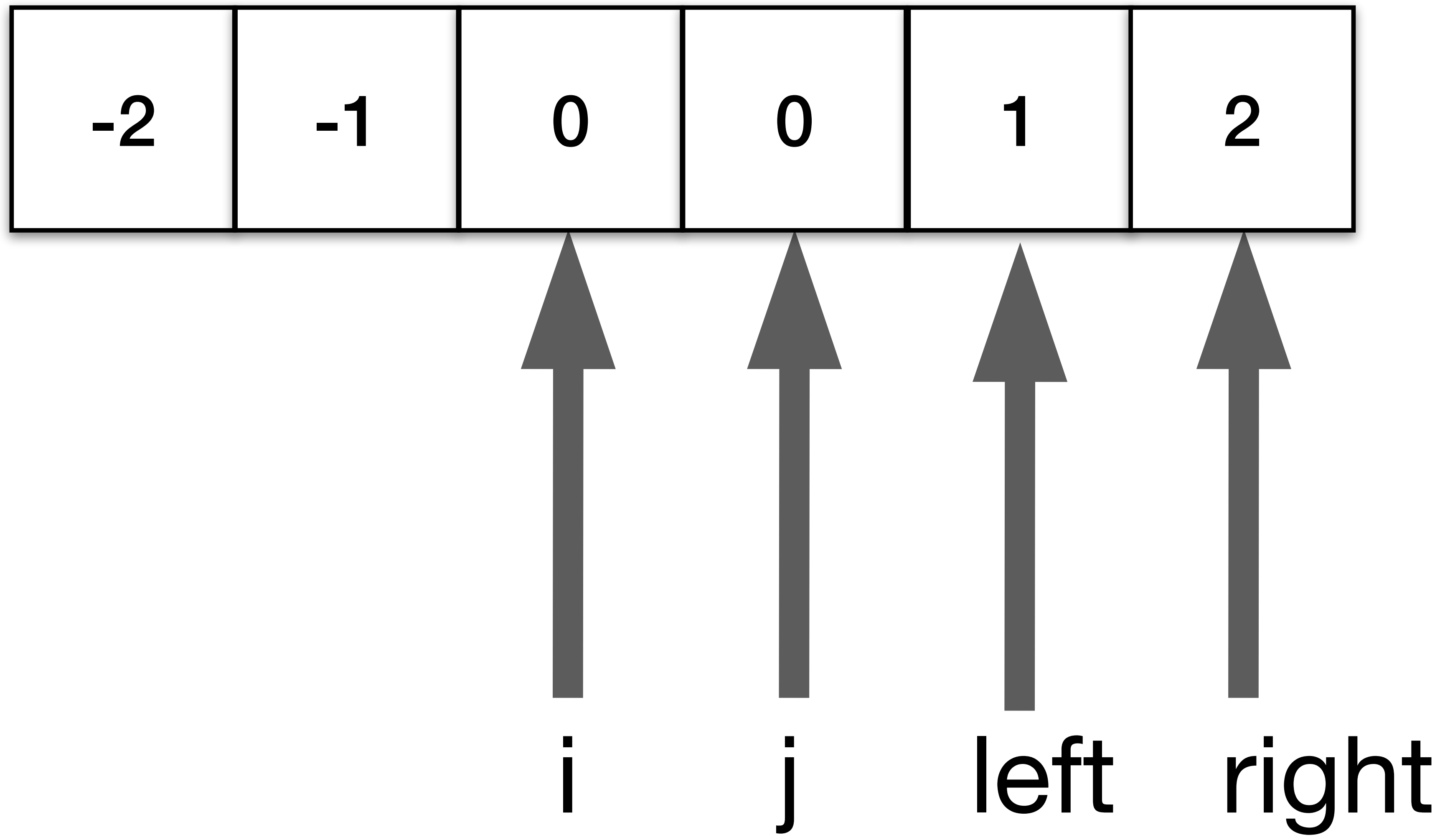
CurrentSum = -1 + 0 + 0 + 1 = 0

Target = 0

0 = 0 OR currentSum == target

Output = [-2, -1, 1, 2], [-2, 0, 0, 2],
[-1, 0, 0, 1]

4 Sum - Test Case -1



-2	-1	0	0	1	2

CurrentSum = 0 + 0 + 1 + 2 = 3

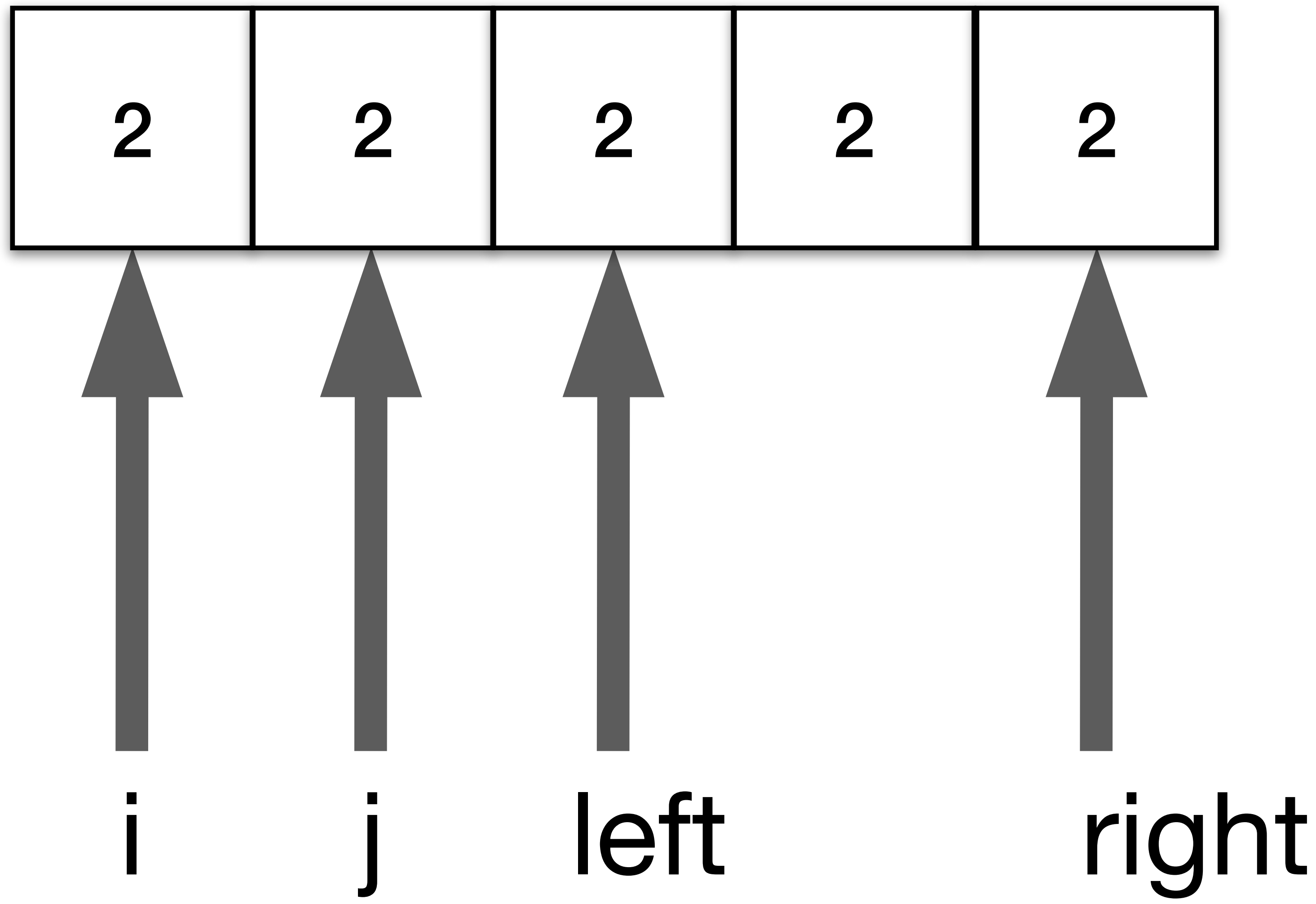
Target = 0

3 > 0 OR currentSum > target

Output = [-2, -1, 1, 2], [-2, 0, 0, 2], [-1, 0, 0, 1]

Greater than check scenario in opposite direction pointers template

4 Sum - Test Case -2



2	2	2	2	2

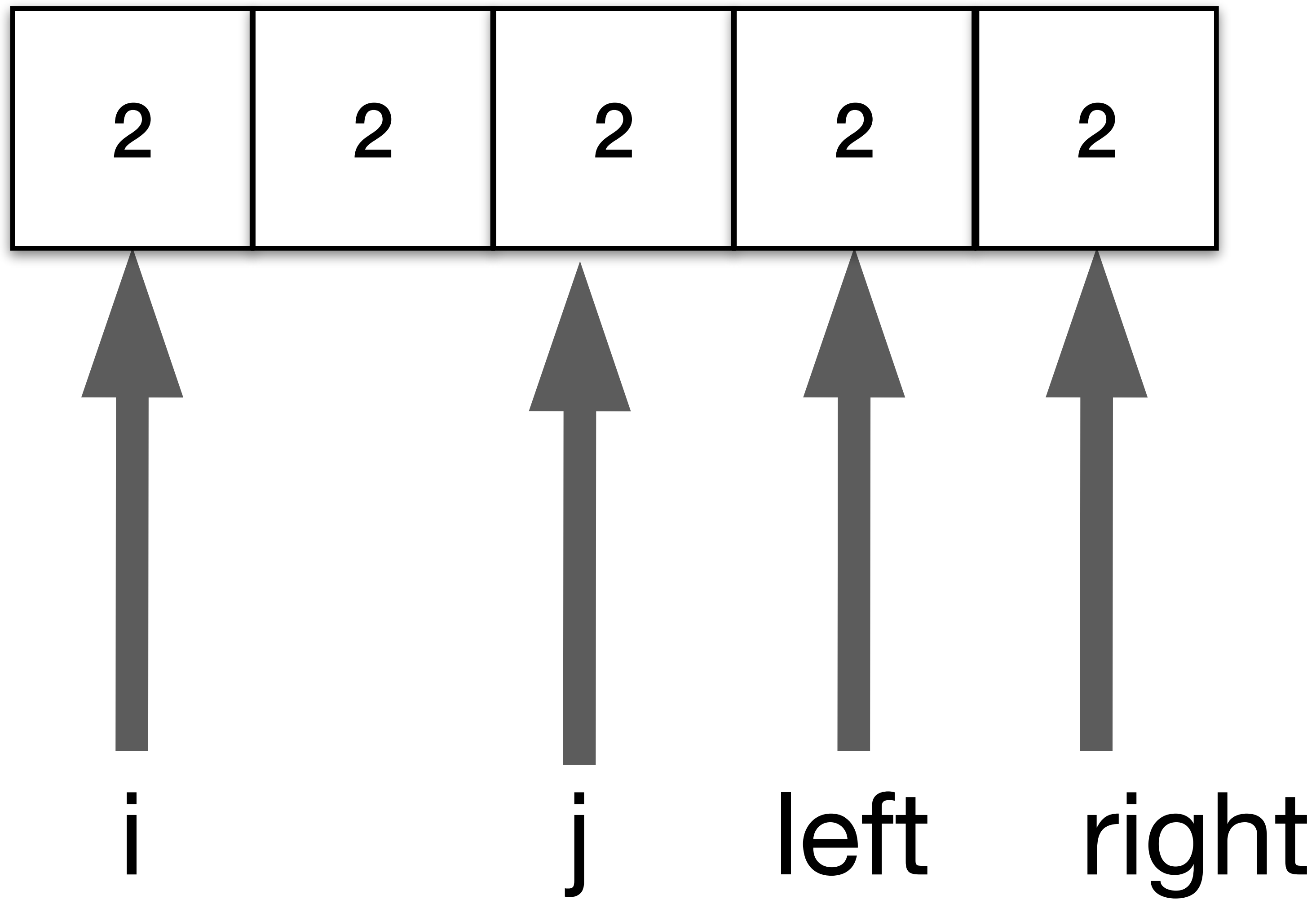
CurrentSum = 2 + 2 + 2 + 2 = 8

Target = 8

8 = 8 OR currentSum == target

Output = [2, 2, 2, 2]

4 Sum - Test Case -2 (Scenario 1 - Duplicate Scenario for j index)



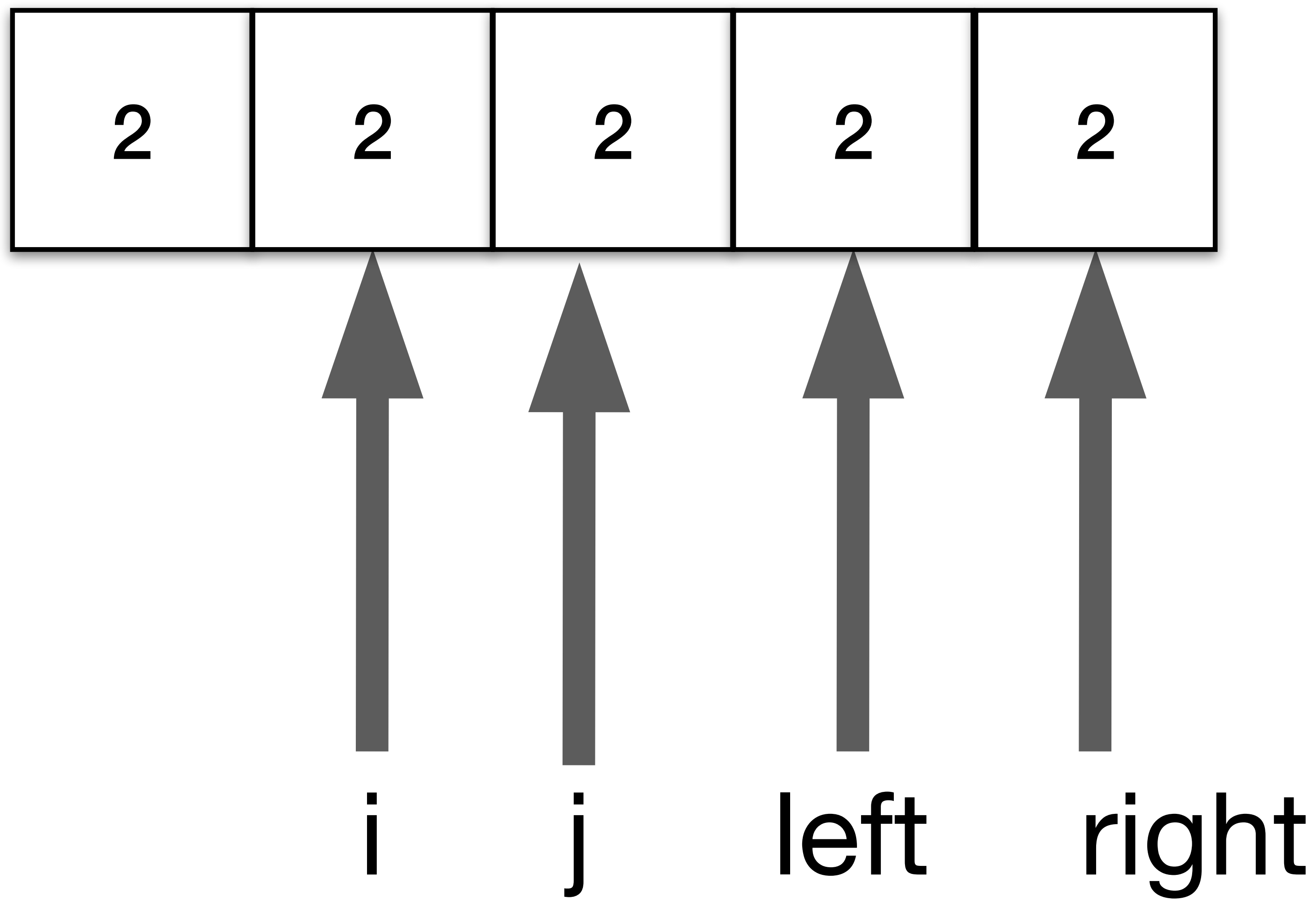
CurrentSum = 2 + 2 + 2 + 2 = 8

Target = 8

8 = 8 OR currentSum == target

Output = [2, 2, 2, 2], [2, 2, 2, 2]

4 Sum - Test Case -2 (Scenario 2 - Duplicate Scenario for i index)



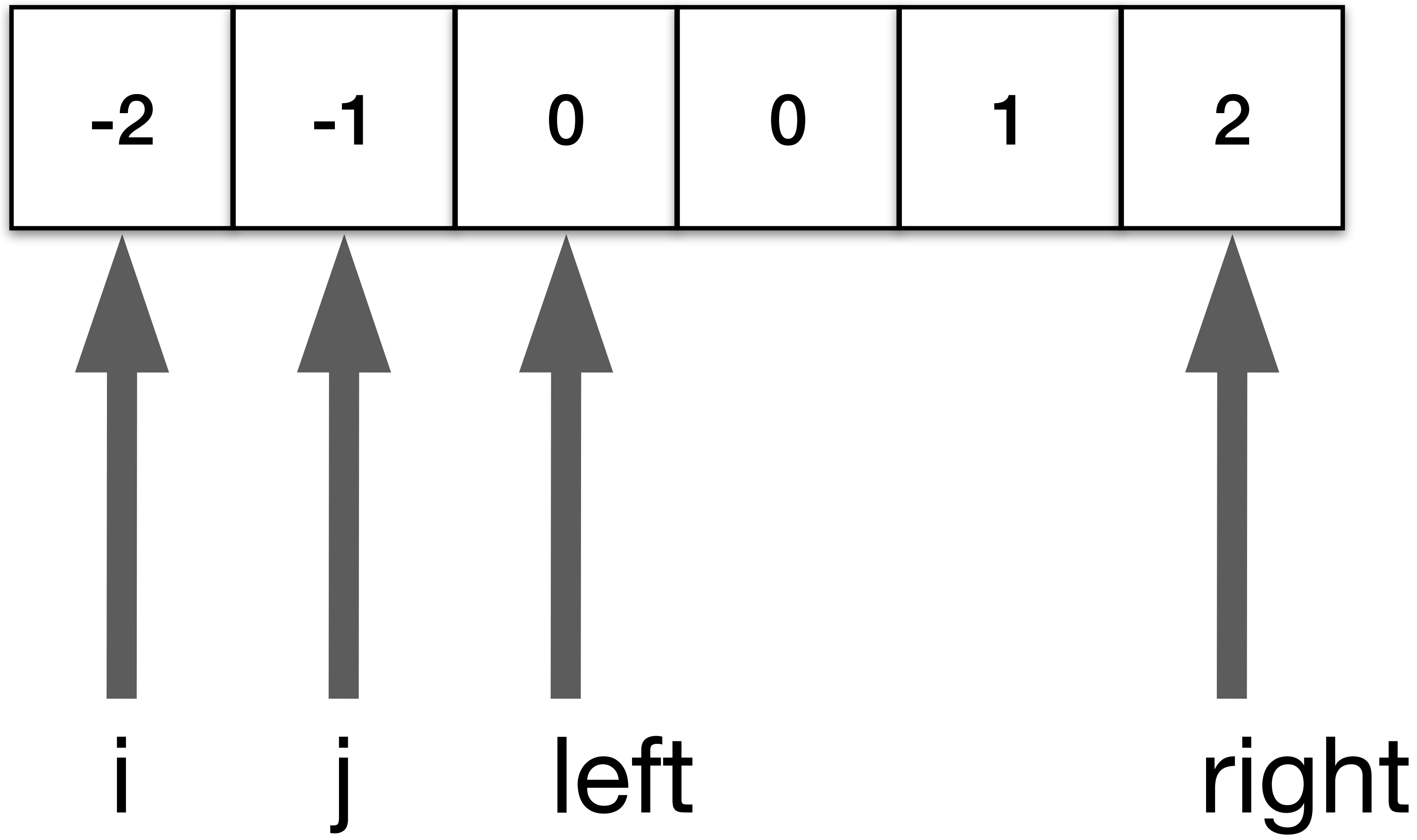
CurrentSum = 2 + 2 + 2 + 2 = 8

Target = 8

8 = 8 OR currentSum == target

Output = [2, 2, 2, 2], [2, 2, 2, 2],
[2, 2, 2, 2]

Step 1 :- Dynamic Params for Template



-2	-1	0	0	1	2

i = 0

j = 1 = i+1

left = 3 = j+1

right = 5 = nums.length-1

sort(nums)

Pointers Details :-

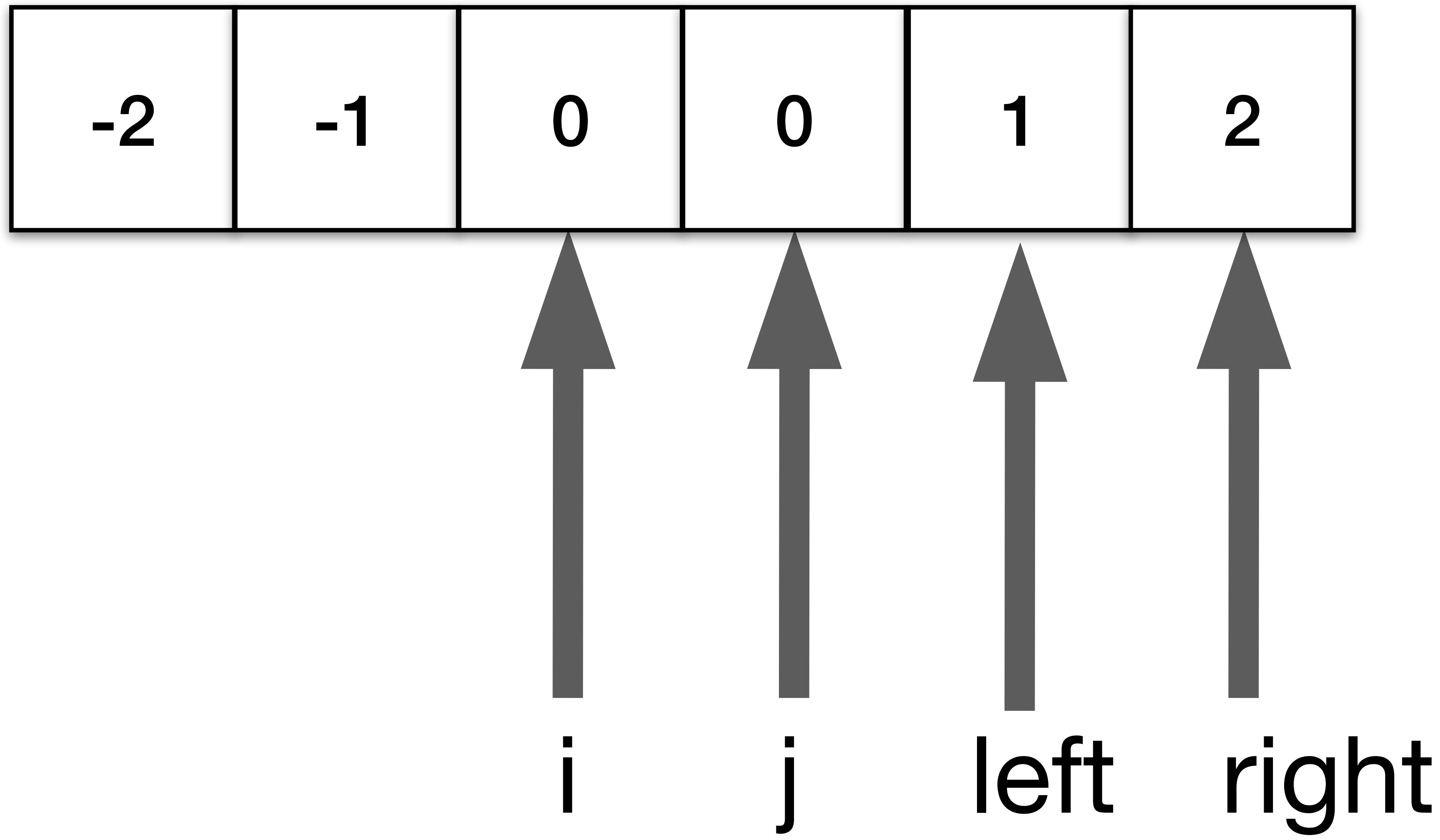
Loop for i :- start from 0 to nums.length-3

Loop for j :- start from i+1 to nums.length-2

left = j+1

right = nums.length -1

Step 2 :- Dynamic Params for I/O



CurrentSum = 0 + 0 + 1 + 2 = 3

Target = 0

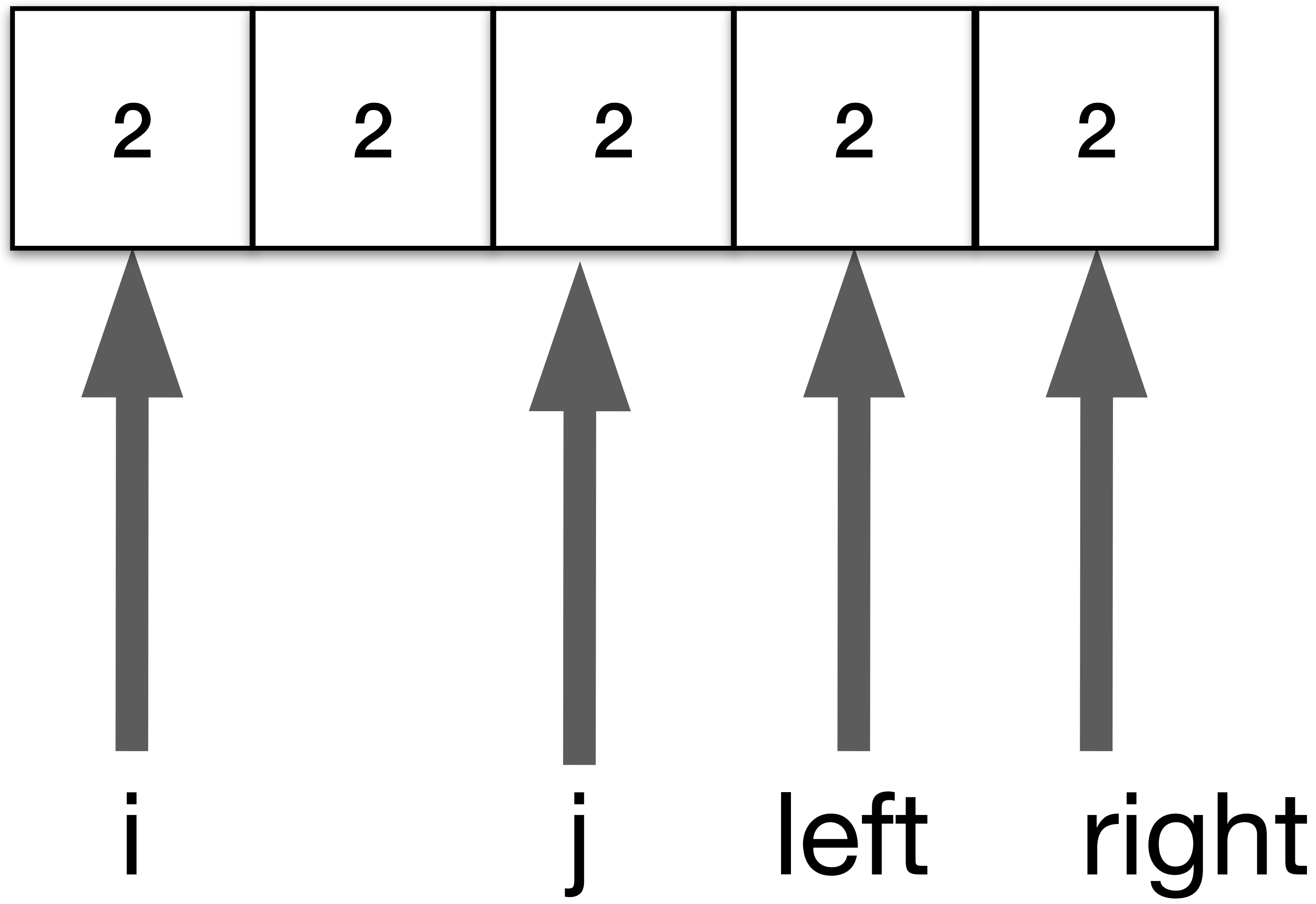
Output = [-2, -1, 1, 2], [-2, 0, 0, 2],
[-1, 0, 0, 1]

Param1 = nums[i] + nums[j] + nums[left] + nums[right]

Param2 = target

output = list of(nums[i] + nums[j] + nums[left] + nums[right])

Step 3 :- Dynamic Params for duplicate Scenario 1 and Scenario 2



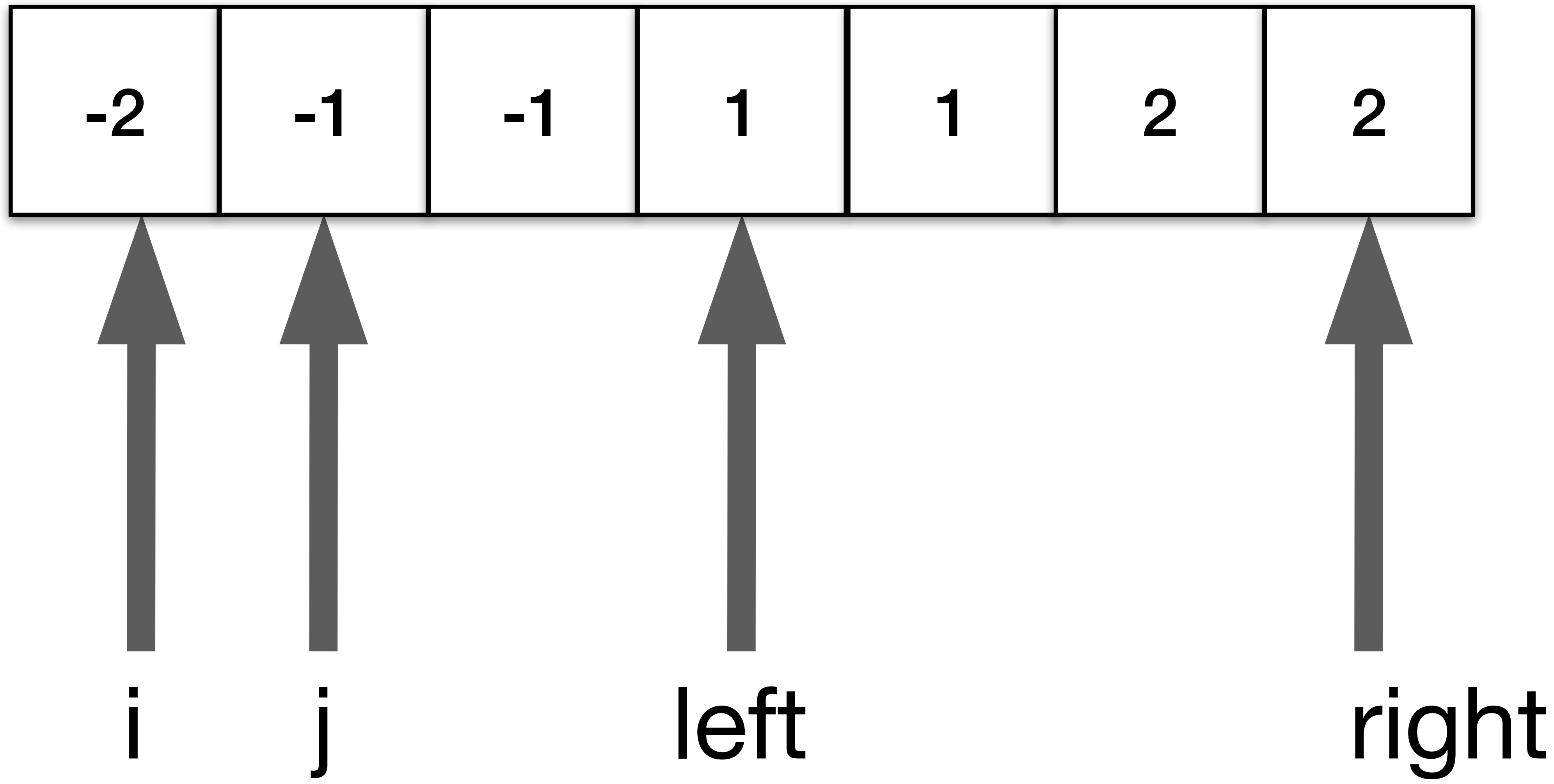
if 2 == 2 so we will skip calculations
for this j index

if nums[j-1] == nums[j] then continue the loop

if 2 == 2 so we will skip calculations
for this i index

if nums[i-1] == nums[i] then continue the loop

4 Sum - Corner Test Case



-2	-1	-1	1	1	2	2

CurrentSum = -2 + -1 + 1 + 2 = 0

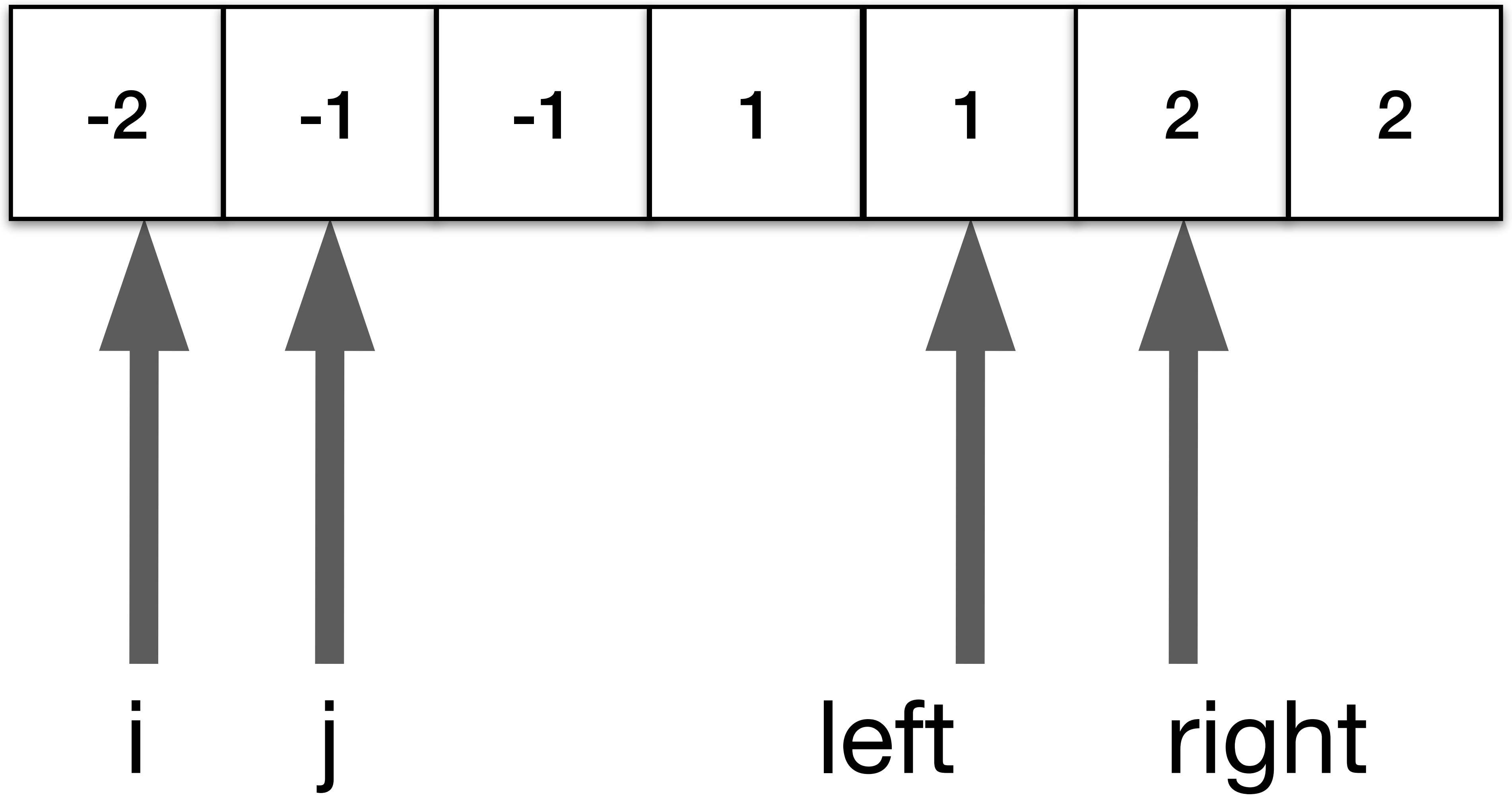
Target = 0

0 = 0 OR currentSum == target

Output = [-2, -1, 1, 2]

Equality check scenario in opposite direction pointers template

4 Sum - Corner Test Case



-2	-1	-1	1	1	2	2

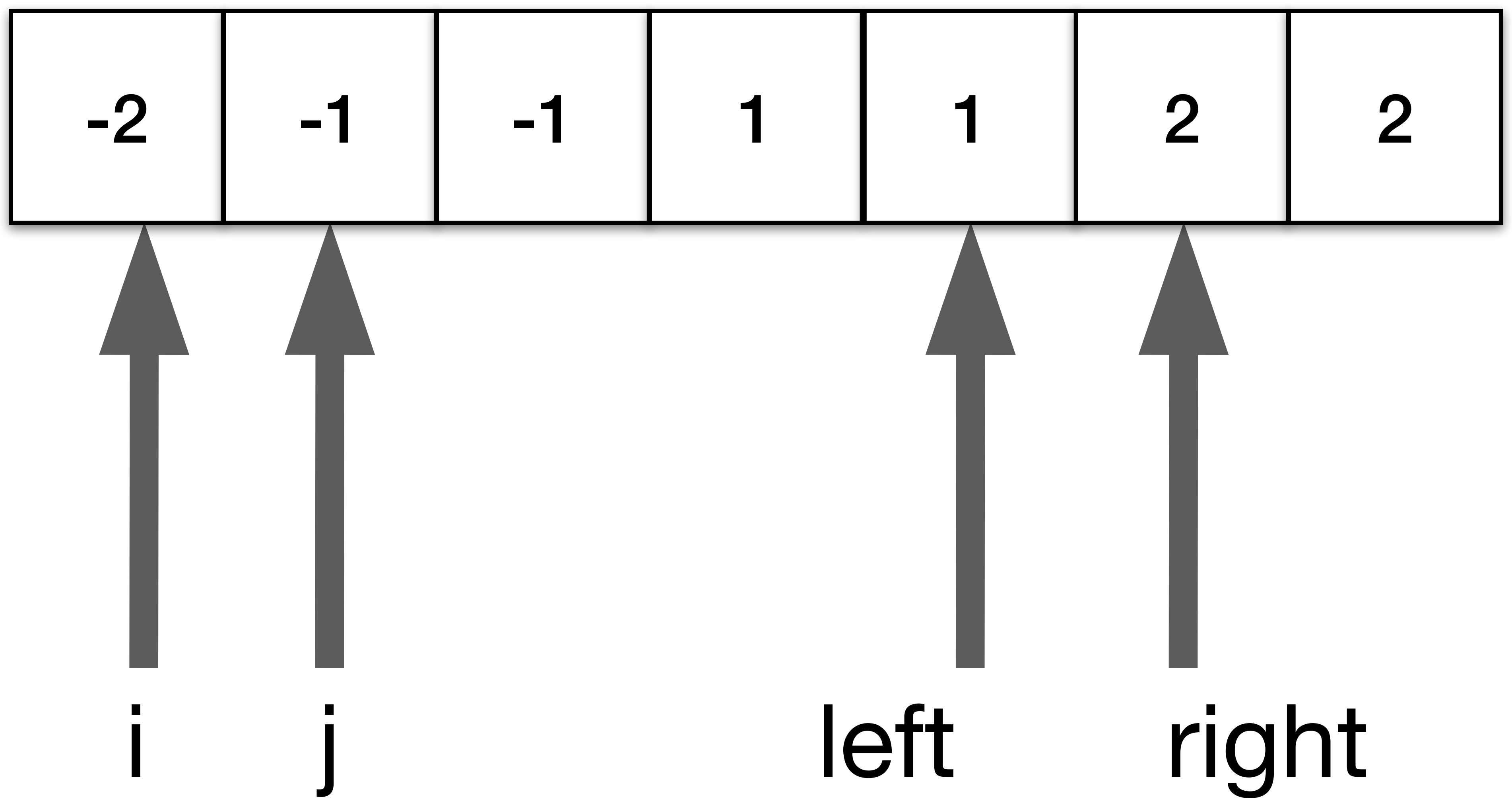
CurrentSum = -2 + -1 + 1 + 2 = 0

Target = 0

0 = 0 OR currentSum == target

Output = [-2, -1, 1, 2], [-2, -1, 1, 2]

Step 4 :- Dynamic params for corner test cases



-2	-1	-1	1	1	2	2

CurrentSum = -2 + -1 + 1 + 2 = 0

Target = 0

0 = 0 OR currentSum == target

Output = [-2, -1, 1, 2], [-2, -1, 1, 2]

while nums[left] == nums[left+1]
so left++

while nums[right] == nums[right-1]
so right—