Q⇒ Given an array, print all subarrays of the given array. (You can print in any order).

Subarray → It is a contiguous cross-section of your array.

Ex → [1, 2, 3, 4]

1
2
3
4
1, 2
2, 3
3, 4
1, 2, 3
2, 3, 4
1, 2, 3, 4

1, 3 → this is not Subarray.

↓ start

[1 , 2 , 3 , 4]
0 1 2 3

↑ end

Str =

1

1 2

1 2 3

1 2 3 4

2

2 3

2 3 4

3

3 4

Every subarray can be uniquly identified by a pair of starting index & ending index.

↓

How about, if we can Calculate all possible pair of start & end in dem.

Calculate freq → obj

[15, 15, 3, 15, 3, 1, 3, 15, 3, 3, 15, 15, 15, 15, 15, 15]

→ if we take one occ of majority element and one occ of non majority element & cancel both of them with each other, after all the possible cancellations we will be still left with the majority elements

[15, 15, 3, 15, 3, 1, 3, 15, 3, 3, 15, 15, 15, 15, 15, 15]

curr = 15

curr_element - freq - 0 1 2 1 2 1 0 1 2 1 0 1

Possible

majority element

num1 → [2, 9, 1, 2, 2, 3, 9]

num2 → [2, 1, 1, 7, 6, 8, 3]

[2, 1, 3]

if somehow we can get all the elements which are unique in nums1

→ freq map

{ 2 : 3
  9 : 2
  1 : 1
  3 : 1 }

freqMap → nums2 → { 2 : 1
                    1 : 2
                    3 : 1 }

$$a + b + c = 0$$

$$a + b = -c$$

assume $-c = m$

$a + b = m$

?? ↑

↳ you need to find pairs that Sum upto m.

$a + b + c = 0$

$a + b = -c$

Sorted

↳ two pointers

$i \to c$

$[a_1 \; a_2 \; a_3 \; \ldots\ldots \; a_n]$

↳ pairs that sum upto $-a_1$