

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

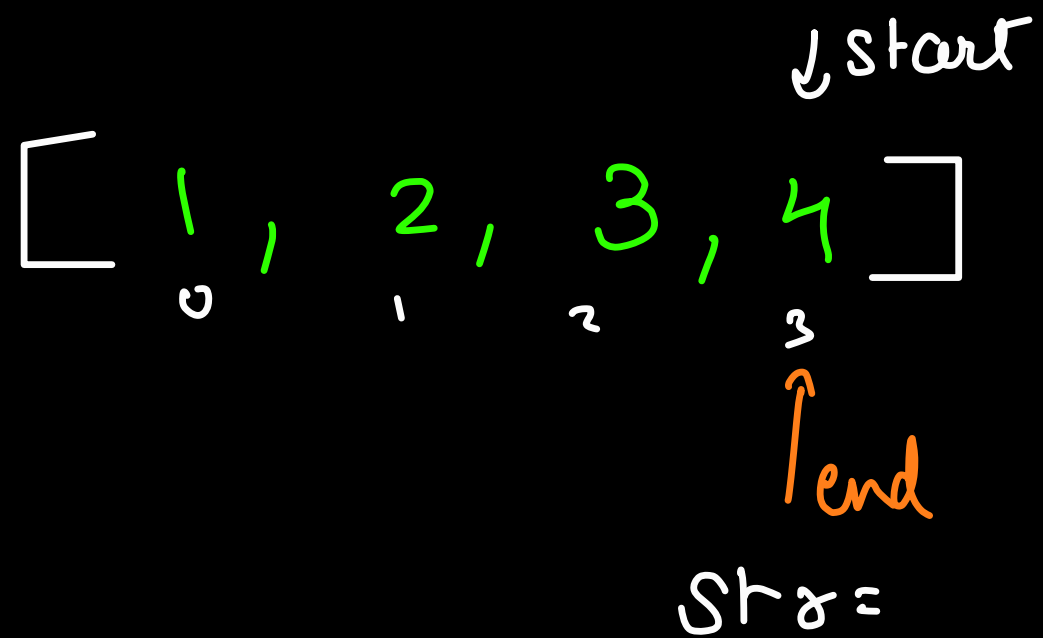
3, 4

1, 2, 3

2, 3, 4

1, 2, 3, 4

1, 3 → this is not Subarray.



- 1
- 1 2
- 1 2 3
- 1 2 3 4
- 2
- 2 3
- 2 3 4
- 3
- 3 4

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

↓

How about, if we can calculate all possible pair of start & end index.

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]



→ curr = 15

curr_element - freq = ~~0~~ 1 ~~2~~ ~~1~~ ~~2~~ ~~1~~ ~~1~~ ~~0~~ 1 ~~2~~ ~~1~~ ~~0~~ 1
2 5 4

Possible
majority
element

num1 \rightarrow [2, 9, 1, 2, 2, 3, 9]

num2 \rightarrow [2, 1, 1, 7, 6, 8, 3]

[2, 1, 3]

if somehow we can get all the elements which are unique
in num1 \rightarrow freq map

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

freq Map \rightarrow num2 \rightarrow { 2: 1
1: 2
3: 1
}

$$a + b + c = 0$$

$$a + b = -c$$

Assume $-c = \underline{\underline{m}}$

$$a + b = m$$

??
?

→ you need to find pairs that
sum up to m.

$$a + b + c = 0$$

$$a + b = -c$$

$\downarrow \rightarrow \underline{\underline{c}}$
[a_1 a_2 a_3 a_n]

\hookrightarrow pairs that sum upto $\underline{\underline{-a_1}}$

sorted

\hookrightarrow two pointers