Array : Video - 88

Leetcode - 41

~~Hard~~

Facebook
Instagram → codestorywithMIK

(Twitter) → CSwithMIK

codestorywithMIK →

Company :- Meta

41. First Missing Positive

1, 2, 3, ...

Hard | Topics | Companies | Hint

Given an unsorted integer array `nums`. Return the *smallest positive integer* that is *not present* in `nums`.

You must implement an algorithm that runs in `O(n)` time and uses `O(1)` auxiliary space.

Example :- nums = {1, 2, ~~0~~}

1, 2, 3, 4, ...

Output = 3

nums = {3, 4, ~~-1~~, 1}

<= 0

-1, 0, -3,

Output = 2

nums = {7, 8, 9, 11, 12}, n = 5

Output = 1

{1, 2, 3, 4, 5}, n = 5

Approach - 1

O(n) Space

nums =

0	1	2	3
3	4	-1	1

, n = 4

bool =

0	1	2	3	4
F	T	F	T	T

→ O(n) space

T.C. = O(n)

num = 2.

1, 2, 3, 4, 5, ...

Approach-2

"Use numbers as index".

n = size of array

[1, n]

1, 2, 3, 4

n+1

<=0

>n

	0	1	2	3
nums =	-3	-4	-2	-1

n = 4

[1, n]

return (n+1);

num = abs(nums[i]); // 1

idx = num-1 ; // 0

	0	1	2	3
nums =	3	4	2	~~0~~ 1

Cont = False

<=0

>n

num =

idx =

(1, n)

	0	1	2	3	
nums =	-3	4	-1	-1	, n = 4
	✗	✗	✗	✗	i

contains1 = True

=

2

num = 1

idx = 0

	0	1	2	3	4	5	6	7	8	9	
nums =	3	4	~~-1~~	~~-2~~	1	5	~~16~~	~~0~~	2	~~0~~	, n = 10

contains1 = True

	0	1	2	3	4	5	6	7	8	9	
	-3	-4	-1	-1	-1	5	1	1	2	1	, n = 10
	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	i

5

11

1, 2, 3, 4, 5, 6 6 → 6-1 5

$T.C = O(n)$ (2 pass).

$S.C = O(1)$.

6

$[1, n]$, n

Pattern ⇒

1

41