- Given an unsorted array, return the smallest missing positive integer.

Restriction: Algorithm MUST be $O(n)$ time and $O(1)$ space.

One solution:

1.) Sort array

2.) Find smallest positive value not in array.

$O(n \log n)$

Ex.) Input: [7, 8, 9, 11, 12]
Output: 1

Input: [3, 5, 1]
Output: 2

result:	1	result:	1
sum	3	sum:	8
size	1	size:	2

If we tracked the sum & size we could get average, but how does this help us determine if we have seen next smallest value?

Let's simplify array.

Input: [11, 13, -1, 0, 2, 3]

1) Change all negative values to 'x'

2) Change all positive values > size of input to 'x'

Input: [x, x, x, x, 2, 3]

[8, 3, 7, 4, 5]

[x, 3, x, 4, 5]

pot sum. 15 $\underline{1}\ \underline{2}\ 3\ 4\ 5$

act. sum. 12

diff: 3

$1\ \underline{2}\ 3\ \underline{4}\ 5$

15 - 9 = 6

Two possible: 1, 2, 3 or 4, 2

But We Know size!

$1\ \underline{2}\ 3\ \underline{4}\ 5$

Pot. Sum : 15 Size : 3

Act. Sum : 9 Size : 2

Diff : 6

[3, 1]

Sum : 3 sum : 2

result : 1 result : 2

[5, 3, 4, 1]

Sum:	5	8	12
res :	1	1	1

Total pot: $\sum_{i=1}^{n} i = 1+2+3+4+5$

$= 15$

Sum of all values: 5+3+4+1

: 13

Total pot - act. sum = ans.

This only works if contiguous

[2 4 6 8 10] [2 3 4 5 6] Pot sum:

$\frac{n(n+1)}{2}$ 15

[2 4]

If no items elim.

Then calc.

[2]

3 - 2 = 1

[0 1 3 4 7] totalF = 1

[0 1 3 4 0] totaF = 2

[0 1 3 4 0] totaF = 3

[0 1 3 0 0] totaF = 4

[0 1 0 0 0] totaF = 4

- Sum

- $\frac{n(n+1)}{2}$ - Sum = answer

[2 5 7 8] total F = 0

[2 0 0 0] total F = 3

[0 0 0 0] total F = 4

It is still $O(n^2)$ if everything is linear.

[2 5 7 8]

1) Set all values < 0 or >= len(input) to 0

2) All values left are within index of input

3.) For all values not 0,

input[input[i] - 1] *= is it > 0 ? 1 : -1

4) For first input[i] > 0, return i + 1

[-2 -4 7 1]

1.) Make all values <= 0 size+1

[4 4 7 1]

i=0

num= 4

is abs(num) <= size? ✓

val = input[abs(input[i])-1]

= input[4 - 1]

= 1

is val > 0 ? ✓

val = -1

[4 4 7 -1]

i= 1

num= 4

is abs(num) <= size ? ✓

val = input[abs (input[i])-1]

= input[4-1]

= -1

is val > 0 ? X

[4 4 7 -1]

i=2

num= 7

is abs(num] <= size ? X

[4 4 1 -1]

i = 3

num = -1

is abs(num) <= size ? ✓

val = input[abs(input[i])-1}

= input[1-1]

= 4

is val > 0 ? ✓

val = -4

[-4 4 7 -1]

i = 1

Answer is i+1

1.) Make all positive values on left
Make all negative values on right

2.) For all positive values, if value - 1 is < len(+nums)
mark input[value - 1] = -input[value - 1]
(-) indicates it is marked.
So make sure we grab abs(value)

3.) First positive value is answer + 1