41. First Missing Positive

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

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

Example 1:

```
Input: nums = [1,2,0]
Output: 3
```

Example 2:

```
Input: nums = [3,4,-1,1]
Output: 2
```

Example 3:

```
Input: nums = [7,8,9,11,12]
Output: 1
```

```
def firstMissingPositive(self, nums: List[int]) -> int:
        one  = False
        n = len(nums)
        for i in range(n):
            if nums[i]==1:
                one = True
            if nums[i]<=0 or nums[i]>n:
                nums[i] = 1
        if one is False:
            return 1

        for i in range(n):
            idx = abs(nums[i])
            nums[idx-1] = -abs(nums[idx-1])

        for i in range(n):
            if nums[i]>0:
                return i+1
        return n+1
```

Method:2

```python
def firstMissingPositive(self, nums: List[int]) -> int:
        freqMap = {}
        for ele in nums:
            if ele>0:
                freqMap[ele] = True

        if len(freqMap)==0:
            return 1
        n = max(nums)
        for i in range(1,n+1):
            if i in freqMap:
                continue
            else:
                return i
        return n+1
```