Daily Coding Problem #4

Problem

This problem was asked by Stripe.

Given an array of integers, find the first missing positive integer in linear time and constant space. In other words, find the lowest positive integer that does not exist in the array. The array can contain duplicates and negative numbers as well.

For example, the input `[3, 4, -1, 1]` should give 2. The input `[1, 2, 0]` should give 3.

You can modify the input array in-place.

Solution

Our lives would be easier without the linear time constraint: we would just sort the array, while filtering out negative numbers, and iterate over the sorted array and return the first number that doesn't match the index. However, sorting

takes O(n log n), so we can't use that here.

Clearly we have to use some sort of trick here to get it running in linear time. Since the first missing positive number must be between 1 and len(array) + 1 (why?), we can ignore any negative numbers and numbers bigger than len(array). The basic idea is to use the indices of the array itself to reorder the elements to where they should be. We traverse the array and swap elements between 0 and the length of the array to their value's index. We stay at each index until we find that index's value and keep on swapping.

By the end of this process, all the first positive numbers should be grouped in order at the beginning of the array. We don't care about the others. This only takes O(N) time, since we swap each element at most once.

Then we can iterate through the array and return the index of the first number that doesn't match, just like before.

```
def first_missing_positive(nums):
    if not nums:
        return 1
    for i, num in enumerate(nums):
        while i + 1 != nums[i] and 0 < nums[i] <= len(nums):
            v = nums[i]
            nums[i], nums[v - 1] = nums[v - 1], nums[i]
            if nums[i] == nums[v - 1]:
                break
    for i, num in enumerate(nums, 1):
        if num != i:
            return i
    return len(nums) + 1
```

Another way we can do this is by adding all the numbers to a set, and then use a counter initialized to 1. Then continuously increment the counter and check whether the value is in the set.

```
def first_missing_positive(nums):
```

```
    s = set(nums)
    i = 1
    while i in s:
        i += 1
    return i
```

This is much simpler, but runs in O(N) time and space, whereas the previous algorithm uses no extra space.