**Problem # 969: Pancake Sorting**

<https://leetcode.com/problems/pancake-sorting/>

Like Bubble Sort. We cannot swap individual elements. We need to swap subarrays.

Note: We start with the whole array at the beginning. So subarray will be the whole array.

After one iteration, the subarray is the first to the last but one element, since the last element will already be in its correct place

1) Find the index of the maximum element of the subarray.

2) Take the subarray up to and including the maximum element and flip it so that the maximum element is at the beginning of the subarray.

Note 1: This step and the following steps need not be if the maximum element is already in the correct position.

Note 2: If the maximum element is already in the beginning of the array, this step need not be done.

3) Add the index of the maximum element + 1 to the result list since we are using 1-based indexing and not 0-based indexing.

4) Flip the array up to the correct position of the maximum element. In the first iteration, we will flip it to the right end of the array. In the second iteration, the second maximum element will be to the second position from the right, and so on.

5) Add the index of the correct position of maximum element + 1 to the result list since we are using 1-based indexing and not 0-based indexing.

\*\* Note: When the last two elements are sorted, the last element is automatically in the correct place.

**Solution:**

class Solution:

def pancakeSort(self, A: List[int]) -> List[int]:

n = len(A)

result = []

for i in range(n):

max\_index = A.index(max(A[0: n - i]))

if max\_index != n - 1 -i :

if max\_index != 0: #if max\_index is not at index 0 already, then flip and get it there

result.append(max\_index + 1) # one based indexing

A[0: max\_index + 1] = A[0: max\_index + 1][::-1]

A[0: n - i] = A[0: n - i][::-1] # put max\_index in the correct place

result.append(n - i) # we are using one based indexin

return(result)

**Solution** (using inline function for doing the subarray flip):

class Solution:

def pancakeSort(self, A: List[int]) -> List[int]:

result = []

def flip(idx):

for i in range(idx//2 + 1):

A[i], A[idx -i] = A[idx - i], A[i]

n = len(A)

for i in range(n - 1, 0, -1):

max\_index = A.index(max(A[0: i + 1]))

if max\_index != i :

if max\_index != 0: # if max\_index is not in position 1 already, then bring it to position 1 by flipping

flip(max\_index)

result.append(max\_index + 1) # we are using one-based index

flip(i)

result.append(i + 1) # we are using one-based index

return(result)

========================================================