Problem #456: 132 Pattern (Medium)

-------------------------------------------------

<https://leetcode.com/problems/132-pattern/>

My Solution:

My Solution used dictionary and not stack.

Runtime beats 9.42%

1. If there are less than 3 unique elements in the list, return False since we need

at least 3 number to form the pattern.

2. If the numbers in the list are sorted in ascending or descending order, we won’t be able to

form the pattern. So return False.

3. Set the min\_val to the first element at index 0, and the remaining elements put them in a dictionary called right\_dict with the key to be the element and the value is the number of occurrence of the element (i.e. frequency).

4. Now iterate through nums array starting with the element at index 1.

5. Calculate min\_val to be the minimum of min\_val and the previous element (i.e. element at index = i – 1). Delete the current element at index i from right\_dict.

6. If min\_val is less than current element (i.e. nums[i]), then there is a possibility that this could have the 132 pattern.

7. Get the right\_keys in sorted order from the right\_dict.

8. Since the right\_keys are sorted in ascending order, right\_keys[0] is the smallest element and right\_keys[-1] is the largest element. If the smallest element is greater than the current element or if the largest element is smaller than min\_val, then there is no possibility for the 132 pattern. So continue.

9. However, if this is not the case, iterate through the keys (i.e. unique elements ) to check if the 132 pattern is present. If so, return True.

10. After iterating through nums array, if the 132 pattern is not found, return False.

------------------------------------------------

from collections import Counter

class Solution:

def find132pattern(self, nums: List[int]) -> bool:

if len(set(nums)) < 3:

return False

sorted\_nums = sorted(nums)

if nums == sorted\_nums or nums == sorted\_nums[::-1]:

return False

min\_val = nums[0]

right\_dict = Counter(nums[1: ])

for i in range(1, len(nums) -1):

min\_val = min(nums[i-1], min\_val)

right\_dict[nums[i]] -= 1

if right\_dict[nums[i-1]] == 0:

del right\_dict[nums[i-1]]

if min\_val < nums[i]:

right\_keys = sorted(right\_dict.keys())

if right\_keys[0] > nums[i] or right\_keys[-1] < min\_val:

continue

else:

for key in right\_keys:

if key > min\_val and key < nums[i]:

return True

return False

---------------

I found a better solution online using stack – Voted the best in discussions.

<https://leetcode.com/problems/132-pattern/discuss/94081/10-line-Python-Solution>

class Solution(object):

def find132pattern(self, nums):

"""

:type nums: List[int]

:rtype: bool

"""

import sys

stack = []

s3 = -sys.maxint

for n in nums[::-1]:

if n < s3:

return True

while stack and stack[-1] < n:

s3 = stack.pop()

stack.append(n)

return False