Longest-Consecutive-Subsequence

May 20, 2020

0.0.1 Problem Statement

Given a list of integers that contain natural numbers in random order. Write a program to find the longest possible sub sequence of consecutive numbers in the array. Return this subsequence in sorted order.

In other words, you have to return the sorted longest (sub) list of consecutive numbers present anywhere in the given list.

For example, given the list `5, 4, 7, 10, 1, 3, 55, 2`, the output should be `1, 2, 3, 4, 5`

Note 1. The solution must take O(n) time. *Can you think of using a dictionary here?* 2. If two subsequences are of equal length, return the subsequence whose index of smallest element comes first.

0.0.2 The Idea:

Every element of the given `input_list` could be a part of some subsequence. Therefore, we need a way (using a dictionary) to keep track if an element has already been visited. Also, store length of a subsequence if it is maximum. For this purpose, we have to check in **forward** direction, if the `(element+1)` is available in the given dictionary, in a "while" loop. Similarly, we will check in **backward** direction for `(element-1)`, in another "while" loop. At last, if we have the smallest element and the length of the longest subsequence, we can return a **new** list starting from "smallest element" to "smallest element + length".

The steps would be:

1. Create a dictionary, such that the elements of input_list become the "key", and the corresponding index become the "value" in the dictionary. We are creating a dictionary because the lookup time is considered to be constant in a dictionary.
2. For each `element` in the `input_list`, first mark it as visited in the dictionary

- Check in forward direction, if the `(element+1)` is available. If yes, increment the length of subsequence
- Check in backward direction, if the `(element-1)` is available. If yes, increment the length of subsequence
- Keep a track of length of longest subsequence visited so far. For the longest subsequence, store the smallest element (say `start_element`) and its index as well.

3. Return a **new** list starting from `start_element` to `start_element + length`.

0.0.3 Exercise - Write the function definition here

```
In [1]: def longest_consecutive_subsequence(input_list):
            pass
```

0.0.4 Test - Let's test your function

```
In [ ]: def test_function(test_case):
            output = longest_consecutive_subsequence(test_case[0])
            if output == test_case[1]:
                print("Pass")
            else:
                print("Fail")
```

```
In [ ]: test_case_1 = [[5, 4, 7, 10, 1, 3, 55, 2], [1, 2, 3, 4, 5]]
        test_function(test_case_1)
```

```
In [ ]: test_case_2 = [[2, 12, 9, 16, 10, 5, 3, 20, 25, 11, 1, 8, 6 ], [8, 9, 10, 11, 12]]
        test_function(test_case_2)
```

```
In [ ]: test_case_3 = [[0, 1, 2, 3, 4], [0, 1, 2, 3, 4]]
        test_function(test_case_3)
```

Show Solution