Karan S. Chauhan · Follow

Feb 22, 2021 · 3 min read

Validate Subsequence with Python

This is a problem that asks us to create a function that will check to see if a sequence of numbers is a subsequence of a given array.

A subsequence of an array is a set of integers that don't have to be adjacent in the given array but are in the same order as they appear in the array.

For example, given an array [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], a valid subsequence

Something to note is that a single number in an array and the array itself are both valid subsequences of a given array input.

With that information, let's start analyzing and solving the problem.

The Prompt

Given two non-empty arrays of integers, write a function that determines whether the second array is a subsequence of the first one.

The Code

Let's start by initializing the function with the correct parameters, an array and a sequence of numbers.

```
1 ▾ def isValidSubsequence(array, sequence):
```

What we can do next is create two variables that will represent the indices:



- seqIdx → for the sequence's index

We will initialize both variables, set to 0, to compare the the individual elements in both arguments.

```
1 ▾ def isValidSubsequence(array, sequence):
2         arrIdx = 0
3         seqIdx = 0
```

Next up will be the logic. We need to compare the elements in the array and the sequence. In order to accomplish this, we'll use the two index variables created earlier to see if there is a match.

If there is a match, then we'll increment the sequence index by 1. Then after successfully incrementing the sequence index, we can increment the array index.

This logic is great for the purposes of this problem, but unless there are constraints, the code will be running continuously. Therefore, we need to implement a while loop and wrap the index incrementing logic within the while

```python
def isValidSubsequence(array, sequence):
    arrIdx = 0
    seqIdx = 0
    while arrIdx < len(array) and seqIdx < len(sequence):
        if array[arrIdx] == sequence[seqIdx]:
            seqIdx += 1
        arrIdx += 1
```

This way, the code will run as long as the lengths of the array and sequence. If there is a matching element in both the array and sequence, then the sequence index will be incremented. Once that successfully occurs, then the array index will be incremented.

To finalize this solution, we can return a boolean value of the comparison between the sequence index and the length of the sequence. At the end of the logic, the two should be equal and will return the value of TRUE.

The final code will look like the following:

```
    seqIdx = 0
    while arrIdx < len(array) and seqIdx < len(sequence):
        if array[arrIdx] == sequence[seqIdx]:
            seqIdx += 1
        arrIdx += 1
    return seqIdx == len(sequence)
```

Code Analysis

The time complexity of this solution is O(n) time, where n is the length of the array.

The space complexity of this solution is O(1) space.