873. Length of Longest Fibonacci Subsequence

Medium Topics Companies

A sequence `x_1, x_2, ..., x_n` is *Fibonacci-like* if:

- `n >= 3`
- `x_i + x_{i+1} == x_{i+2}` for all `i + 2 <= n`

Given a **strictly increasing** array `arr` of positive integers forming a sequence, return *the **length** of the longest Fibonacci-like subsequence of* `arr`. If one does not exist, return `0`.

A **subsequence** is derived from another sequence `arr` by deleting any number of elements (including none) from `arr`, without changing the order of the remaining elements. For example, `[3, 5, 8]` is a subsequence of `[3, 4, 5, 6, 7, 8]`.

Example 1:

```
Input: arr = [1,2,3,4,5,6,7,8]
Output: 5
Explanation: The longest subsequence that is fibonacci-like: [1,2,3,5,8].
```

Example 2:

```
Input: arr = [1,3,7,11,12,14,18]
Output: 3
Explanation: The longest subsequence that is fibonacci-like: [1,11,12], [3,11,14]
or [7,11,18].
```

← All Submissions

Accepted 57 / 57 testcases passed

AndrewC275 submitted at Feb 27, 2025 10:07

Editorial Solution

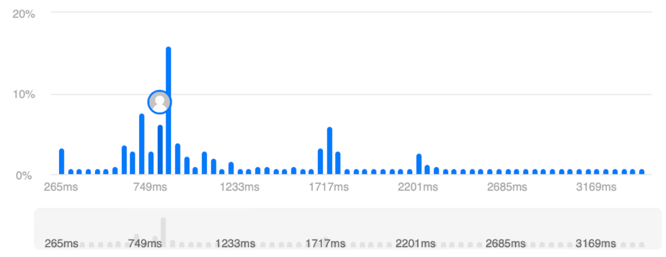
Runtime

836 ms | Beats 65.56%

Analyze Complexity

Memory

17.95 MB | Beats 80.79%



Code | Python3

Python3 Auto

```python
class Solution:
    def lenLongestFibSubseq(self, arr: List[int]) -> int:
        nums = set(arr)

        res = 0

        for i in range(len(arr)-1):
            for j in range(i+1, len(arr)):
                c1, c2 = arr[i], arr[j]
                nxt = c1 + c2
                count = 2
                while nxt in nums:
                    count += 1
                    res = max(res, count)
                    c1, c2 = c2, nxt
                    nxt = c1 + c2

        return res

```

Ln 10, Col 30 | Saved

Testcase | Test Result

roughly $O(n^2 \cdot \underline{\log_2 N})$

- for Fibonacci, almost doubling the next num.

- N : max(arr) or len(arr)

$a,\ b,\ a+b,\ a+2b,\ 2a+3b,$

$3a+5b,\ \ldots$