# Problem 873: Length of Longest Fibonacci Subsequence

## Problem Information

**Difficulty:** Medium
**Acceptance Rate:** 57.56%
**Paid Only:** No
**Tags:** Array, Hash Table, Dynamic Programming

## Problem Description

A sequence `x1, x2, ..., xn` is _Fibonacci-like_ if:

* `n >= 3` * `xi + xi+1 == xi+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].

**Constraints:**

* `3 <= arr.length <= 1000` * `1 <= arr[i] < arr[i + 1] <= 109`

## Code Snippets

**C++:**

```
class Solution {
public:
int lenLongestFibSubseq(vector<int>& arr) {

}
};
```

**Java:**

```
class Solution {
public int lenLongestFibSubseq(int[] arr) {

}
}
```

**Python3:**

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