@LeetCode

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 `A` of positive integers forming a sequence, find the **length** of the longest fibonacci-like subsequence of `A`. If one does not exist, return 0.

(*Recall that a subsequence is derived from another sequence* `A` *by deleting any number of elements (including none) from* `A`*, 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: [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: [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].
```

Note:

- `3 <= A.length <= 1000`
- `1 <= A[0] < A[1] < ... < A[A.length - 1] <= 10^9`
- *(The time limit has been reduced by 50% for submissions in Java, C, and C++.)*