904. Fruit Into Baskets

Solved

Medium | Topics | Companies

You are visiting a farm that has a single row of fruit trees arranged from left to right. The trees are represented by an integer array `fruits` where `fruits[i]` is the **type** of fruit the `i^th` tree produces.

You want to collect as much fruit as possible. However, the owner has some strict rules that you must follow:

- You only have **two** baskets, and each basket can only hold a **single type** of fruit. There is no limit on the amount of fruit each basket can hold.
- Starting from any tree of your choice, you must pick **exactly one fruit** from **every** tree (including the start tree) while moving to the right. The picked fruits must fit in one of your baskets.
- Once you reach a tree with fruit that cannot fit in your baskets, you must stop.

Given the integer array `fruits`, return *the **maximum** number of fruits you can pick*.

Example 1:

> **Input:** fruits = [1,2,1]
> **Output:** 3
> **Explanation:** We can pick from all 3 trees.

```python
class Solution:
    def totalFruit(self, fruits: List[int]) -> int:
        # find the longest continous subarry with given two keys, sliding window
        basket = {}
        res = 0
        left = 0

        for right in range(len(fruits)):
            basket[fruits[right]] = basket.get(fruits[right], 0) + 1

            while len(basket) > 2:
                basket[fruits[left]] -= 1
                if basket[fruits[left]] == 0:
                    del basket[fruits[left]]
                left += 1

            res = max(res, right - left + 1)

        return res
```

Standard sliding window

```python
class Solution:
    def totalFruit(self, fruits: List[int]) -> int:
        # find the longest continous subarry with given two keys, sliding window
        basket = {}
        left = 0

        for right, f in enumerate(fruits):
            basket[f] = basket.get(f, 0) + 1

            if len(basket) > 2:
                basket[fruits[left]] -= 1

                if basket[fruits[left]] == 0:
                    del basket[fruits[left]]

                left += 1

        return right - left + 1
```

Smarter method:

no need to consider the situation when basket length > 2 and right - left + 1 < existing result