

## Fruits into Baskets (medium)

### Problem Statement #

Given an array of characters where each character represents a fruit tree, you are given **two baskets** and your goal is to put **maximum number of fruits in each basket**. The only restriction is that **each basket can have only one type of fruit**.

You can start with any tree, but once you have started you can't skip a tree. You will pick one fruit from each tree until you cannot, i.e., you will stop when you have to pick from a third fruit type.

Write a function to return the maximum number of fruits in both the baskets.

### Example 1:

Input: Fruit=['A', 'B', 'C', 'A', 'C']

Output: 3

Explanation: We can put 2 'C' in one basket and one 'A' in the other from the subarray ['C', 'A', 'C']

### Example 2:

Input: Fruit=['A', 'B', 'C', 'B', 'B', 'C']

Output: 5

Explanation: We can put 3 'B' in one basket and two 'C' in the other basket.

This can be done if we start with the second letter: ['B', 'C', 'B', 'B', 'C']

### Try it yourself #

Try solving this question here:

Java

Python3

JS

C++

```

1  using namespace std;
2
3  #include <iostream>
4  #include <unordered_map>
5  #include <vector>
6
7  class MaxFruitCountOf2Types {
8  public:
9      static int findLength(const vector<char>& arr) {
10         int maxLength = 0;
11         // TODO: Write your code here
12         return maxLength;
13     }
14 };
15

```

Show Results

Show Console

×

0 of 2 Tests Passed

Result	Input	Expected Output	Actual Output	Reason
✗	fruits_into_baskets([A, B, C, A, C])	3	-1	Incorrect Output
✗	fruits_into_baskets([A, B, C, B, B, C])	5	-1	Incorrect Output

0.468s

## Solution #

This problem follows the **Sliding Window** pattern and is quite similar to [Longest Substring with K Distinct Characters](#). In this problem, we need to find the length of the longest subarray with no more than two distinct characters (or fruit types!). This transforms the current problem into **Longest Substring with K Distinct Characters** where  $K=2$ .

## Code #

Here is what our algorithm will look like, only the highlighted lines are different from [Longest Substring with K Distinct Characters](#):

Java

Python3

C++

JS

```
1 using namespace std;
2
3 #include <iostream>
4 #include <unordered_map>
5 #include <vector>
6
7 class MaxFruitCountOf2Types {
8 public:
9     static int findLength(const vector<char>& arr) {
10         int windowStart = 0, maxLength = 0;
11         unordered_map<char, int> fruitFrequencyMap;
12         // try to extend the range [windowStart, windowEnd]
13         for (int windowEnd = 0; windowEnd < arr.size(); windowEnd++) {
14             fruitFrequencyMap[arr[windowEnd]]++;
15             // shrink the sliding window, until we are left with '2' fruits in the frequency map
16             while ((int)fruitFrequencyMap.size() > 2) {
17                 fruitFrequencyMap[arr[windowStart]]--;
18                 if (fruitFrequencyMap[arr[windowStart]] == 0) {
19                     fruitFrequencyMap.erase(arr[windowStart]);
20                 }
21                 windowStart++; // shrink the window
22             }
23             maxLength = max(maxLength, windowEnd - windowStart + 1);
24         }
25
26         return maxLength;
27     }
28 };
29
30 int main(int argc, char* argv[]) {
31     cout << "Maximum number of fruits: "
32           << MaxFruitCountOf2Types::findLength(vector<char>{'A', 'B', 'C', 'A', 'C'}) << endl;
33     cout << "Maximum number of fruits: "
34           << MaxFruitCountOf2Types::findLength(vector<char>{'A', 'B', 'C', 'B', 'B', 'C'}) << endl;
35 }
36
```



Output

✕

1.222s

Maximum number of fruits: 3

Maximum number of fruits: 5

### Time Complexity #

The time complexity of the above algorithm will be  $O(N)$  where 'N' is the number of characters in the input array. The outer **for** loop runs for all characters and the inner **while** loop processes each character only once, therefore the time complexity of the algorithm will be  $O(N + N)$  which is asymptotically equivalent to  $O(N)$ .

### Space Complexity #

The algorithm runs in constant space  $O(1)$  as there can be a maximum of three types of fruits stored in the frequency map.

### Similar Problems #

#### **Problem 1: Longest Substring with at most 2 distinct characters**

Given a string, find the length of the longest substring in it with at most two distinct characters.

**Solution:** This problem is exactly similar to our parent problem.