JS

function fruits\_into\_baskets(fruits) {

let windowStart = 0,

maxLength = 0,

fruitFrequency = {};

// try to extend the range [windowStart, windowEnd]

for (let windowEnd = 0; windowEnd < fruits.length; windowEnd++) {

const rightFruit = fruits[windowEnd];

if (!(rightFruit in fruitFrequency)) {

fruitFrequency[rightFruit] = 0;

}

fruitFrequency[rightFruit] += 1;

// shrink the sliding window, until we are left with '2' fruits in the fruit frequency dictionary

while (Object.keys(fruitFrequency).length > 2) {

const leftFruit = fruits[windowStart];

fruitFrequency[leftFruit] -= 1;

if (fruitFrequency[leftFruit] === 0) {

delete fruitFrequency[leftFruit];

}

windowStart += 1; // shrink the window

}

maxLength = Math.max(maxLength, windowEnd - windowStart + 1);

}

return maxLength;

}

console.log(`Maximum number of fruits: ${fruits\_into\_baskets(['A', 'B', 'C', 'A', 'C'])}`);

console.log(`Maximum number of fruits: ${fruits\_into\_baskets(['A', 'B', 'C', 'B', 'B', 'C'])}`);

C++

using namespace std;

#include <iostream>

#include <unordered\_map>

#include <vector>

class MaxFruitCountOf2Types {

public:

static int findLength(const vector<char>& arr) {

int windowStart = 0, maxLength = 0;

unordered\_map<char, int> fruitFrequencyMap;

// try to extend the range [windowStart, windowEnd]

for (int windowEnd = 0; windowEnd < arr.size(); windowEnd++) {

fruitFrequencyMap[arr[windowEnd]]++;

// shrink the sliding window, until we are left with '2' fruits in the frequency map

while ((int)fruitFrequencyMap.size() > 2) {

fruitFrequencyMap[arr[windowStart]]--;

if (fruitFrequencyMap[arr[windowStart]] == 0) {

fruitFrequencyMap.erase(arr[windowStart]);

}

windowStart++; // shrink the window

}

maxLength = max(maxLength, windowEnd - windowStart + 1);

}

return maxLength;

}

};

int main(int argc, char\* argv[]) {

cout << "Maximum number of fruits: "

<< MaxFruitCountOf2Types::findLength(vector<char>{'A', 'B', 'C', 'A', 'C'}) << endl;

cout << "Maximum number of fruits: "

<< MaxFruitCountOf2Types::findLength(vector<char>{'A', 'B', 'C', 'B', 'B', 'C'}) << endl;

}

Python:

def fruits\_into\_baskets(fruits):

window\_start = 0

max\_length = 0

fruit\_frequency = {}

# try to extend the range [window\_start, window\_end]

for window\_end in range(len(fruits)):

right\_fruit = fruits[window\_end]

if right\_fruit not in fruit\_frequency:

fruit\_frequency[right\_fruit] = 0

fruit\_frequency[right\_fruit] += 1

# shrink the sliding window, until we are left with '2' fruits in the fruit frequency dictionary

while len(fruit\_frequency) > 2:

left\_fruit = fruits[window\_start]

fruit\_frequency[left\_fruit] -= 1

if fruit\_frequency[left\_fruit] == 0:

del fruit\_frequency[left\_fruit]

window\_start += 1 # shrink the window

max\_length = max(max\_length, window\_end-window\_start + 1)

return max\_length

def main():

print("Maximum number of fruits: " + str(fruits\_into\_baskets(['A', 'B', 'C', 'A', 'C'])))

print("Maximum number of fruits: " + str(fruits\_into\_baskets(['A', 'B', 'C', 'B', 'B', 'C'])))

main()

JAVA:

import java.util.\*;

class MaxFruitCountOf2Types {

public static int findLength(char[] arr) {

int windowStart = 0, maxLength = 0;

Map<Character, Integer> fruitFrequencyMap = new HashMap<>();

// try to extend the range [windowStart, windowEnd]

for (int windowEnd = 0; windowEnd < arr.length; windowEnd++) {

fruitFrequencyMap.put(arr[windowEnd], fruitFrequencyMap.getOrDefault(arr[windowEnd], 0) + 1);

// shrink the sliding window, until we are left with '2' fruits in the frequency map

while (fruitFrequencyMap.size() > 2) {

fruitFrequencyMap.put(arr[windowStart], fruitFrequencyMap.get(arr[windowStart]) - 1);

if (fruitFrequencyMap.get(arr[windowStart]) == 0) {

fruitFrequencyMap.remove(arr[windowStart]);

}

windowStart++; // shrink the window

}

maxLength = Math.max(maxLength, windowEnd - windowStart + 1);

}

return maxLength;

}

public static void main(String[] args) {

System.out.println("Maximum number of fruits: " +

MaxFruitCountOf2Types.findLength(new char[] { 'A', 'B', 'C', 'A', 'C' }));

System.out.println("Maximum number of fruits: " +

MaxFruitCountOf2Types.findLength(new char[] { 'A', 'B', 'C', 'B', 'B', 'C' }));

}

}