Back To Course Home

Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Search Course

(hard) (/courses/grokking-the-coding-interview/R8DVgjq78yR)

Longest Subarray with Ones after Replacement (hard) (/courses/grokking-the-coding-interview/B6VypRxPoIJ)

Problem Challenge 1 (/courses/grokking-the-coding-interview/N8vB7OVYo2D)

Solution Review: Problem Challenge 1 (/courses/grokking-the-coding-interview/N0o9QnPLKNv)

Problem Challenge 2 (/courses/grokking-the-coding-interview/YQ8N2OZq0VM)

Longest Subarray with Ones after Replacement (hard)

We'll cover the following

Problem Statement

Given an array containing 0s and 1s, if you are allowed to **replace no more than 'k' 0s with 1s**, find the length of the **longest contiguous subarray having all 1s**.

Example 1:

Back To Course Home

Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Search Course

(hard) (/courses/grokking-the-coding-interview/R8DVgjq78yR)

Longest Subarray with Ones after Replacement (hard) (/courses/grokking-the-coding-interview/B6VypRxPolJ)

Problem Challenge 1 (/courses/grokking-the-coding-interview/N8vB7OVYo2D)

Solution Review: Problem Challenge 1 (/courses/grokking-the-coding-interview/N0o9QnPLKNv)

Problem Challenge 2 (/courses/grokking-the-coding-interview/YQ8N2OZq0VM)

```
Input: Array=[0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1], k=2
Output: 6
Explanation: Replace the '0' at index 5 and 8 to have the longest contiguous suba
rray of 1s having length 6.
```

Example 2:

```
Input: Array=[0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1], k=3
Output: 9
Explanation: Replace the '0' at index 6, 9, and 10 to have the longest contiguou
s subarray of 1s having length 9.
```

Try it yourself

Try solving this question here:

Java | Python3 | JS | C++

```
1  class ReplacingOnes {
2    public static int findLength(int[] arr, int k) {
3      // TODO: Write your code here
4      int start = 0, end = 0, ans =0, onesCount =0 ;
5      int n = arr.length;
6
7      while(end<n)
8      {
9        if(arr[end] == 1)
10       {
11         onesCount++;
12       }
13       if(end-start+1 - onesCount > k)
14       {
15         if(arr[start] == 1)
16         {
17           onesCount--;
```

Back To Course Home

Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Search Course


```
18          }
19          start++;
20        }
21        ans = Math.max(ans , end-start+1);
22        end++;
23      }
24      return ans;
25    }
26  }
27
```

Test | Save * | Reset

Show Results | Show Console

2 of 2 Tests Passed

Result	Input	Expected Output	Actual Output	Reason
✓	findLength([0, 1, 1, 0, 0, 0, 1, 1, 0, 1 …	6	6	Succeeded
✓	findLength([0, 1, 0, 0, 1, 1, 0, 1, 1, 0 …	9	9	Succeeded

5.247s

Solution

Back To Course Home

Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Search Course

(hard) (/courses/grokking-the-coding-interview/R8DVgjq78yR)

Longest Subarray with Ones after Replacement (hard) (/courses/grokking-the-coding-interview/B6VypRxPoIJ)

Problem Challenge 1 (/courses/grokking-the-coding-interview/N8vB7OVYo2D)

Solution Review: Problem Challenge 1 (/courses/grokking-the-coding-interview/N0o9QnPLKNv)

Problem Challenge 2 (/courses/grokking-the-coding-interview/YQ8N2OZq0VM)

This problem follows the **Sliding Window** pattern and is quite similar to Longest Substring with same Letters after Replacement (https://www.educative.io/collection/page/5668639101419520/5671464854355968/6497958910492672/). The only difference is that, in the problem, we only have two characters (1s and 0s) in the input arrays.

Following a similar approach, we'll iterate through the array to add one number at a time in the window. We'll also keep track of the maximum number of repeating 1s in the current window (let's call it `maxOnesCount`). So at any time, we know that we can have a window with 1s repeating `maxOnesCount` time, so we should try to replace the remaining 0s. If we have more than 'k' remaining 0s, we should shrink the window as we are not allowed to replace more than 'k' 0s.

Code

Here is how our algorithm will look like:

Java | Python3 | C++ | JS

```
class ReplacingOnes {
  public static int findLength(int[] arr, int k) {
    int windowStart = 0, maxLength = 0, maxOnesCour
    // try to extend the range [windowStart, windov
    for (int windowEnd = 0; windowEnd < arr.length;
      if (arr[windowEnd] == 1)
        maxOnesCount++;

      // current window size is from windowStart to
      // repeating a maximum of 'maxOnesCount' time
      // 'maxOnesCount' 1s and the remaining are 0s
      // now, if the remaining 0s are more than 'k'
      // are not allowed to replace more than 'k' 0
      if (windowEnd - windowStart + 1 - maxOnesCour
        if (arr[windowStart] == 1)
```

```
16            maxOnesCount--;
17          windowStart++;
18        }
19
20        maxLength = Math.max(maxLength, windowEnd - w
21      }
22
23      return maxLength;
24    }
25
26    public static void main(String[] args) {
27      System.out.println(ReplacingOnes.findLength(new
28      System.out.println(ReplacingOnes.findLength(new
29    }
30  }
31
```

Run Save Reset

Time Complexity

The above algorithm's time complexity will be $O(N)$, where 'N' is the count of numbers in the input array.

Space Complexity

The algorithm runs in constant space $O(1)$.

4% completed, meet the criteria and claim your course certificate!

Claim Certificate

Back To Course Home

Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Search Course

Report an Issue

Ask a Question (https://discuss.educative.io/tag/longest-subarray-with-ones-after-replacement-hard__pattern-sliding-window__grokking-the-coding-interview-patterns-for-coding-questions)