Full Name:	Sem Limi
Email:	mssemmi8@gmail.com
Test Name:	**Lambda School - Web Whiteboard Fitness Assessment**
Taken On:	29 Aug 2019 08:48:21 PDT
Time Taken:	237 min 55 sec/ 240 min
Invited by:	Josh
Tags Score:	Algorithms 240/250 Core Skills 50/50 Data Structures 65/75 Easy 100/100 Linked Lists 50/50 Medium 140/150 Problem Solving 190/200 Recursion 50/50

96%
240/250

scored in **Lambda School - Web Whiteboard Fitness Assessment** in 237 min 55 sec on 29 Aug 2019 08:48:21 PDT

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Fibonacci Sequence > Coding	29 min 41 sec	50/ 50	✓
Q2	Balanced Brackets > Coding	56 min 55 sec	65/ 75	✓
Q3	Remove Kth Linked List Node > Coding	1 hour 27 min 44 sec	50/ 50	✓
Q4	Three Number Sum > Coding	58 min 29 sec	75/ 75	✓

QUESTION 1

Correct Answer

Score 50

Fibonacci Sequence > Coding

Easy | Recursion | Algorithms | Problem Solving | Core Skills

QUESTION DESCRIPTION

The Fibonacci numbers are a sequence of numbers where each number after the first two is a sum of the prior two. As an illustration, here is an example sequence with $n = 8$: *(0, 1, 1, 2, 3, 5, 8, 13)*.

Given an integer *n*, calculate the first *n* numbers in the Fibonacci sequence. Return an array containing the *n* integers.

Function Description

Complete the function *fibonacci* in the editor below. The function must return an array of *n* Fibonacci

numbers starting with 0 as the first fibonacci number.

fibonacci has the following parameter(s):
n: integer, the length of the Fibonacci series to return

Constraints
- $1 \leq n \leq 10$

▶ Input Format for Custom Testing
▶ Sample Case 0

CANDIDATE ANSWER

Language used: **JavaScript (Node.js)**

```
/*
 * Complete the 'fibonacci' function below.
 *
 * The function is expected to return an INTEGER_ARRAY.
 * The function accepts INTEGER n as parameter.
 */

function fibonacci(n) {
    let fibArr = Array(n);

    if (n < 2) {
        fibArr[0] = 0;
    }
    else {
         fibArr[0] = 0;
    fibArr[1] = 1;

    for (let i = 2; i < n; i++) {
        fibArr[i] = fibArr[i - 1] + fibArr[i-2];
    }
    }
    return fibArr;
}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	5	0.1819 sec	36.1 KB
Testcase 1	Easy	Sample case	Success	5	0.1042 sec	36.4 KB
Testcase 3	Easy	Sample case	Success	8	0.1266 sec	36.2 KB
Testcase 4	Easy	Sample case	Success	8	0.1978 sec	36.6 KB
Testcase 4	Medium	Sample case	Success	10	0.1222 sec	35.8 KB
Testcase 5	Hard	Sample case	Success	14	0.0884 sec	35.9 KB

No Comments

QUESTION 2

Balanced Brackets › Coding

Algorithms | Problem Solving | Data Structures | Medium

Correct Answer

Score 65

QUESTION DESCRIPTION

Write a function that takes a string as input. The string can contain four types of brackets: "[]", "()", "{}", and "||". Your function should return a boolean indicating whether or not the string is balanced. Note that pipes brackets use the same character to indicate both opening and closing.

A string is considered balanced if it has as many opening brackets of a given type as it has closing brackets of that same type. No bracket can be left unmatched if the string is to be considered balanced. A closing bracket also cannot match a corresponding opening bracket that comes after it. Brackets also cannot overlap each other, i.e., "[(])".

Examples:
"[(])" should return false
"foo(bar)baz" should return true
"{{||[]||}}" should return true
"I am happy to take your donation; any amount will be greatly appreciated." should return true
"I (wa)n{t to buy a on}esie[…] b(u{[t] kno}w it) won't suit me." should return true
"This is t(he la[st random sentence I will be writing |and| I am going to stop mid-sent]" should return false

Note that HackerRank's expected output from your code will be a 0 indicating `false` or a 1 indicating `true`. You may opt to return 1 and 0 as your truthy and falsey values, or actual booleans `true` and `false`; either option will work. HackerRank's platform will coerce a `true` value to 1 and a `false` value to 0 behind the scenes.

CANDIDATE ANSWER

Language used: **JavaScript (Node.js)**

```
/*
 * Complete the 'balancedBrackets' function below.
 *
 * The function is expected to return a BOOLEAN.
 * The function accepts STRING string as parameter.
 */

function balancedBrackets(string) {
let i, ch;

let bracketsMap = new Map();
bracketsMap.set(']', '[');
bracketsMap.set('}', '{');
bracketsMap.set(')', '(');
bracketsMap.set('|', '|');

// Spread operator return map as a 2D key-value Array.
let closingBrackets = [...bracketsMap.keys()];
let openingBrackets = [...bracketsMap.values()];

let temp = [];
let len = string.length;

for (i = 0; i < len; i++) {
  ch = string[i];

  if (openingBrackets.indexOf(ch) > -1) {
    temp.push(ch);
  } else if (closingBrackets.indexOf(ch) > -1) {

    let expectedBracket = bracketsMap.get(ch);
    if (temp.length === 0 || (temp.pop() !== expectedBracket)) {
      return false;
    }
```

```
35
36    } else {
37      // Ignore the characters which do not match valid Brackets symbol
38      continue;
39    }
40  }
41
42  return (temp.length === 0);
43
44  }
45
46
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0851 sec	36.2 KB
Testcase 1	Easy	Sample case	Success	4	0.0907 sec	35.9 KB
Testcase 2	Easy	Sample case	Wrong Answer	0	0.0876 sec	36.1 KB
Testcase 3	Easy	Sample case	Success	4	0.0846 sec	36.3 KB
Testcase 4	Medium	Sample case	Success	11	0.0878 sec	36 KB
Testcase 5	Medium	Sample case	Success	15	0.0945 sec	36.4 KB
Testcase 6	Easy	Sample case	Success	6	0.0843 sec	35.5 KB
Testcase 7	Medium	Sample case	Success	15	0.0915 sec	36.1 KB

No Comments

QUESTION 3

Correct Answer

Score 50

Remove Kth Linked List Node › Coding

Linked Lists | Algorithms | Easy

QUESTION DESCRIPTION

Write a function that receives as input the head node of a linked list and an integer k. Your function should remove the kth node from the end of the linked list and return the head node of the updated list.

For example, if we have the following linked list:
`(20) -> (19) -> (18) -> (17) -> (16) -> (15) -> (14) -> (13) -> (12) -> (11) -> null`

The head node would refer to the node `(20)`. Let `k = 4`, so our function should remove the 4th node from the end of the linked list, the node `(14)`.

So after the function executes, the state of the linked list should be:
`(20) -> (19) -> (18) -> (17) -> (16) -> (15) -> (13) -> (12) -> (11) -> null`.

If k is longer than the length of the linked list, the linked list should not be changed.

The optimal solution exhibits a runtime complexity of O(n) and a space complexity of O(1).

How to Read the Input that is Used to Test Your Implementation
The way HackerRank's platform works, passing in a linked list, say the one above that was used in the example, as input to your implementation would look like the following:

```
10  <----- The length of the linked list comes first
20
```

19
18
17
16
15
14
13
12
11
4 <----- The k parameter indicating which node from the end we want removed

Note that the first element in the input is ***not*** part of the linked list itself. The first parameter of the input indicates the length of the linked list (10 elements), then the next 10 pieces of input each specify an element in the linked list in the prescribed order. The last parameter then is the k parameter indicating which node from the end we want removed from the list. Your function then, given the above input, would receive the node containing 20 as its `head` parameter and 4 as its `k` parameter.

Also note that the format of the expected output only includes the contents of the linked list. So the expected output of the above input would be:

20
19
18
17
16
15
13
12
11

Just the elements of the linked list and nothing else.

CANDIDATE ANSWER

Language used: **JavaScript (Node.js)**

```
/*
 * Complete the 'removeKthLinkedListNode' function below.
 *
 * The function is expected to return an INTEGER_SINGLY_LINKED_LIST.
 * The function accepts following parameters:
 *  1. INTEGER_SINGLY_LINKED_LIST head
 *  2. INTEGER k
 */

/*
 * For your reference:
 *
 * SinglyLinkedListNode {
 *     int data;
 *     SinglyLinkedListNode next;
 * }
```

```
 *
 */

function removeKthLinkedListNode(head, k) {
    var list = [],
        currentNode = head;

    while(currentNode.next !== null) {
        list.push(currentNode);
        currentNode = currentNode.next;
    }
    list.push(currentNode);


    if (list.length -k - 1 >= 0 && list.length -k + 1 < list.length) {
        list[list.length - k - 1].next = list[list.length -k + 1];
        return list[0];
    }

    if(list.length - k == 0) {
        return list[1]
    }

    if(list.length - k - 1 < 0) {
        return list.length <=1 ? [] : list[0]
    }



    if (list.length - k + 1 >= list.length) {
        list[list.length - k - 1 ].next = null;
        return list[0];
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0884 sec	36 KB
Testcase 1	Easy	Sample case	Success	10	0.0894 sec	35.8 KB
Testcase 2	Easy	Sample case	Success	10	0.0855 sec	35.9 KB
Testcase 3	Medium	Sample case	Success	20	0.0957 sec	36.1 KB

No Comments

QUESTION 4

Correct Answer

Score 75

Three Number Sum › Coding

Algorithms | Medium | Problem Solving

QUESTION DESCRIPTION

Write a function that takes in a non-empty array of distinct integers and a target integer. Your function should find all triplets in the array that sum up to the target sum and return a two-dimensional array of all these triplets. Each inner array containing a single triplet should have all three of its elements ordered in ascending order. The triplets themselves should be ordered in ascending order with respect to the first number of each triplet.

If no such triplets can be found in the array, your function should return an empty array.

Example 1:
Input: [12, 3, 1, 2, -6, 5, -8, 6], 0
Expected Output: [[-8, 2, 6], [-8, 3, 5], [-6, 1, 5]]

Example 2:
Input: [1, 2, 3, 4, 5, 6, 7, 8, 9, 15], 30
Expected Output: [[6, 9, 15], [7, 8, 15]]

How to Read the Input that is Used to Test Your Implementation
The way HackerRank's platform works, passing in an array of values, say the one above that was used in the example, as input to your implementation would look like the following:

```
16  <----- The length of the array comes first
19
27
28
33
39
41
46
49
2
5
6
10
13
14
15
17
33 <----- target sum
```

Note that the first element in the input is ***not*** part of the array itself. The first parameter of the input indicates the length of the array (16 elements), then the next 16 pieces of input each specify an element in the array in the prescribed order. The last parameter is the target element to search for.

Also note that the format of the expected output only includes the index of the element in the array, or -1 if the target element is not found in the array. So the expected output of the above input would be:

```
3 <----- index of the target element in the input array
```

CANDIDATE ANSWER

Language used: **JavaScript (Node.js)**

```
/*
 * Complete the 'threeNumberSum' function below.
 *
 * The function is expected to return a 2D_INTEGER_ARRAY.
 * The function accepts following parameters:
 *  1. INTEGER_ARRAY arr
```

```
 *  2. INTEGER target
 */

function threeNumberSum(arr, target) {
  arr.sort((a, b) => {
    return a - b;
  });

  const result = [];

  for (let leftIndex = 0; leftIndex < arr.length - 2; leftIndex++) {

    let middleIndex = leftIndex + 1;
    let rightIndex = arr.length - 1;

    if (leftIndex > target && arr[leftIndex] === arr[leftIndex - 1])
continue;

    while (middleIndex < rightIndex ) {

      let sum = arr[leftIndex] + arr[middleIndex] + arr[rightIndex];

      if (sum < target) {
        middleIndex++;
      } else if (sum > target) {
        rightIndex--;
      } else {
        result.push([arr[leftIndex], arr[middleIndex], arr[rightIndex]]);
        while (arr[middleIndex] === arr[middleIndex + 1]) middleIndex++;
        while (arr[rightIndex] === arr[rightIndex - 1]) rightIndex--
        middleIndex++;
        rightIndex--;
      }
    }
  }
  return result;

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0845 sec	36.1 KB
Testcase 1	Easy	Sample case	Success	10	0.0995 sec	35.9 KB
Testcase 2	Easy	Sample case	Success	10	0.0907 sec	36.2 KB
Testcase 3	Easy	Sample case	Success	10	0.0927 sec	36.1 KB
Testcase 4	Medium	Sample case	Success	35	0.1914 sec	38.6 KB

No Comments