Recursion Exercises

Download Starter Code <../dsa-recursion.zip>

All of these problems should be solved using recursion.

Product of Nums

Write a function that finds the product of an array of numbers:

```
product([2, 3, 4])   // 24
```

Longest Word

Given a list of words, return the *length* of the longest:

```
longest(["hello", "hi", "hola"])  // 5
```

Every Other Character

Write a function that returns a string of every other character:

```
everyOther("hello")  // "hlo"
```

Is Palindrome?

Write a function that returns true/false depending on whether passed-in string is a palindrome:

```
isPalindrome("tacocat")  // true
isPalindrome("tacodog")  // false
```

Find Index

Given an array and a string, return the index of that string in the array (or -1 if not present):

```
let animals = ["duck", "cat", "pony"];
```

```
findIndex(animals, "cat");  // 1
findIndex(animals, "porcupine");   // -1
```

Reverse String

Return a copy of a string, reversed:

```
revString("porcupine") // 'enipucrop'
```

Gather Strings

Given an object, return an array of all the values in the object that are strings:

```
let nestedObj = {
  firstName: "Lester",
  favoriteNumber: 22,
  moreData: {
    lastName: "Testowitz"
  },
  funFacts: {
    moreStuff: {
      anotherNumber: 100,
      deeplyNestedString: {
        almostThere: {
          success: "you made it!"
        }
      }
    },
    favoriteString: "nice!"
  }
};

gatherStrings(nestedObj) // ["Lester", "Testowitz", "you made it!", "nice!"];
```

Further Study

Binary Search

Given an array (not a linked list!) of sorted numbers and a value, return the index of that value. If not found, return -1. This algorithm should run in O(log(N)) time (where N is the number of elements in the array):

```
binarySearch([1,2,3,4],1) // 0
binarySearch([1,2,3,4],3) // 2
binarySearch([1,2,3,4],5) // -1
```

Additional Practice

If you'd like some additional practice before moving on to more challenging problems:

https://www.codewars.com/kata/the-real-size-of-a-multi-dimensional-array/train/javascript <https://www.codewars.com/kata/the-real-size-of-a-multi-dimensional-array/train/javascript>

https://www.codewars.com/kata/sum-squares-of-numbers-in-list-that-may-contain-more-lists/train/javascript <https://www.codewars.com/kata/sum-squares-of-numbers-in-list-that-may-contain-more-lists/train/javascript>

https://www.codewars.com/kata/recursive-replication <https://www.codewars.com/kata/recursive-replication>

Balanced Brackets

Re-write the Balanced Brackets challenge from Stacks and Queues to use recursion, rather than a stack.

Split Square

A four-part intermediate recursion challenge: Split Square <split-square/index.html>

Boggle

A tricky recursion challenge: Boggle <boggle/index.html>

Solution

View our Solution <solution/index.html>