JavaScript Lab 2

Upload in your GitHub account and send the link
Due Date: Friday, 16th

# Create a function for each question (Total 15) in one JavaScript file. You have to submit your js file with 15 functions by the end of the class.

1. Given an array of ints length 3, return an array with the elements "rotated left" so {1, 2, 3} yields {2, 3, 1}.

rotateLeft3([1, 2, 3]) → [2, 3, 1]
rotateLeft3([5, 11, 9]) → [11, 9, 5]
rotateLeft3([7, 0, 0]) → [0, 0, 7]

2. Given an array of ints length 3, return a new array with the elements in reverse order, so {1, 2, 3} becomes {3, 2, 1}.

reverse3([1, 2, 3]) → [3, 2, 1]
reverse3([5, 11, 9]) → [9, 11, 5]
reverse3([7, 0, 0]) → [0, 0, 7]

3. Given an int array length 2, return true if it contains a 2 or a 3.

has23([2, 5]) → true
has23([4, 3]) → true
has23([4, 5]) → false

4. Given an int array length 3, if there is a 2 in the array immediately followed by a 3, set the 3 element to 0. Return the changed array.

fix23([1, 2, 3]) → [1, 2, 0]
fix23([2, 3, 5]) → [2, 0, 5]
fix23([1, 2, 1]) → [1, 2, 1]

5. Given an array of ints of odd length, look at the first, last, and middle values in the array and return the largest. The array length will be a least 1.

maxTriple([1, 2, 3]) → 3
maxTriple([1, 5, 3]) → 5
maxTriple([5, 2, 3]) → 5

6. Given an array of ints, swap the first and last elements in the array. Return the modified array. The array length will be at least 1.

swapEnds([1, 2, 3, 4]) → [4, 2, 3, 1]
swapEnds([1, 2, 3]) → [3, 2, 1]
swapEnds([8, 6, 7, 9, 5]) → [5, 6, 7, 9, 8]

7. We'll say that a 1 immediately followed by a 3 in an array is an "unlucky" 1. Return true if the given array contains an unlucky 1 in the first 2 or last 2 positions in the array.

unlucky1([1, 3, 4, 5]) → true
unlucky1([2, 1, 3, 4, 5]) → true
unlucky1([1, 1, 1]) → false

8. Given 2 int arrays, a and b, of any length, return a new array with the first element of each array. If either array is length 0, ignore that array.

front11([1, 2, 3], [7, 9, 8]) → [1, 7]
front11([1], [2]) → [1, 2]
front11([1, 7], []) → [1]

9. We'll say that a value is "everywhere" in an array if for every pair of adjacent elements in the array, at least one of the pair is that value. Return true if the given value is everywhere in the array.

isEverywhere([1, 2, 1, 3], 1) → true
isEverywhere([1, 2, 1, 3], 2) → false
isEverywhere([1, 2, 1, 3, 4], 1) → false

10. For each multiple of 10 in the given array, change all the values following it to be that multiple of 10, until encountering another multiple of 10. So {2, 10, 3, 4, 20, 5} yields {2, 10, 10, 10, 20, 20}.

tenRun([2, 10, 3, 4, 20, 5]) → [2, 10, 10, 10, 20, 20]
tenRun([10, 1, 20, 2]) → [10, 10, 20, 20]
tenRun([10, 1, 9, 20]) → [10, 10, 10, 20]

11. Given an array of ints, return true if every 2 that appears in the array is next to another 2.

twoTwo([4, 2, 2, 3]) → true
twoTwo([2, 2, 4]) → true
twoTwo([2, 2, 4, 2]) → false

12. Return an array that is "left shifted" by one -- so {6, 2, 5, 3} returns {2, 5, 3, 6}. You may modify and return the given array, or return a new array.

shiftLeft([6, 2, 5, 3]) → [2, 5, 3, 6]
shiftLeft([1, 2]) → [2, 1]
shiftLeft([1]) → [1]

13. Return an array that contains the exact same numbers as the given array, but rearranged so that all the even numbers come before all the odd numbers. Other than that, the numbers can be in any order. You may modify and return the given array, or make a new array.

evenOdd([1, 0, 1, 0, 0, 1, 1]) → [0, 0, 0, 1, 1, 1, 1]
evenOdd([3, 3, 2]) → [2, 3, 3]
evenOdd([2, 2, 2]) → [2, 2, 2]

14. This is slightly more difficult version of the famous FizzBuzz problem which is sometimes given as a first problem for job interviews. (See also: FizzBuzz Code.) Consider the series of numbers beginning at **start** and running up to but not including **end**, so for example start=1 and end=5 gives the series 1, 2, 3, 4. Return a new String[] array containing the string form of these numbers, except for multiples of 3, use "Fizz" instead of the number, for multiples of 5 use "Buzz", and for multiples of both 3 and 5 use "FizzBuzz". In Java, String.valueOf(xxx) will make the String form of an int or other type. This version is a little more complicated than the usual version since you have to allocate and index into an array instead of just printing, and we vary the start/end instead of just always doing 1..100.

fizzBuzz(1, 6) → ["1", "2", "Fizz", "4", "Buzz"]
fizzBuzz(1, 8) → ["1", "2", "Fizz", "4", "Buzz", "Fizz", "7"]
fizzBuzz(1, 11) → ["1", "2", "Fizz", "4", "Buzz", "Fizz", "7", "8", "Fizz", "Buzz"]

15. Say that a "clump" in an array is a series of 2 or more adjacent elements of the same value. Return the number of clumps in the given array.

countClumps([1, 2, 2, 3, 4, 4]) → 2
countClumps([1, 1, 2, 1, 1]) → 2
countClumps([1, 1, 1, 1, 1]) → 1