Javascript Coding Exercises

# Coding Challenges

Use JavaScript and a JavaScript Playground tool (jsfiddle.net) to complete the following coding exercises. Paste your answers into a text file. You may use books, notes and the web to research your solution, however you should not copy solutions directly from any source.

In cases where you need to display values to the screen, you may use console.log(str) to display the values to the browser console.

## Captcha Test

Create a “captcha test” where a page will prompt the user with a simple question. If the user isn’t a browsing “bot” but a real user, it should be able to answer the question. For this purpose, do the following:

* Generate two random positive integer numbers between 0 and 100.
* Prompt the user to input the result of adding both numbers.
* Confirm that the user wants to enter that value (“Accept” moves forward, and “Cancel” asks for a value again.)
* If the result is correct, log to the console: “Human user identified.”
* If the result isn’t correct, ask the user to try again.

Hint: use "prompt(promptMessage)" to ask for user input - it will return the value input by the user. And use "confirm(confirmMessage)" to display a confirmation dialog - it will return true if the user accepts and false if the user cancels.

Note that both of these functions stop execution and wait till the user has entered a value or made a selection.

Hint: input values will be treated as Strings, while numbers that are the result of calculations will be treated as Numbers. Hence, if you compare them, to ignore their type, don’t use strict inequality or equality (use != or == instead of !== or ===).

**Example**:

Prompt: How much is 45 + 20?  
// User inputs 75  
Confirm: Are you sure that 45 + 20 = 75?  
// User clicks “Accept”  
Prompt: That’s not correct. Try again. How much is 45 + 20?  
// User inputs 65  
Confirm: Are you sure that 45 + 20 = 75?  
// User clicks “Accept”  
// “Human user identified” is logged to the console

## Find the Biggest Factor

Create a function ‘findBiggestFactor’ that returns the biggest positive factor (also called the biggest positive divisor) of a given positive integer number greater than zero. Note that the remainder of dividing the given integer by the returned factor has to be zero.

Also exclude the trivial factors from your result set – that is, when finding the biggest divisor of a given integer, don’t consider that same integer (or ‘1’) as a valid result.

If no divisors were found, return ‘NaN’ (Not a Number.)

**Example**:

findBiggestFactor(4); // returns ‘2’  
findBiggestFactor(125); // returns ‘25’  
findBiggestFactor(7); // returns ‘NaN’

## Validate Email Address

Create a function ‘isEmailValid that checks the given string with an email address, and returns true if all the following basic validation rules pass (and false otherwise:)

* The given email address contains at least an “@” symbol
* The given email address contains at least a “.”
* The “.” appears after the “@” symbol

We’ll test it with the following simple cases (so you can stick to the given basic rules, and don’t worry about more complex cases:)

**Example**:

isEmailValid("invalidEmail.com"); // false  
isEmailValid("invalidEmail@domain"); // false  
isEmailValid("invalid.Email@domain"); // false  
isEmailValid("validEmail@example.com"); // true

## Calculate Batting Average

Create a function ‘calculateBattingAverage’ that, given an array of Boolean values where “true” represents a strike, and “false” represents a successful batting attempt, returns a number with the batting average of the player.

**Example**:

var exampleStrikes =   
 [true, true, false, true, false, true, true, true, true, false, false, true, true, true, true];  
calculateBattingAverage(exampleStrikes); // returns 0.26666666666666666

## Finding the Year With Most Wins

Create a function ‘logYearWithMostWins that:

* Is given an Object as a parameter where the keys are the years, and the values are Objects with the stats for such year.
* Where every stats Object has the following properties: ‘wins’ (number of victories,) ‘loses’ (number of defeats) and ‘place’ (finishing place in the championship.)
* That logs to the console the year with most wins, the number of victories in that year, and the finishing place in the championship.

**Example**:

**var *cardinalStats*** = {  
 2015: {  
 **wins**: 100,  
 **loses**: 62,  
 **place**: 1  
 },  
 2014: {  
 **wins**: 90,  
 **loses**: 72,  
 **place**: 1  
 },  
 2013: {  
 **wins**: 97,  
 **loses**: 65,  
 **place**: 1  
 },  
 2012: {  
 **wins**: 88,  
 **loses**: 74,  
 **place**: 2  
 },  
 2011: {  
 **wins**: 90,  
 **loses**: 72,  
 **place**: 2  
 },  
 2010: {  
 **wins**: 88,  
 **loses**: 76,  
 **place**: 2  
 }  
};  
  
*logYearWithMostWins*(***cardinalStats***);

*// Logs "The year with most wins was 2015 with 100 victories, and the finishing place was 1"*

## Create a Baseball Player Object

Create a new object class BaseballPlayer to represent a baseball player, that contains the following:

* name: a String with the name of the player. This should be provided as part of the constructor when creating a new object.
* successfulBattingAttempts: integer Number that tracks the amount of successful hits. Should be initialized with the value of 0.
* strikes: integer Number that tracks the amount of strikes. Should be initialized with the value of 0.
* hit(): function that takes no parameters, and that has a 50% chance of increasing the number of successful hits of this player by one, and a 50% chance of increasing the number of strikes by one.
* getSuccessfulBattingAttempts(): function that takes no parameters and returns the number of successful hits from the player
* showSuccessfulBattingAttempts(): function that takes no parameters, internally invokes getSuccessfulBattingAttempts(), and logs to the console the name of the player, followed by a colon (:), followed the number of successful hits from the player, and ended in the word “hits.”

Then create a couple of object instances, execute four hits in each instance, and log the successful hits of each player.

**Example**:

**var *matt*** = **new** *BaseballPlayer*(**"Matt"**);  
***matt***.hit();  
***matt***.hit();  
***matt***.hit();  
***matt***.hit();  
  
**var *greg*** = **new** *BaseballPlayer*(**"Greg"**);  
***greg***.hit();  
***greg***.hit();  
***greg***.hit();  
***greg***.hit();  
  
***matt***.showSuccessfulBattingAttempts(); *// In my case, it logged: Matt: 3 hits****greg***.showSuccessfulBattingAttempts(); *// In my case, it logged: Greg: 2 hits*

## Create a Book Object

Create a new object class Book with the following methods:

* addPage(string) – adds a new page to the book (passed as a parameter string)
* countOccurrences(word) – returns how many times a word appears in the book (provided as a string.) These matches should be case-insensitive. A partial match should not count as a match (i.e.: “king” in “kingdom” would not count as a match.) Assume that your pages contain no punctuation symbols, and that everything separated by a space is a word.
  + Note: creating a “countOccurrencesInPage(pageContents, word)” function as a utility function can help you split the problem in simpler fragments, and make your code more readable.
* logAllText() – logs to the console the whole text of the book.

**Example**:

*// Creating an instance***var *lordOfTheRings*** = **new** *Book*();  
***lordOfTheRings***.addPage(**"Upon great pedestals founded in the deep waters stood two great kings of stone"**);  
***lordOfTheRings***.addPage(**"The left hand of each was raised palm outwards in gesture of warning"**);  
***lordOfTheRings***.addPage(**"Great power and majesty they still wore the silent wardens of a long-vanished kingdom"**);  
  
***lordOfTheRings***.logAllText();  
**console**.log(**"The word 'king' appears "** + ***lordOfTheRings***.countOccurrences(**"king"**) + **" times"**);  
*// Logs: "The word 'king' appears 0 times"***console**.log(**"The word 'great' appears "** + ***lordOfTheRings***.countOccurrences(**"great"**) + **" times"**);  
*// Logs: "The word 'great' appears 3 times"*

## Word Count

Create a function ‘wordCount’ that takes a string parameter ‘str’. The function should return the count of words in the ‘str’ parameter string. A word is considered any series of characters (letter, number, punctuation) separated by one or more spaces.

**Example**:

wordCount(‘This is a short sentence!’); // returns 5  
wordCount(‘ThisIsA!$ReallyLongWord’); // returns 1  
wordCount(‘ ‘); // returns 0

## String Capitalization

Create a function ‘capitalize’ that takes a parameter ‘str’. The function should return a string with the first letter in each word capitalized.

**Example**:

capitalize(“i love to code!”); // returns “I Love To Code!”

## Fizz Buzz Exercise

Create a JavaScript function ‘fizzBuzz’ that takes 1 parameter, ‘n’. The function should return a string that contains the following for each number between 1 and n.

* If n <= 0
  + Return an empty string
* If n > 0
  + Print the number
  + If the number is divisible by 3 write ‘fizz’
  + If the number is divisible by 5 write ‘buzz’
  + If the number is divisible by 3 and 5 write ‘fizzbuzz’

**Example**:

fizzBuzz(0); // returns ‘’  
fizzBuzz(15); // returns ‘1, 2, 3fizz, 4, 5buzz, 6fizz, 7, 8, 9fizz, 10buzz, 11, 12fizz, 13, 14, 15fizzbuzz’