Lab #2

Due: 10/10/22 @ 11:59pm

Content Covered
Selection Statements, Loops, and Arrays in JavaScript

Getting Started

1. Sign into replit.com
2. Create a new JavaScript REPL, name it whatever you like

Editing Your Code

In this lab you will be defining functions to solve 6 problems as described below. All of your function definitions should be written in index.js. It is important that you name your functions exactly as specified by each problem so that the autograder will be able to call them.

Any variables and parameters can be named however you like. You can write any additional functions to help you solve the problems as you see fit. Each problem is more complex than the ones before it.

Submission

The Autolab submission for this lab will open up no later than 10/3/22 @ 5:00pm.

Your final submission for this lab should be a .zip file download from replit.com and submitted to Autolab. You may submit as many times as you like, but only your last submission will count for your final grade.

Submissions close at 11:59pm on 10/10/22 and no late submissions will be accepted.

Problems

Problem 1

Define a JavaScript function named `sum` which takes an array of numbers as input and returns their sum. For example, `sum([3,6,5,2])` must return `16`, since 3+6+5+2 = 16. When called with an empty array as the argument, the function must return `0`.

Problem 2

Define a JavaScript function named `product` which takes an array of numbers as input and returns their product. For example, `product([3,6,5,2])` must return `180`, since 3*6*5*2 = 180. When called with an empty array as the argument the function must return `1`.

Problem 3

Define a JavaScript function named `smallest` which takes an array of numbers as input and returns the smallest. For example, `smallest([3,6,5,2])` must return `2`. When called with an empty array as argument the function must return the value `undefined`.

Problem 4

Define a JavaScript function named `shortest` which takes an array of strings as input and returns the shortest. If there is not a unique shortest string the function must return the one at the lowest index in the array. For example, `shortest(["aardvark","cat","lizard","dog"])` must return `"cat"`. When called with an empty array as argument the function must return the value `undefined`.

Problem 5

Define a JavaScript function named average which takes an array of numbers as input and returns their average. For example, `average([3,6,5,2])` must return `4`. When called with an empty array as argument the function must return the value `undefined`.

Problem 6

Define a JavaScript function named `names` which takes two arrays of strings representing first and last names respectively, and returns a single array with the full names in the format "last, first".

For example, `names(["Jon", "Peter", "Grace"],["Snow", "Pan", "Hopper"])` must return `["Snow, Jon", "Pan, Peter", "Hopper, Grace"]`. You can assume that the two argument arrays will be the same length. When called with empty arrays, the function must return an empty array.

Useful Suggestions and Advice

- **Think before you code** - think about the input to the function, and the desired output, and plan out how you would solve the problem in your head or on paper before you start to code. Having an idea of where you are going before you start typing can make the coding process smoother.
- **Write your own tests** - each problem describes an example test case or two that you can use to check if your implementation is correct. But they are not sufficient to guarantee you have a completely correct implementation. It can help to think through and write additional test cases (even before you code) to make sure your functions are behaving as you expect them to. This will also allow you to more quickly test your code as you go, rather than having to wait for Autolab to be open, and to go through the entire submission process just to get feedback.
- **Break down bigger problems into smaller ones** - if you are stuck on a problem, try solving a smaller piece of the problem first. You can even write it as a separate function, and test it out. Then once you have the small pieces working, building the solution to the bigger problem from your smaller solutions should be easier.
- **Ask for help** - during lab sessions, in office hours, and on Piazza. If you do choose to ask questions on Piazza **do not post your own code in public posts.** If you are going to make your question public, make sure it is general/conceptual in nature. Otherwise, make your question private to just the instructors.