Variables & Data Types

Challenge 1

Step 1:

Create a single line comment that says *"This is a comment that won't be executed like regular Javascript code".*

Step 2:

Create a multiline comment that says *"This is a multiline comment that also won't be executed like regular Javascript code!!!"*

Challenge 2

Let's do a little variable work first.

Step 1

Start off by creating a `const` variable called `eyeColor` and set it equal to the string of your eye color.

Next, create a `let` variable called `hairColor` and set it equal to the string of your hair color.

Lastly, log the `eyeColor` and `hairColor` variables to the console.

Step 2

Try to reassign the `hairColor` variable to a different color and then log it to the console.

Next, try to reassign the `eyeColor` to a different color and log it to the console.

Oh no! Our first error 😰. It's all good, errors are our friends 🤝. If you take a look in at the error code you should get a sense of what our mistake is.

This should reinforce what we already know about variables declarations `const` variables like `eyeColor` can not be reassigned like `let` variables can. Comment out the last line of code to get rid of our error 🙂.

Challenge 3

Step 1:

There are multiple ways to create strings in Javascript. Try to use a different one for each variable in this step.

Create a variable `fName` and set it equal to your first name.

Create a variable `lName` and set it equal to your last name.

Create a variable `favShow` and set it equal to your favorite TV show.

Log the previous 3 variables to the console to check your work.

Step 2:

Declare a variable `greeting`, but don't set it equal to a value.

On another line of code, set your `greeting` variable to a greeting. Use the "+" to concatenate your string `fName` and `lName` variables from step 1 together with some additional text to form a greeting (e.g. *"Hi, Phillip Troutman!"*)

If you don't know how to use the "+" symbol to concatenate strings, try looking here in the MDN documentation here 🙂.

Step 3 (Bonus Step)

Lastly, declare a variable `fancyGreeting`.

On another line of code, set your `fancyGreeting` variable to another greeting (e.g *"Good evening to you, Phillip Troutman!"*). This time though, see if you can use backticks and something called template literals to more easily concatenate your `fName` and `lName` variables from Step 1.

If you are unfamiliar with template literals check out the MDN documentation again here 😎.

Log your fancy greeting to the console to check your work.

Challenge 4

Step 1

Create a variable `favNumber` and set it equal to your favorite number.

Create a variable `randomNumber` and set it equal to a random number.

Step 2

Create a variable `sum` and set it equal to the sum of `favNumber` and `randomNumber`.

Create a variable `product` and set it equal to the product of `favNumber` and `randomNumber`.

Create a variable `difference` and set it equal to the difference of `favNumber` and `randomNumber`.

Log the previous 3 variables to the console to check your work.

Step 3 (Bonus Step)

Create a variable `remainder` and set it equal to whatever the remainder value is from dividing your `favNumber` variable by your `randomNumber` variable.

If you are unfamiliar with how to do this in Javascript try looking in the MDN documentation here 😊.

Challenge 5

Step 1

Declare a `favSnacks` variable and set it equal to an array of your favorite snacks (3 or 4 snacks should do 😁)

Log the `favSnacks` array to the console.

Log the number of items in `favSnacks` to the console. E.g. if there are 5 snacks in the array, the number 5 should log to the console.

Finally, Log just the 2nd snack in the array to the console.

Step 2

Declare a `favAnimals` variable and set it equal to an array of your favorite animals.

Log the the `favAnimals` array to the console.

Log the number of items in `favAnimals` to the console.

Log the 1st animal in the array to the console.

Step 3 (Bonus Step)

This is a bit of a tough one 😅.

Write a bit of code that, no matter the length of the `favAnimals` array, will always log the last item in that array to your console.

For example, if the array was `["item 1", "item2" , "item3"]` it will log to your console "item 3", but if you added a "item 4" and re-ran your code it will log "item 4".

Here is a hint ... you will need to use the ".length" property 😸.

Challenge 6

Step 1

Create a variable `person` and set it equal to an object.

That object should have some defined properties on it. It should have a `fName` key with a value of your first name.

Log the entire `person` object to the console.

Step 2

Manually add another property to the `person` object using dot notation. This property should have a key of `favColor` and a value of your favorite color.

Log just the `favColor` value to the console.

Step 3

Manually add another property to the `person` object using bracket notation. This property should have a key of `likes to code` and a boolean value of `false`.

Log just the `likes to code` value to the console.

Update the `likes to code` key's value from the boolean value of `false` to the boolean value of `true`.

Step 4 (Bonus step)

Manually add another property to the `person` object using dot notation. This property should have a key of `siblings` and a value of another object (this is referred to as a "nested object").

The siblings object should have a `brothers` and a `sisters` key with values that represent the number of brothers and sisters you have.

Log the value of the `brothers` to the console.

Log the value of the `sisters` to the console.

Challenge 7(Bonus Challenge 😮)

Given the mental model of Javascript that we have learned so far, practice diagramming out all of the code in Challenge 6. Try sharing your screen and using Excalidraw for your diagramming.

Alternatively you can use an iPad (or other tablet), or good old fashioned pen and paper to diagram the code. You should still keep the proper pair programing dynamic here. One person should be navigating (explaining code) while the other is driving (diagraming the code out).