Mark Gutierrez 02/11/2020

Homework part 2

1. What is the JavaScript code doing?

```
//Add

const addNums = () => {

   let num1 = document.getElementById("num1").value;

   let num2 = document.getElementById("num2").value;

   let sum = parseInt(num1) + parseInt(num2);

   document.getElementById("add_result").innerHTML = sum;

};
```

The JavaScript code is using an anonymous arrow function to get the sum of "num1" and "num2." First, we declare const and assign the anonymous arrow function. Inside the anonymous arrow function, we declare the variable "let num1" and assign the "document.getElementById("num1").value." This will get the element in your HTML with the Id "num1." Because we are using the "input" element we need to get the "value" that the user types and do that we use the method ".value." We are adding 2 values so we do the same for the second input "num2." We make a third variable let called "sum" which will get both numbers and add them using the "+" arithmetic operator. Before we can add the 2 values, we need to convert the input from a primitive data type string to a primitive data type number. To do that we use the function "parseInt()" and add our "let" variable "num1" inside it. To return the user the answer in the Browser, we must get the Id with "document.getElementById("add_result")" and using the property "innerHTML" we can change the HTML content without reloading the browser. We want the HTML content to change and reflect the sum of the 2 values so we set the "innerHTML = sum." Lastly, in order for the user to click the button and call the anonymous arrow function we must add the "onclick" attribute inside the button element.

2. Var, let, const, and variable naming

   a. I_Love_$
   b. $_sponsored
   a. _$100_Salary

3. Let, const, and var

a. Let:

```
let d;
undefined
d = 10;
10
d = 20;
20
```

You can re-assign let

b. Const:

```
const earthGravity = 9.8;
earthGravity = 200;
console.log(earthGravity);
main.js:5 Uncaught TypeError: Assignment to constant variable.
    at main.js:5
```

Const cannot be re-assigned.

c. Var:

```
var b = 10;
b = 20;
console.log(b);
20
```

Var can be re-assigned.

4. Global scope vs local scope

Global scopes are available everywhere in the javascript code including functions. When a variable is declared outside a function, they are global. Any variable that is undeclared is considered global. Variables are executed first in the code and are hoisted or moved at the top, so is generally good to code them at the top. Because of hoisting variables can be used before being declared. However, it is not going to affect the initialization value. Var does not have a block scope but rather a function scope.

Local scopes are only available in a function and can only be accessed within the function. Const and let have block scope, unlike var.