Exercise: JavaScript Scope and Closure

Questions:

1. Research JavaScript variable scope and closure.

Scope is how a computer keeps track of all the variables in a program. It refers to the specific environment where a variable is accessible and can be used. JavaScript has two types of scope i.e., global, and local scope.

A global scope variable is one that is declared outside a function or curly brackets. It is accessible anywhere in the program.

A local scope variable is one that is only available in a section of your code known as the local scope, e.g., variables declared within functions are only accessible in those functions or those declared within curly braces.

Closures are defined when functions are defined within other functions, that is, when the inner function has access to the variable scope of the outer function. The nesting of functions also results in a nesting of scope. The outer scope is said to “enclose” the scope of the inner function, hence the term closure.

# References

Vincent, W. (2017, September 12). *JavaScript Scope and Closures - William Vincent*. Retrieved from William Vincent: https://wsvincent.com/javascript-scope-closures/

1. Define variable scope as it relates to JavaScript in your own words. Give a code example of two different types of variable scope in JavaScript.

A scope variable is a placeholder or a place to store a value, it can either be global or local.

const ***welcome*** = "Welcome"; // global scope  
  
const greeting = () => {  
 let morning = "Good Morning"; // local scope  
 ***console***.log(morning);  
}  
  
***console***.log(***welcome***);  
greeting();

1. Consider the following JavaScript code:
   1. Is the access to ‘myValue’ on line 7 allowed without error? Why or why not?

Yes, the access to ‘myValue’ is allowed because the myFunction gains access to myObject function as it is being passed in as a parameter, this allows myFunction to accesss my value.

* 1. What does the code print to the console on line 12? Why?

The code will print an uncaught ReferenceError: input is not defined at scope.js:22. This is because input is a parameter that is defined in the myFunction function and is a local variable to that function. Hence it cannot be accessed outside the function.

1. Does the code succeed in preventing access to ‘myValue’ on line 8? Why or why not?

Yes, because getMyObject’s return value is being passed in as a parameter to the myFunction hence the input.myValue is accessible.

1. Define closure as it relates to JavaScript in your own words. Give one advantage **AND** one disadvantage of using closure in JavaScript.

A closure gives you access to outer function scopes from an inner function, that is, a function within a function.

1. What is printed to the console when the following code is executed? Explain how the code works?

1

2

3

The code starts at zero then enters the first function and initializes the count, then it enters the second function and adds one to the initialized and stored variable count until the loop id finished.

1. What is printed to the console when the following code is executed? Explain how the code works? Why does the value of ‘myValue’ differ from ‘getMyValue’ when printed to the console?

It gets in and tries to access the value and it gets the value which is undefined because no actual value of myValue has been set. Then after setting the value it gets into gets into the get myValue Function but cannot access anything because the value must be accessed through the method get my value only, as my value is enclosed within the get my value function.