Luis Mejia
CIS-14A-48590
Written Assignment 11

Please include both questions and answers with your submission.

Submit in word or PDF format.

1. **Define Closure. Show an example, in JavaScript, of a program with a closure from chapter 11 that is not the lab. Explain how the program works.**

   A closure is a function together with a referencing environment. With closure, a variable defined outside a function can be used inside that same function. For example, because of closure, the code below will print "*Just an every day LOCAL*". Where you define things is what matters so although *whereAreYou* is invoked in the same scope as the global *justAVar* variable, the value of the local *justAVar* variable will actually be the one to be stored in the results variable. In this case, *whereAreYou()* returns inner and when *inner* is invoked with *innerFunction()*, *inner()* uses the variables in its environment being "*var justAVar* = *Just an every day LOCAL*". Closures are basically functions with preserved data and a form of encapsulation.

```
var justAVar = "Oh, don't you worry about it, I'm GLOBAL";

function whereAreYou() {

 var justAVar = "Just an every day LOCAL";

 function inner() {

   return justAVar;

 }

 return inner;

}

var innerFunction = whereAreYou();

var result = innerFunction();

console.log(result); // Prints "Just an every day LOCAL"
```

Luis Mejia
CIS-14A-48590
Written Assignment 11

2. **Implement the following function that takes a password argument and returns a function that accepts a password guess and returns true if the guess matches the password, with a closure.**

   The implemented code for a function that takes a password argument and returns a function that accepts a password guess and returns true if the guess matches the password is:

   function makePassword (_____) {

   return _____ {

   return (_____);

   };

   }

```
function makePassword(password) {
 return function guess(passwordGuess) {
   return (passwordGuess === password);
 };
}
```

Luis Mejia
CIS-14A-48590
Written Assignment 11

3. **Implement a program in JavaScript that bakes a cake and sets a timer for 10 minutes.**

   The following is a program that bakes a cake and sets a timer for10 minutes.

```
// Function declaration

function bakeCake() {

 // Notify baking has started

 console.log("Baking!");

 // Set timer to bake for 10 minutes

 setTimeout(function() {

             console.log("Cake is finished!")

            },

            600000);

}

// Bake cake

bakeCake();
```

4. **In an 80s movie, computer animated characters were forced to fight in a computerized blood sport. The ones that died were referred to as derezzed. Name the movie.**

   The 80s movie where computer animated characters were forced to fight in a computerized blood sport and where the ones that died were referred to as derezzed is called Tron.

Luis Mejia
CIS-14A-48590
Written Assignment 11

5. **How do closures affect scope? Explain.**

   Closure is very powerful in that it affects scope. For example, if a function makes use of a variable that is not defined within its scope, it will look as far up and outside its scope as needed to find it.

6. **What is an anonymous function? Write a line of JavaScript code that exemplifies an anonymous function.**

   An anonymous function is a function that does not use a name or simply a function expression without a name. By using anonymous functions, we can make our code more concise, more efficient, and more maintainable. We can use anonymous functions where we would expect a function reference. Below is code that exemplifies an anonymous function.

```
window.onload = function () {

 alert("The window has finished loading!");

}
```

7. **What is a nested function? Show me an example of a nested function from Chapter 11.**

   A nested function is a function nested within another function. In other terms, it is a function within a function. Below is an example of a nested function from chapter 11 where the counter() function is nested inside of the makeCounter() function.

```
function makeCounter() {

 var count = 0;

 function counter() {

   count = count + 1;

 }

 return count;

}
```

Luis Mejia
CIS-14A-48590
Written Assignment 11

8. **Can JavaScript determine the scope of a variable by reading the structure of code? If so, what is that called?**

   Yes, JavaScript can determine the scope of a variable by reading the structure of the code. This is called *Lexical Scope*. The advantage of lexical scope is that we can always look at the code to determine the scope. With lexical scope, where you define things is what matters.

9. **What is a free variable and why would you use it when coding?**

   A *free variable* is one that is not defined locally in a function. You would use free variables when coding and working with closures. When a function that has free variables is combined with an environment variable that provides variable bindings for all those free variables, a closure results.

10. **When is a function declaration defined? When is a function expression evaluated?**

    When a browser begins to work with code, it will do two passes. On the first pass, the browser will look for any function declarations and will create/define any and all functions that are declared. Afterwards, the browser will begin to evaluate and execute code top down. It is during the second run that the browser will evaluate function expressions along with the rest of the code in the program.