Week 3

- **Explain the difference between the == operator and the === operator.**

The (= = =) operator in Javascript behaves exactly like the (= =) operator in Python except for the fact that there is no conversion done. For the (= = =) the types must be the same to be considered equal.
The (= =) operator will compare the things around the operator **after** performing the necessary type conversions, in contrary to the (= = =) operator. So for two values that do not have the same type, but do have the same value, the (= =) operator will return True.

- **Explain what a closure is. (Note that JavaScript programs use closures very often)**

JavaScript variables can belong to the local or global scope. Global variables can be made local (private) with closures. In the first example **a** is a local variable, meaning it can only be used inside the function where it is defined, being unable to access from other function or other code. In the second example the variable **a** is a global variable, can be used (and therefore also be changed) by all scripts in the page.
Global and local variables can have the same name, but are still different variables. Meaning that if you modify one, it does not change the other.

For example: A function can access the variables that are defined *inside* the function as follows:

```
function myFunction() {
        var a = 4;
        return a*a;
}
```

Also *outside* a function, which is done as follows:

```
var a = 4;
function myFunction() {
        return a*a;
}
```

- **Explain what higher order functions are.**

Functions that operate on other functions, either by taking them as arguments or by returning them, are called higher-order functions. Since we have already seen that functions are regular values, there is nothing particularly remarkable about the fact that such functions exist.

- **Explain what a query selector is and give an example line of JavaScript that uses a query selector.**

The querySelector() method returns the first element that matches a specified CSS selector(s) in the document. Note: The querySelector() method only returns the first element that matches the specified selectors. To return all the matches, use the querySelectorAll() method instead. If the selector matches an ID in document that is used several times (Note that an "id" should be unique within a page and should not be used more than once), it returns the first matching element.

This example returns the first <p> element in the document with the class=”example”:

```
Document.querySelector(“p.example”);
```

This example changes the text of an element with id=”demo”:

```
document.querySelector(“#demo”).innerHTML = “Hello World!”;
```