Zusammenfassung EIA2 Workshop 22.03.2022

## parseInt/parseFloat:

String wird interpretiert (geparsed) und die erste Zahl der folgenden Zahlenarten zurückgegeben:Int 🡪 Ganzzahl (Integer)  
Float 🡪 Fließkommazahl (Floating Point)

Beispiel:  
„123,45“ + 1 🡪 „123,451“

* 1 wird dem String „123,45“ hinzugefügt

**parseInt(**„123,45“**)** + 1 🡪 123 + 1 🡪 124

* Da parseInt nur ganze Zahlen zurückgibt wird die Zeichenfolge „123“ bis zum Komma als Integer interpretiert, auf welche 1 dazu addiert werden kann.

**parseFloat(**„123,45“**)** + 1 🡪 123,45 +1 🡪 124,45

* Die Zeichenkette 123,45 wird als Float interpretiert, auf welche 1 dazu addiert werden kann.

## return:

* Beendet eine Funktion
* Kann einen Wert zurückgeben

Beispiel:

function doSomething(\_x: String, …): boolean { Datentyp des Rückgabewerts  
 let result: boolean = true; Deklaration | Definition  
 …  
 **return result;**  
}

*In der Konsole wird jetzt true ausgegeben, da hier einfach der Rückgabewert von doSomething ausgegeben wird.*  
console.log(doSomething(“Hello”));

*Für Beispielzwecke ist doSomething in diesem Fall false. Man kann Rückgabewerte von Funktionen auch Variablen desselben Datentyps zuweisen.*  
let greeted: boolean = doSomething(“Piss off!”);

## Werte vs Referenzen | Komplexe Datentypen

*Die folgenden Zeilen Code sind in JavaScript Syntax geschrieben*  
**Beispiel:**  
let x = 1;  
let y = x;  
x += 1;

* **x = 2; y = 1;**

Bei der Zuweisung primitiver Datentypen (String, boolean, Integer) wird der Variable zugewiesen. Bei let y = x; wird y lediglich eine Kopie von x zugewiesen. In diesem Augenblick haben x und y den gleichen Wert (Cara’sches Bild: sie tragen die gleiche Hose).

**Beispiel:**  
let x = [1, 123, 17];  
let y = x;  
x[1] += 1;

* **x = [1, 124, 17]; y = [1, 124, 17];**

Bei der Zuweisung komplexer Datentypen (Bsp. Array), wird auf den Datentyp referiert. x und y beziehen sich hier auf dasselbe Array (Cara’sches Bild: sie tragen dieselbe Hose).