PYTHON TO JAVASCRIPT!!! - PART 2

Instruction

- You need to complete the **XXXXX** part with the JAVASCRIPT equivalent code
- You can work in team or by yourself –
  - Search on internet
  - or read the **1-Javascript Cheat Sheet.pdf**
  - https://www.w3schools.com/js/default.asp

- **IMPORTANT** : you need to test the code before writing it !!!

	PYTHON	JAVASCRIPT

BOOLEAN OPERATORS	**IS EQUAL, IS GREATER**
	`x = 5`
	`y = 5`
	`print (x == y)`
	`>True`
	
	AND / OR / NOT
	`x = 5`
	`y = 5`
	`print (not (x == y and ( x>5 or y<10) ))`
	`>false`
		IS EQUAL, IS GREATER
		`let x=5`
		`let y=5`
		`console.log(x==y)`
		
		AND / OR / NOT
		`let x = 5`
		`let y = 5`
		`console.log(!(x==y && (x>5 || y<10)))`

TYPES	CONVERT A STRING TO INTEGER
	int(<STRING>)
	
	`n = '5'`
	`print (int(n) + int(n))`
	`>10`
	
	CONVERT A INTEGER TO STRING
	str(<INTEGER>)
	
	`n = 5`
	`print (str(n) + str(n))`
	`>55`
		CONVERT A STRING TO INTEGER
		
		`let string = "5"`
		`let num = parseInt(string)+parseInt(string);`
		`console.log(num)`
		
		CONVERT A INTEGER TO STRING
		`let num = 5;`
		`let sumNumToString = num.toString() +`
		`num.toString();`
		`console.log(sumNumToString)`

FUNCTION	DEFINE A FUNCTION
	`def sum(n1, n2):`
	`  total = n1 + n2`
	`  return total`
	
	`print(sum(100,200)) -> 300`
		DEFINE A FUNCTION
		`function sum(n1,n2){`
		`  return n1 + n2;`
		`}`
		`console.log(sum(100,200));`
		
		DEFINE AN ARRAY FUNCTION

DATA STRUCTURES

ARRAY

```
# Create empty array
array = []
fruits = [“apple”, “banana”]


# Create array with values
array = [12, 13, 15, 16]

# Access using index
value = array[2]

# Insert value at index
array.insert(1, 20)

# Insert value  at the end
array.append(20)

# Remove using index
array.pop(2)

# Get a sub array
subarray = array[2:25]
```

ARRAY 2D

```
# Create array2D with values
array2D = [ [12, 13, 15, 16], [4, 5, 6, 7]]

# Access using index
value = array2D[2][0]
```

DICTIONARY

```
# Create empty dictionary
dic = {}

# Create array with values
dic = { key1:value1, key2:value2 … }

# Access using key
value = dic[key1]

# Add value for a new key
dic[key3] = value3

# Update  value from existing key
dic[key2] = value2New


# Remove using key
```

ARRAY

```
# Create empty array
let firstArr = [];
let secondArr = ["Nathan", "Jack"];

# Create array with values
Let array= [12, 13, 15, 16]
# Access using index
let fruits = ["Banana", "Orange", "Apple", "Mango"];
let fruit = fruits[0];
console.log(fruit)

# Insert value at index
let fruits = ["Banana", "Orange", "Apple","Mango"];
fruits[1] = "Lemon";
console.log(fruits)

# Insert value  at the end
const fruits = ["Banana", "Orange", "Apple"];
let insert=fruits.push("Lemon");
console.log(fruits)

# Remove using index
let numbers = [1,2,3,4]
let index_num = numbers.splice(2,1);
console.log(numbers)
# Get a sub array
console.log(object.sort());
```

ARRAY 2D

```
# Create array2D with values
let items = [
   [1, 2],
   [3, 4],
   [5, 6]
 ];
# Access using index
console.log(items[0][1]);
```

OBJECT

```
# Create empty object
Let object= {}
# Create array with values
let person = {
  firstName : "John",
  lastName  : "Doe",
  age     : 50,
};


# Access using key
For (let key in object){
```

	dic. pop(key2)	consol.log(key)
		}
		
		# Add value for a new key
		object.key[3]=”value3” or
		object[“key3”]=”value3”
		# Update value from existing key
		const newObj = Object.assign({});
		
		# Remove using key
		delet object[“keu”] or
		delet object.key

Q2 **The 3 ways to declare a variable in JS**

```
var a = 4
Let a = 4
const a = 4
```

⇨ Can you explain what the differences?

Var: Always declare JavaScript variables with var , let , or const . The var keyword is used in all JavaScript code from 1995 to 2015. The let and const keywords were added to JavaScript in 2015.

Let: The `let` keyword was introduced in ES6 (2015).

Variables defined with `let` cannot be Redeclared.

Variables defined with `let` must be Declared before use.

Variables defined with `let` have Block Scope.

Const: The `const` keyword was introduced in ES6 (2015).

Variables defined with `const` cannot be Redeclared.

Variables defined with `const` cannot be Reassigned.

Variables defined with `const` have Block Scope.