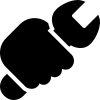
**W3** PRACTICE

*JS – ES6 Arrays + NPM*

## At the end of this practice, you should be able to…

* **Create**, **update** and **remove** items in array
* Use the **arrow** **syntax** to define functions as parameters: f = () => {}
* Use ES6 arrays methods such as: **find, map, filter, foreach** for effective array operations

## How to work?



**BEFORE THE PRACTICE**

* First watch and understand the **following pages and videos**:

[basic operations](https://www.w3schools.com/js/js_array_methods.asp), [map](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map), [find](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/find), [filter](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter), [foreach](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach)

[video 1](https://www.youtube.com/watch?v=yQ1fz8LY354), [video 2](https://www.youtube.com/watch?v=R8rmfD9Y5-c)

* Then complete the **quiz** (*you can re-do it until you have 100% score*)

**DURING THE PRACTICE**

* To start the practice. **download the start code** from Google classroom

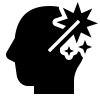
## How to submit?

* **Push your final code** on this GitHub repository (if you are lost, [follow this tutorial](https://www.datacamp.com/tutorial/git-push-pull))
* Finally, submit on **LMS** your GitHub repository URL

# About Node JS



## What is Node.js?



Node.js is a JavaScript runtime environment that can run on Windows, Linux, Unix, macOS, and more.

Node.js is able to **execute JavaScript** code **outside** **a web browser.**

We need to be able to run our JS and our ReactJS code using Node.js during this course.

## How to run JavaScript with Node.js?

Check node is installed

node –v

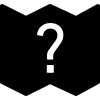
If not, you need to re-install Node or update it.

Create a new JS file, index.js with:

console.log(“Hello !”)

Open the VS code terminal and run

node ./index.js

 Are you lost?

*You can read the following documentation:*

<https://nodejs.org/en/learn/getting-started/differences-between-nodejs-and-the-browser>

<https://nodejs.org/en/download>

# UNDERSTAND THE CONCEPTS…

Before starting the exercise, complete this table with some code examples, to show you understood the theory.



Let's say we have the following start code:

Let numbers = [1, 2, 3, 4, 5]

|  |  |
| --- | --- |
| **Usage** | **Example of code** |
| **Add** an element at the end of the array | *<your code>* |
| **Loop** on all array elements | *<your code>* |
| **Access** to the array element with its index | *<your code>* |
| **Remove** an array element with its index | *<your code>* |
| **Filter** array elements | *<your code>* |
| **Transform each array element** by applying a function on them | *<your code>* |

# EXERCISE 1

Your task is to add the missing logic to a transformToObjects() function that should transform a list of numbers into a list of objects.

In the newly returned array, every object must have a val key and the input array's number as a value.

/\*\*

 \* Creates transform a list of numbers into a list of objects.

 \* @param {array} listOfNumbers - a list of numbers

 \* @returns a  list of objects

 \*/

function transformToObjects(listOfNumbers) {

  result = [];

  // Write your code here

  return result;

}

*Examples of inputs/outputs:*

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| [1, 2, 3] | [{val: 1}, {val: 2}, {val: 3}] |
| [44] | [{val: 44}] |

# EXERCISE 2

We are managing a data structure of students - representing a student with first Name and age properties.

const STUDENTS\_DATA = [

  { firstName: "An", age: 20 },

  { firstName: "Bình", age: 22 },

  { firstName: "Cẩm", age: 21 },

  { firstName: "An", age: 19 }, // Duplicate first name !

];

The updateStudentAge function is supposed to update the age of a student his/her first name

However, some students **have the same first name**!



Your task is to **update the data structure and the function** to manage the last name and the batch, and fix our problem!

# EXERCISE 3

# 

In order to manage an online store, we have 2 data structures:

* **A list of products in the shop**: *each product having a unique id, name and unit price*

const PRODUCTS = [

  { id: 1, name: "Apple", price: 2.5 },

  { id: 2, name: "Banana", price: 1.5 },

  { id: 3, name: "Orange", price: 3 },

  { id: 4, name: "Rice", price: 1.5 },

  { id: 5, name: "Chocolate", price: 3 },

];

* **A shopping cart**: which contain the **items the customer wants** to buy and their **quantity**

const SHOPPING\_CART = [

  { id: 1, quantity: 2 },

  { id: 3, quantity: 1 },

];

**Q1** - Complete the getCartTotalAmount() function to get the total amount of the current shopping cart.

*Example:*

* The cart contains 2 apples and 1 orange:

const SHOPPING\_CART = [

  { id: 1, quantity: 2 },

  { id: 3, quantity: 1 },

];

* Each apple costs 2.5 $
* Each orange costs 3 $
* The function return value shall be: 8 $

**Q2** - Complete the addProductToCart() function to add a product to the shopping cartcart.

- If the product **id already exists** in the cart, just **increment** its quantity:

addProductToCart(1)

[{ id: 1, quantity: 2 }] ---> [{ id: 1, quantity: **3** }]

- If the product id **does NOT exist** in the cart, **add a new item**, with a quantity 1

addProductToCart(2)

[{ id: 1, quantity: 2 }] ---> [{ id: 1, quantity: 2 },**{ id: 2, quantity: 1 }]**

**Q3** - Complete the removeProductFromCart() function to remove a product from the shopping cartcart.

* If the product id already exists in the cart, and quantity if >=2 : just decrement its quantity

removeProductToCart(1)

[{ id: 1, quantity: 2 }] ---> [{ id: 1, quantity: **3** }]

- if the product id already exists in the cart, and quantity is 1 : remove the item from the card

removeProductToCart(1) :

[{ id: 1, quantity: 1 }] ------> []

- if the product id does not exist in the cart, do nothing !