Alex Samos Muñoz

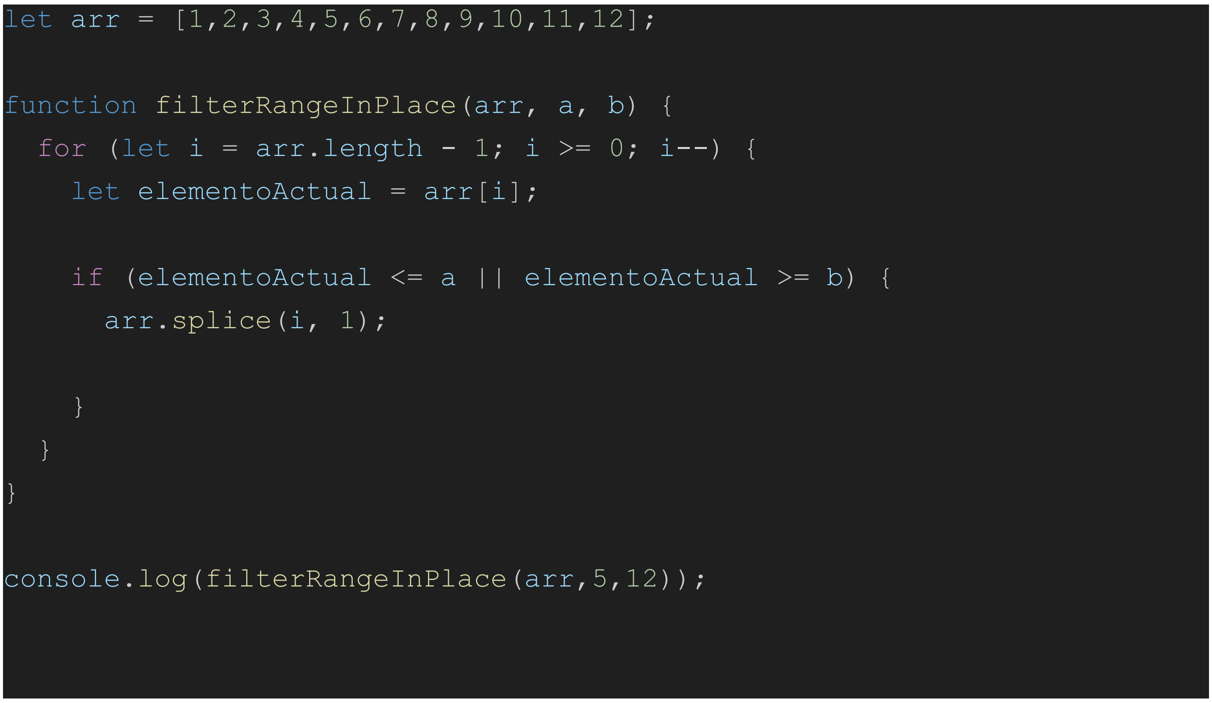
1. Write the function camelize(str) that changes dash-separated words like “my-shortstring” into camel-cased “myShortString”. That is: removes all dashes, each word after dash becomes uppercased.
P.S. Hint: use split to split the string into an array, transform it and join back.

```
let cadena = "hola-adios-bye";
function camelize(cadena) {
  let arr = cadena.split("-");
  let palabraNueva = "";
  for (let i = 0; i < arr.length; i++) {
    if (i > 0) {
       palabraNueva += arr[i].slice(0,1).toUpperCase() +
arr[i].slice(1);
    }else{
      palabraNueva += arr[i]
    }

  }
  return palabraNueva;
}

console.log(camelize(cadena));
```

2. Write a function filterRange(arr, a, b) that gets an array arr, looks for elements with values higher or equal to a and lower or equal to b and return a result as an array. The function should not modify the array. It should return the new array.

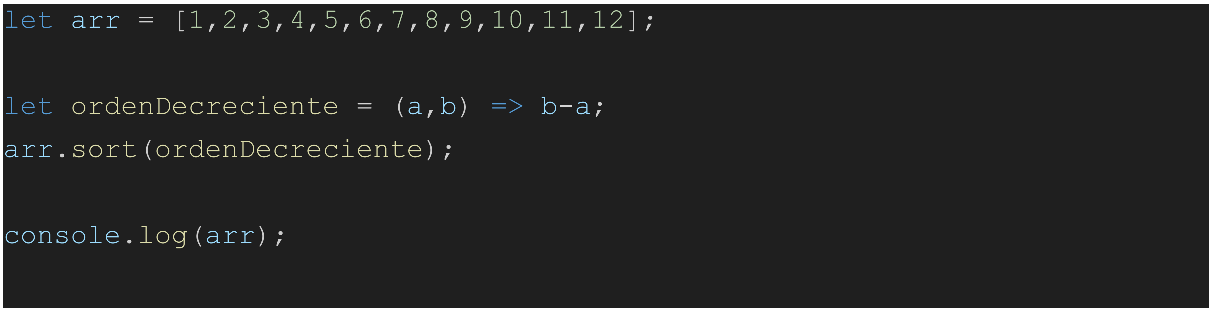
```
let arr = [1,2,4,5,6,7,8,9,10,11,12,13]
function filter(arr, a, b) {
  let arrModificado = new Array;
  for (let i = 0; i < arr.length; i++) {
    if (arr[i] >= a && arr[i] <= b) {
      arrModificado.push(arr[i]);
    }
  }
  return arrModificado;
}

console.log(filter(arr,5,10));
```

3. Write a function filterRangeInPlace(arr, a, b) that gets an array arr and removes from it all values except those that are between a and b. The test is: a ≤ arr[i] ≤ b. The function should only modify the array. It should not return anything.



```
let arr = [1,2,3,4,5,6,7,8,9,10,11,12];

function filterRangeInPlace(arr, a, b) {
  for (let i = arr.length - 1; i >= 0; i--) {
    let elementoActual = arr[i];

    if (elementoActual <= a || elementoActual >= b) {
      arr.splice(i, 1);

    }
  }
}

console.log(filterRangeInPlace(arr,5,12));
```

4. Sort an array in decreasing order



```
let arr = [1,2,3,4,5,6,7,8,9,10,11,12];

let ordenDecreciente = (a,b) => b-a;
arr.sort(ordenDecreciente);

console.log(arr);
```

5. We have an array of strings arr. We'd like to have a sorted copy of it, but keep arr unmodified. Create a function copySorted(arr) that returns such a copy.

```
let palabras = ["A", "B", "D", "C", "Z", "b", "a", "z", "y"];

function copySorted(arr) {
  let copiaArr = Array.from(arr);


  return copiaArr.sort();
}

console.log(copySorted(palabras));
```

6. You have an array of user objects, each one has user.name and user.age. Write the code that converts it into an array of names.

```
let users = [
  {name: "Laura", age:20},
  {name: "Pablo", age:30},
  {name: "Juan", age:26},
];
let nombresEmpleados = [];


users.forEach(function(trabajador) {
  nombresEmpleados.push(trabajador.name);
});

console.log(nombresEmpleados)
```

7. You have an array of user objects, each one has name, surname and id. Write the code to create another array from it, of objects with id and fullName, where fullName is generated from name and surname.

```
let objetos = [
  {name: "Juan", surname: "Perez", id: 123},
  {name: "Pedro", surname: "Gonzalez", id: 12},
  {name: "Carla", surname: "Sanchez", id: 56},
  {name: "Marta", surname: "Lopez", id: 30},
]
```

```
let nuevoObjeto = [];


objetos.forEach(function(empleado) {
  let nombreTrabajador = {fullname: empleado.name + " " +
empleado.surname, id: empleado.id };
  nuevoObjeto.push(nombreTrabajador);
});

console.log(nuevoObjeto);
```

8. Write the function sortByAge(users) that gets an array of objects with the age property and sorts them by age.

```
let objetos = [
  {age: 78},
  {age: 12},
  {age: 56},
  {age: 30},
]

function sortByAge(users) {
  let ordenarEdad = (a,b) => a.age-b.age;

  users.sort(ordenarEdad);
}

sortByAge(objetos);

console.log(objetos);
```

9. Write the function shuffle(array) that shuffles (randomly reorders) elements of the array. All element orders should have an equal probability. For instance, [1,2,3] can be reordered as [1,2,3] or [1,3,2] or [3,1,2] etc, with equal probability of each case.
To do so, use Fisher-Yates shuffle algorithm. The idea is to walk the array in the reverse order and swap each element with a random one before it.

```
let arr = [1,2,3,4,5,6,7,8,9]

function shuffle(array){
  for (let i = array.length - 1; i >= 0; i--) {
    let indiceRandom = Math.round(Math.random() * (i + 1));
    let indiceActual = array[i];
    array[i] = array[indiceRandom];
    array[indiceRandom] = indiceActual;
  }
}

shuffle(arr);

console.log(arr);
```

10. Write the function getAverageAge(users) that gets an array of objects with property age and returns the average age.

```
let users = [
  {name: "Juan", surname: "Perez", age: 15},
  {name: "Pedro", surname: "Gonzalez", age: 25},
  {name: "Carla", surname: "Sanchez", age: 56},
  {name: "Marta", surname: "Lopez", age: 30},
]

function getAverageAge(users){
  let contador = 0;
  let sumaEdades = 0;
  users.forEach(function(elemento){
     sumaEdades = elemento.age + sumaEdades;
     contador ++;
  });
  return sumaEdades / contador;
}

console.log(getAverageAge(users));
```

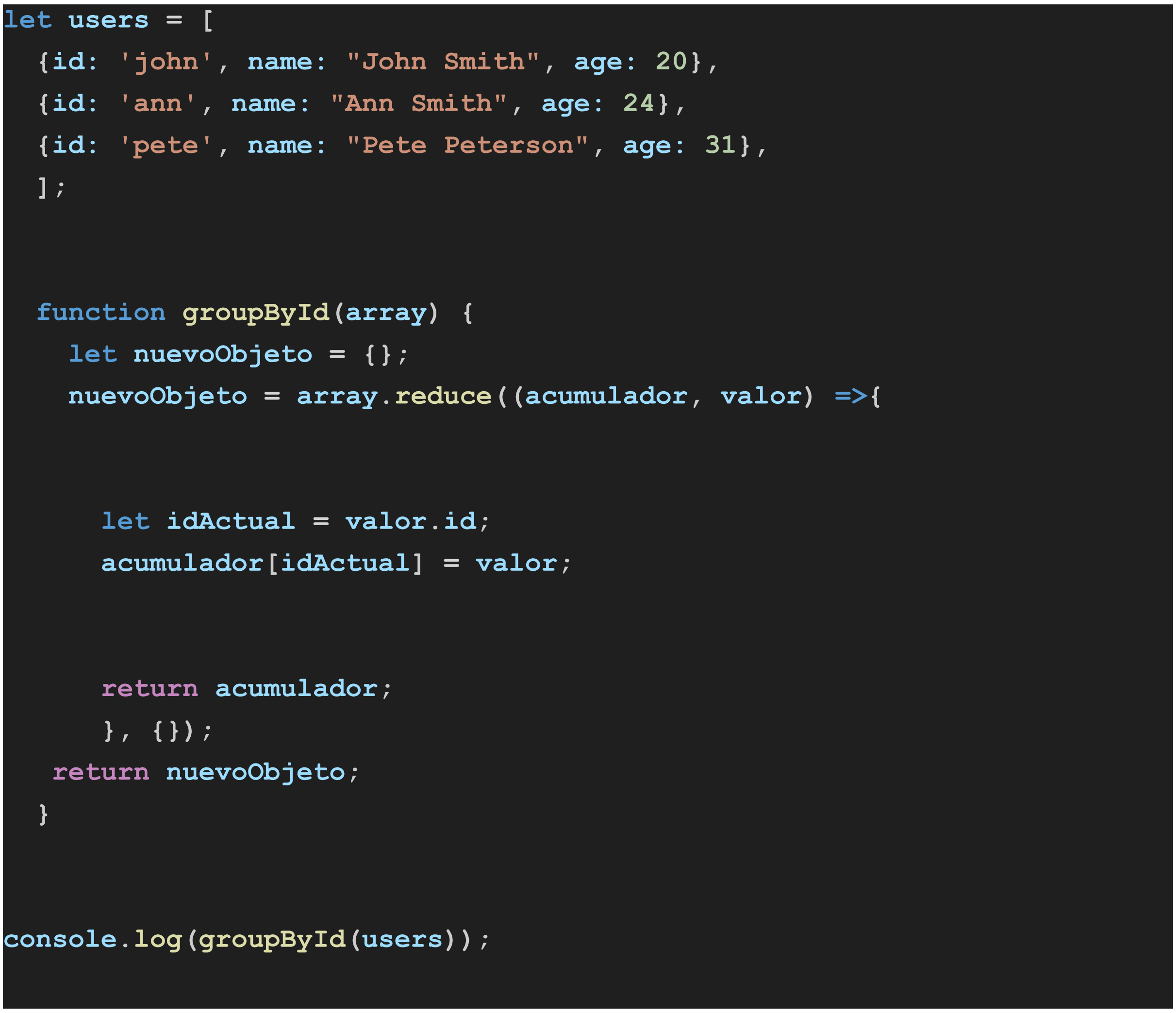
11. Let's say we received an array of users in the form {id:..., name:..., age:... }. Create a function groupById(arr) that creates an object from it, with id as the key, and array items as values. In this task we assume that id is unique. There may be no two array items with the same id. Please use array .reduce method in the solution. For instance:

let users = [

{id: 'john', name: "John Smith", age: 20},
{id: 'ann', name: "Ann Smith", age: 24},
{id: 'pete', name: "Pete Peterson", age: 31},
];

let usersById = groupById(users); /* // after the call we should have:

usersById = {
john: {id: 'john', name: "John Smith", age: 20},
ann: {id: 'ann', name: "Ann Smith", age: 24},
pete: {id: 'pete', name: "Pete Peterson", age: 31},
}
***/**



```
let users = [
  {id: 'john', name: "John Smith", age: 20},
  {id: 'ann', name: "Ann Smith", age: 24},
  {id: 'pete', name: "Pete Peterson", age: 31},
  ];


  function groupById(array) {
    let nuevoObjeto = {};
    nuevoObjeto = array.reduce((acumulador, valor) =>{


      let idActual = valor.id;
      acumulador[idActual] = valor;


      return acumulador;
      }, {});
   return nuevoObjeto;
  }


console.log(groupById(users));
```