Array methods

1. Write the function `camelize(str)` that changes dash-separated words like “my-short-string” 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.

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.

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.

4. Sort an array in decreasing order

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.

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.

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`.

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

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. In order to do so there are some algorithms being Fisher-Yates shuffle algorithm one of the most equality. It consists on walking the array in the reverse order and swapping each element with a random one before it.

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

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},
}
*/
```