Higher-Order Functions

# Data set:

Use the data set from file **JavaScript Advanced 02 - Higher-Order Functions - Exercises - Mock Data.json** to do the following Exercises.

Suppose the data is saved to a variable named **users**.

# Exercise 01: forEach

User forEach to get first\_name and last\_name of all user and put it another array.

The order in new array must be the same order as the user appear in the **users** array.

Expected output:

|  |
| --- |
| [  'Eamon Harhoff',  'Laney Whittam',  'Lynett Twinberrow',  …  ] |

# Exercise 02: filter

User filter to find out which user is male and age is under 40

# Exercise 03: map

Same like Exercise 01 but you must use map

# Exercise 04: Map cont.

Use map to transform **users** array where the key of each record in new array is **camelCase**

Example:

|  |
| --- |
| [  { "id": 1, "firstName": "Eamon", "lastName": "Harhoff", "email": "eharhoff0@imageshack.us", "gender": "Male", "age": 76, "salary": 18888 },  { "id": 2, "firstName": "Laney", "lastName": "Whittam", "email": "lwhittam1@issuu.com", "gender": "Female", "age": 42, "salary": 15018 },  { "id": 3, "firstName": "Lynett", "lastName": "Twinberrow", "email": "ltwinberrow2@gov.uk", "gender": "Female", "age": 99, "salary": 13343 }  …  ] |

# Exercise 05: reduce

Use reduce to calculate the average age in **users**

# Exercise 06: reduce cont.

Use **reduce** to implement Exercise 01 – Exercise 4

# Exercise 07: Implement map

Write a function named **map** that takes an array, and a transformation function.

Map function have the same functionality like Array.prototype.map

# Exercise 08: Implement filter

Write a function named **filter** that takes an array, and a predicate function.

Map function have the same functionality like Array.prototype.filter

# Exercise 09: Implement reduce

Write a function named **reduce** that take an array, a function and a default value.

Map function have the same functionality like Array.prototype.reduce

# Exercise 10: Implement map, filter using reduce

Reuse function **reduce** of Exercise 09 to write function map (Exercise 07) and function filter (Exercise 08) without using loops.

# Exercise 11: Using reduce

1. Use reduce to create function **sum** which will calculate the sum of every item in array.
2. Use reduce to create function **product** which will calculate the product of every item in array.
3. Use reduce to create function **reverse** which will reverse the position of every item in array.