

## ASSIGNMENT: #05 TOPIC: JAVASCRIPT: PART II

---

For the exercises in this assignment, create a new HTML document with a dedicated external script for each exercise.

### EXERCISE 1. WORKING WITH PROTOTYPES AND INHERITANCE

In this exercise, you will practice with prototypes and inheritance in JavaScript.

- i. Define a Person object constructor function that takes a first name (string) and a last name (string) as parameters.
- ii. Use Person to create two new objects representing a person.
- iii. Add a greet method to all persons. The method prints a message along the lines of “Hello, I’m `${FirstName}` `${LastName}`” to standard output. For this exercise, you are **not allowed** to add the method as a property of each of the existing Person object. You are required to use prototype-based inheritance.
- iv. Describe, step by step, what is happening under the hood when you invoke the greet method on one of the Person objects.
- v. Invoke the greet method on each person object and make sure that it works as expected.
- vi. Create a new constructor function Student, inheriting from Person. The Student constructor should take as inputs the same inputs as the Person constructor, plus an additional parameter “degreeProgram” (string). Before setting the degreeProgram property, the Student constructor should call in turn the constructor of Person, passing the other parameters. To do that, you can use the [Function.prototype.call\(\)](#) method. Make sure to set the `[[Prototype]]` of `Student.prototype` to `Person.prototype`.
- vii. Create two new Student objects, representing two persons enrolled in a Computer Science degree program.
- viii. Invoke the greet method on the two newly-created students and ensure that the greet method is inherited from the Person prototype as expected.
- ix. Explain, step by step, what is going on under the hood when the greet method is invoked on a Student object created by the Student constructor.
- x. Override the greet method inherited from the Person prototype for all Student objects. The new implementation of the greet method should print to standard output a message along the lines of “Hello! I’m `${FirstName}` `${LastName}` and I’m a `${degreeProgram}` student”.
- xi. Invoke the greet method on all Person and Student objects, and make sure that the overridden version applies only to students.
- xii. Explain, step by step, what is going on under the hood when the greet method is invoked on an object created using the Student constructor and what happens when the greet method is invoked on an object created using the Person constructor.

## ASSIGNMENT: #05 TOPIC: JAVASCRIPT: PART II

### EXERCISE 2. MULTI-STEP FILTERING FUNCTION

Your task for this exercise is to define a `createMultiStepFilterer` function that takes as input an array, and returns a new function (`filterer`) that can be used to iteratively filter the elements of the array.

In particular, the `filterer` function takes as input a `filterCriterion` function. The `filterCriterion` function takes as input an element of the array and returns a Boolean value. Array elements for whom the `filterCriterion` function returns `false` should be removed from the array. If the provided `filterCriterion` is not a function, the `filterer` function returns the current array as is, and no filtering is performed. Moreover, the `filterer` function must not change the original array.

To solve this exercise, use the [`filter\(\)`](#) method defined in `Array.prototype`. Start by looking at the documentation on MDN (link above).

The following snippet of code demonstrates the usage of the `createMultiStepFilterer` function.

```
let arr = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
let filterer = createMultiStepFilterer(arr);

console.log(filterer()); //Array(10) [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

console.log(filterer(function(elem){
  return elem % 2 !== 0 //remove elements that are even
})); //Array(5) [1, 3, 5, 7, 9]

console.log(filterer("Foo")); //Array(5) [1, 3, 5, 7, 9]

console.log(filterer(function(elem){
  return elem % 3 !== 0 //remove elements that are also multiple of three
})); //Array(3) [1, 5, 7]

console.log(arr); //Array(10) [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

### EXERCISE 3. MULTI-STEP FILTERING — FROM SCRATCH

Re-implement the same multi-step filtering function, this time without using the [`filter\(\)`](#) from `Array.prototype`.