IMY 220
Assignment 3:
ES6

Due: Wednesday 9 September at 13:00.

The submission instructions are available on ClickUP. Any deviation from these instructions will cause a 10% deduction from your mark.

Instructions

- Create two class definitions for handling Euclidian geometry functions: a class definition for a Euclidian point in n-dimensional space and a class definition for a list of such points.
- Download *index.html* and *script.js* from ClickUP. These contain the basic HTML for a valid HTML5 document and instances of *EuclidianPoint* objects. You must include *script.js* inside *index.html*.
- You must create two new files called *EuclidianPoint.js* and *EuclidianPointList.js* which must contain your class definitions for the two classes as discussed below.
- You are not allowed to write any loops for this assignment or use the *.foreach()* function. All of the functionality must be implemented with the appropriate JS Array functions as discussed in the slides and resources linked from the slides.
  Note: where you are unable to complete functionality using JS Array functions, you will still receive some marks for completing the functionality with loops. You will receive more marks for functionality that works according to the requirements but uses loops than for code that uses JS Array functions but does not work.
- You must use ES6 syntax discussed in class wherever appropriate. This includes class definitions, arrow functions, template strings, and variable declarations. (Note that if you create all the required functionality using JS Array functions you will not need a single variable declaration in either of your class definitions.)

Part 1: EuclidianPoint class

Create a class called EuclidianPoint which receives the n-dimensional coordinates for the point as an array of numbers for its constructor. The constructor should use a try-catch block to check that it received parameters and that each element in the array received is a numeric value. Only if both these conditions are met should it set a member variable to the array of coordinates received.

HINT: use the Array.some() function to check for numeric values.

The class should also contain a setter to allow for setting the coordinates as follows (where p1 is a EuclidianPoint object):

```
p1.coordinates = [1, 2];
```

The class should also contain a function called *calculateDistance()* which must calculate the Euclidian distance between the current point, i.e. the one for which the current instance is being defined, and another point, which is given in the function parameter. Given two two-dimensional points in space, x and y, the Euclidian distance between them is defined as:

$$\sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

…where `x1` refers to x's first coordinate, `x2` refers to x's second coordinate, etc. In other words, if two *EuclidianPoint* instances were created with the following arrays: [1, 1] and [2, 2], the distance would be 1.4142135623730951.

Note that the JS array functions cannot loop through two arrays simultaneously, you will need to call the function on one array and refer to the other in the function body. Also note that for the purposes of this function you may assume that the two points will always have the same number of coordinates.

Part 2: EuclidianPointList

Create a class called EuclidianPointList which receives an array of EuclidianPoint objects for its constructor. The constructor should use a try-catch block to check it received parameters and that each EuclidianPoint object given has the same number of coordinates. Only if both these conditions are met should it set a member variable to the array of EuclidianPoints received.

The EuclidianPointList class should provide two getter functions. The first should return an array of all the objects whose coordinates are all positive values. For example, when creating a EuclidianPointList with three objects that have [1, 1], [2, -1], and [2, 2] as their coordinates, the following…

```
list.positivePoints
```

…should return an array of the first and third object only.

The EuclidianPointList class should also provide a getter which calculates the cumulative distance between all EuclidianPoint objects in its array of objects. For example, the following coordinates…

```
[1, 1], [2, 2], [2, 3]
```

…give a cumulative distance between them of 2.414213562373095. You must use the *calculateDistance()* function from the *EuclidianPoint* class to accomplish this.

Note about testing: this assignment does not have a DOM-aspect, thus you are expected to do your own testing using *console.log*. However, make sure to take out all *console.log* statements before submitting as only your class definition and Array prototype extensions will be marked.

Bonus

Use the ES6 module functionality to export the classes from *EuclidianPoint.js* and *EuclidiantPointList.js* and import them inside *script.js*.

Additional Information

- Refer to the slides and online resources for help

Submit only the following file(s) according to the submission instructions.

- *index.html*
- *script.js*
- *EuclidianPoint.js*
- *EuclidianPointList.js*