# JS Advanced: Exam 15 July 2018

# Problem 2. Calculator (Unit Testing)

You are given the following **JavaScript class**:

|  |
| --- |
| Calculator.js |
| **class** Calculator {  constructor() {  **this**.**expenses** = [];  }   add(data) {  **this**.**expenses**.push(data);  }   divideNums() {  **let** divide;  **for** (**let** i = 0; i < **this**.**expenses**.length; i++) {  **if** (**typeof** (**this**.**expenses**[i]) === **'number'**) {  **if** (i === 0 || divide===***undefined***) {  divide = **this**.**expenses**[i];  } **else** {  **if** (**this**.**expenses**[i] === 0) {  **return 'Cannot divide by zero'**;  }  divide /= **this**.**expenses**[i];  }  }  }  **if** (divide !== ***undefined***) {  **this**.**expenses** = [divide];  **return** divide;  } **else** {  **throw new** Error(**'There are no numbers in the array!'**)  }  }  toString() {  **if** (**this**.**expenses**.length > 0)  **return this**.**expenses**.join(**" -> "**);  **else return 'empty array'**;  }  orderBy() {  **if** (**this**.**expenses**.length > 0) {  **let** isNumber = **true**;  **for** (**let** data **of this**.**expenses**) {  **if** (**typeof** data !== **'number'**)  isNumber = **false**;  }  **if** (isNumber) {  **return this**.**expenses**.sort((a, b) => a - b).join(**', '**);  }  **else** {  **return this**.**expenses**.sort().join(**', '**);  }  }  **else return 'empty'**;  } } |

### Functionality

The above code defines a **class** that holds items (of **any** type). An **instance** of the class should support the following operations:

* Contains a property expenses that is initialized to an **empty** array.
* Functionadd(data) – **adds** the passed in **item** (of **any** type) to the expenses.
* Function **divideNums()** – divides **only** the **numbers** from the **expenses** and returns the result. If there are no numbers in the array, the function throws the following error: "**There are no numbers in the array!**"
* FunctiontoString()– **returns** a string, containing a list of all items from the expenses, joined with

an **arrow: " -> "**. If there are **no** items stored, it should **return** the string **"empty array".**

* Function **orderBy()** – **returns a string joined with ", "** which is the **sorted expenses,** sorting them by **two criteria** - one for **numbers** and another for **mixed** data.

### Examples

This is an example how this code is **intended to be used**:

|  |
| --- |
| Sample code usage |
| **let *output*** = **new** Calculator(); ***output***.add(10); ***output***.add(**"Pesho"**); ***output***.add(**"5"**); ***console***.log(***output***.toString()); ***output***.add(10); ***console***.log(***output***.divideNums()); ***output***.add(1); ***console***.log(***output***.orderBy()); ***console***.log(***output***.toString()); |
| Corresponding output |
| 10 -> Pesho -> 5  1  1, 1  1 -> 1 |

### Your Task

Using **Mocha** and **Chai** write **JS unit tests** to test the entire functionality of the Calculator class. You may use the following code as a template:

|  |
| --- |
| describe(**"*TODO* …"**, **function**() {  ***it***(**"*TODO …*"**, **function**() {  *//* ***TODO:*** …  });  *//* ***TODO:*** …  }); |

### Submission

Submit your tests inside a describe() statement, as shown above.

### Notes

The methods should function correctly for **positive**, **negative** and **floating point** numbers. In case of **floating point** numbers the result should be considered correct if it is **within 0.01** of the correct value.

**There will be no function chaining.**