# JS Advanced: Exam 13 November 2016

Problems for exam preparation for the [“JavaScript Advanced” course @ SoftUni](https://softuni.bg/courses/javascript-advanced). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/355/>.

# Problem 2. Add / Swap / Shift Left / Right in List (Unit Testing)

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

|  |
| --- |
| list-add-swap-shift-left-right.js |
| **function** *createList*() {  **let** data = [];  **return** {  add: **function** (item) {  data.push(item)  },  shiftLeft: **function** () {  **if** (data.**length** > 1) {  **let** first = data.shift();  data.push(first);  }  },  shiftRight: **function** () {  **if** (data.**length** > 1) {  **let** last = data.pop();  data.unshift(last);  }  },  swap: **function** (index1, index2) {  **if** (!***Number***.isInteger(index1) || index1 < 0 || index1 >= data.**length** ||  !***Number***.isInteger(index2) || index2 < 0 || index2 >= data.**length** ||  index1 === index2) {  **return false**;  }  **let** temp = data[index1];  data[index1] = data[index2];  data[index2] = temp;  **return true**;  },  toString: **function** () {  **return** data.join(**", "**);  }  }; } |

### Functionality

The above code creates a **list** data structure that holds items (of any type). It supports the following operations:

* add(item) – **appends** given item to the end of the list.
* shiftLeft() – shifts all elements **one position left** and the first elements comes last (with **rotation**).
* shiftRight() – shifts all elements **one position right** and the last elements comes first (with **rotation**).
* swap(index1, index2) – swaps the items at the specified indexes and returns **true**. If any of the two indexes **does not exist** or they are **equal** the collection stays **unchanged** and the method returns **false**.
* toString() – returns the string representations of the **list items**, separated by “, “.

### Examples

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

|  |  |  |
| --- | --- | --- |
| Sample code usage |  | Corresponding output |
| **let *list*** = *createList*(); ***list***.add(1); ***list***.add(**"two"**); ***list***.add(3); **console**.log(**`list = [**${***list***}**]`**); ***list***.shiftLeft(); **console**.log(**"shifted left <--"**); **console**.log(**`list = [**${***list***}**]`**); ***list***.add([**"four"**]); **console**.log(**`list = [**${***list***}**]`**); ***list***.shiftRight(); **console**.log(**"shifted right -->"**); **console**.log(**`list = [**${***list***}**]`**); **console**.log(**`Swaping [0] and [3]:** ${***list***.swap(0,3)}**`**); **console**.log(**`list = [**${***list***}**]`**); **console**.log(**`Swaping [1] and [1]:** ${***list***.swap(1,1)}**`**); **console**.log(**`list = [**${***list***}**]`**); | list = [1, two, 3]  shifted left <--  list = [two, 3, 1]  list = [two, 3, 1, four]  shifted right -->  list = [four, two, 3, 1]  Swaping [0] and [3]: true  list = [1, two, 3, four]  Swaping [1] and [1]: false  list = [1, two, 3, four] |

### Your Task

Using **Mocha** and **Chai** write **JS unit tests** to test the entire functionality of the list object. Your code will only be provided the createList function, how you test the list is entirely up to you - whether you create a new list before each test or share the same list between tests.

You should have at least **6 test cases**, make sure you cover all **edge cases**. You may use the following code as a template:

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

### Submission

Submit **only your tests** as “**JavaScript code (Unit Tests with Sinon and Mocha)**”.