Test for JavaScript Developer

Create a function `sortProductsByPrice` that takes a list of products and returns an object containing products with the highest and the lowest prices. The test contains 3 subtasks: in every next task you extend your function with new conditions. Submit your function for every task.

Task 1.

Start with the basic code. Create a function `sortProductsByPrice` that finds 5 products with the highest prices and 5 products with the lowest prices from the product array.

```
const products = [
    {id: 1, price: 10}, {id: 2, price: 11}, {id: 3, price: 1}, {id: 4,
price: 3},  {id: 5, price: 1},  {id: 6, price: 8},
    {id: 7, price: 3},  {id: 8, price: 0},  {id: 9, price: 4}, {id: 10,
price: 5}, {id: 11, price: 9}, {id: 12, price: 13},
];


/**
 * @params [Array] products - list of products
 * @params [Number] options.size - Optional parameter. By default it
should be 5
 **/
function sortProducts(products, options) {
//...
}

const result = sortProducts(products); // {highest: [...], lowest:
[...]}
```

Task 2.

If there are less products than `options.size` the priority should be given to fill the array with the highest prices. If no elements left `null` should be returned.

```
const products = [
    {id: 1, price: 10},
    {id: 2, price: 11},
    {id: 3, price: 1}
];

const result = sortProducts(products, {size: 4}); // {highest: null,
lowest: null}
const result = sortProducts(products, {size: 3}); // {highest: [...],
lowest: [null}
const result = sortProducts(products, {size: 2}); // {highest: [...],
lowest: [...]}
const result = sortProducts(products, {size: 1}); // {highest: [...],
lowest: [...]}
const result = sortProducts(products, {size: 0}); // {highest: null,
lowest: null}
```

Task 3.

When we call the function with unmodified params it should return `null` in the data fields

```
const products = [
    {id: 1, price: 10},
    {id: 2, price: 11},
    {id: 3, price: 1},
    {id: 4, price: 2},
    {id: 5, price: 100},
    {id: 6, price: 0.1}
];

const result1 = sortProducts(products); // {highest: [...], lowest:
[...]}

// call without modifications
const result2 = sortProducts(products); // {highest: null, lowest: null}

products[1] = {id: 2, price: 11.5};

// call with modified data
const result2 = sortProducts(products); // {highest: [...], lowest:
[...]}

// call without modifications
const result3 = sortProducts(products); // {highest: null, lowest: null}

products.push({...});

// call with modified data
const result4 = sortProducts(products); // {highest: [...], lowest:
[...]}
```

Requirements

- ES2015+ or Typescript
- Vanilla JS, without any libraries or frameworks
- Unit tests