**Searching and Sorting: Final Assignment**

You are working on a customer management system for a large retail organization. They have been collecting customer data for some time and now has data for tens of thousands of customers in an array.

Your task is to write a few helper functions that will facilitate manipulating the customer data.

**Instructions**

Your goal for this checkpoint is to get the tests to pass.  
To do so, you will be implementing the algorithms for each of the files listed below.

**Existing files**

| **file path** | **description** |
| --- | --- |
| \_\_tests\_\_ | Contains the tests necessary to verify the search and sort algorithms. You will not need to make changes to the files in this folder. |
| src/sort.js | Change this file to implement the solution to the sorting problem. |
| src/sortByName.js | Change this file to implement the solution to the Sort by Name problem. |
| src/searchByEmail.js | Change this file to implement the solution to the Search by Email problem. |
| src/searchByName.js | Change this file to implement the solution to the Search by Name problem. |
| src/data.json | An sample of the array of data that will be manipulated |

**The Data**

The array of customers takes the form:

[

{

"username": "Maude.Torp",

"email": "Taya.Kerluke53@gmail.com",

"address": {

"street": "Rowe Fields",

"suite": "Suite 231",

"city": "Tiannamouth",

"zipcode": "07584-6653",

"geo": { "lat": "75.0283", "lng": "-17.1824" }

},

"phone": "795-827-5446 x18366",

"website": "nico.com",

"company": {

"name": "Champlin, Feest and Barrows",

"catchPhrase": "Object-based user-facing orchestration",

"bs": "transition integrated content"

},

"firstName": "Maida",

"lastName": "Feeney"

},

{

"username": "Brendan\_Lehner",

"email": "Chelsey.Little@gmail.com",

"address": {

"street": "Kuhic Mews",

"suite": "Suite 469",

"city": "Chadrickview",

"zipcode": "60257-0239",

"geo": { "lat": "75.9918", "lng": "-118.5832" }

},

"phone": "1-336-548-0143",

"website": "olga.com",

"company": {

"name": "Treutel, Schmitt and Marvin",

"catchPhrase": "Open-source exuding algorithm",

"bs": "envisioneer interactive partnerships"

},

"firstName": "Golden",

"lastName": "Fahey"

},

...

]

**Tasks**

Complete the following tasks to pass the tests and this assignment.

**1. Sorting**

In order to present the data in a meaningful way it would be necessary to sort the array. Implement a sorting algorithm of your choice. Your sorting function should be named sort and should accept two parameters.

function sort(compare, elements) {}

Where:

* **compare:** A function that compares two elements, so it takes two parameters: left and right. It returns one of the following:
  + 0 if the left element is equal to the right element
  + A positive value if the left element is greater than the right element by the ordering criterion
  + A negative value if the left element is less than the right element by the ordering criterion
* **elements:** The array to which the sort algorithm is applied.

The sorting algorithm must be stable.

**2. Sort by name**

The company wants the entire array of customers sorted by last name and first name. That is, the entire array should be sorted by last name. But, if more than one persons have the same last name then they should be sorted by first name.

For example, Amy Bogan comes before Beatrice Bogan who comes before Amy Crooks.

Write a function named nameSort that accepts the array of elements to be sorted and returns the array sorted as described. The nameSort function should use the sort function defined in task 1 to accomplish the task. Hint: you may consider calling the sort function twice.

**3. Search By Name**

Write a function named searchByName that accepts three parameters.

* **firstName** the first name of a customer
* **lastName** the last Name of a customer
* **customers** the array of customers sorted by first and last name

The function should implement a binary search and return the index of the customer with the given first and last names. If no customer with those names are found then return -1.

**4. Search by Email**

Write a function named searchByEmail that accepts two parameters.

* **email** the email address to search for
* **customers** the array of customers sorted by first and last name

The function should return the index of the customer with the given email address. If no customer with that email address is found return -1.

Once these tasks are complete, all tests should pass.