OOP Model Test

Exercise: Number Manipulation with Map

Write a JavaScript function called `manipulateNumbers` that takes in an array of numbers called `numbers` and a string called `operation`.

The `operation` string will either be `"square"`, `"cube"`, or `"sqrt"`.

- If the `operation` is `"square"`, the function should return a new array with the squares of the numbers in the `numbers` array.
- If the `operation` is `"cube"`, the function should return a new array with the cubes of the numbers in the `numbers` array.
- If the `operation` is `"sqrt"`, the function should return a new array with the square roots of the numbers in the `numbers` array.

Use the `map` function to implement the array transformations.

Example usage:

```
const numbers = [1, 2, 3, 4, 5];
const squares = manipulateNumbers(numbers, 'square');
console.log(squares); // Output: [1, 4, 9, 16, 25]

const cubes = manipulateNumbers(numbers, 'cube');
console.log(cubes); // Output: [1, 8, 27, 64, 125]

const roots = manipulateNumbers(numbers, 'sqrt');
console.log(roots); // Output: [1, 1.4142135623730951,
1.7320508075688772, 2, 2.23606797749979]
```

Note:

- You can assume that the `numbers` array will only contain numeric values.
- You can use built-in JavaScript methods to perform the numerical operations, or you can write your own implementation if you prefer.
- The original `numbers` array should not be modified; instead, the function should return a new array with the transformed values.

Exercise: List Manipulation

Write a JavaScript function called `manipulateList` that takes in two arguments: an array called `list` and a string called `operation`.

The `list` array will contain a series of strings.

The `operation` string will either be `"reverse"`, `"sort"`, or `"extract"`.

- If the `operation` is `"reverse"`, the function should reverse the order of the strings in the `list` array.
- If the `operation` is `"sort"`, the function should sort the strings in alphabetical order.
- If the `operation` is `"extract"`, the function should extract a random subarray of strings in the `list` array.

The function should return the modified `list` array.

Example usage:

```
const myList = ['apple', 'banana', 'cherry', 'orange'];
const reversedList = manipulateList(myList, 'reverse');
console.log(reversedList);
// Output: ['orange', 'cherry', 'banana', 'apple']

const sortedList = manipulateList(myList, 'sort');
console.log(sortedList);
// Output: ['apple', 'banana', 'cherry', 'orange']

const shuffledList = manipulateList(myList, 'extract');
console.log(shuffledList);
// Output: ['banana', 'cherry']
```

Note:

- You can assume that the `list` array will only contain string values.
- You can use built-in JavaScript methods to perform the list manipulation operations, or you can write your own implementation if you prefer.
- The original `list` array should not be modified; instead, the function should return a new array with the modified order of elements.

Exercise: Adding and Removing Items from a Shopping List

You've been tasked with building a simple shopping list application. The application should allow users to add items to a list, remove items from the list, and display the current list of items.

Requirements

1. When the user submits the form, the item should be added to an array of objects that represents the shopping list. Each object should have two properties: a `name` property that contains the name of the item, and a `quantity` property that contains the quantity of the item.

2. The shopping list should be displayed as an unordered list on the page. Each item in the list should display the name of the item and the quantity.
3. When the user clicks on an item in the list, the item should be removed from the array of objects and the list should be updated on the page.

Instructions

1. Create an HTML file with a form that includes input fields for the name and quantity of the item, as well as a submit button.
2. Create a JavaScript file that:
    - Defines an empty array to store the shopping list items
    - Defines a function that handles the form submission event. This function should:
        - Create a new object with the name and quantity properties based on the form inputs
        - Push the new object to the shopping list array
        - Clear the form inputs
        - Call the function that updates the list on the page
    - Defines a function that updates the list on the page. This function should:
        - Get the unordered list element from the DOM
        - Clear any existing list items from the unordered list
        - Loop through the shopping list array and create a new list item for each item in the array
        - Add the name and quantity properties to the list item
        - Add an event listener to each list item that calls a function to remove the item from the shopping list array
        - Append the list item to the unordered list
    - Defines a function that removes an item from the shopping list array. This function should:
        - Remove the item from the shopping list array based on the index of the clicked list item
        - Call the function that updates the list on the page
3. Add event listeners to the form submit button and the document's DOMContentLoaded event that call the appropriate functions.