You will be using this as an event listener for the sortButton. Because buttons associated with a form element submit by default, you need to prevent that behavior. Call event.preventDefault() in your function to do this.

**Step 5**

Back in your sortInputArray function, you need to get the values from your select elements. Since they all have the class values-dropdown, you can query them all at once.

Use document.getElementsByClassName() to get all the elements with this class by passing in the argument "values-dropdown". Assign that to an inputValues variable with const

Remember that .getElementsByClassName() method returns an HTMLCollection, which is an array-like object of all the elements that have a matching class name. You can use the spread operator to convert it into an array.

Convert the document.getElementsByClassName() call to an array with the spread operator and assign it to a variable called inputValues.

You should use console.log() to print out the result of inputValues. Write the code for this inside the sortInputArray function.

To see the logged inputValues array, click on the sort button and open up the console. You should see an array of strings like this:

Example Code

[ "8", "2", "4", "1", "3" ]

Before going further, make sure you observe the data type of the printed result in the console.

In the next step, you will convert those strings into numbers.

**Step 11**

You need a function to update the display with the sorted numbers. Start by using arrow syntax to declare an updateUI function that takes a single array parameter.

Because you will be writing algorithms that won't immediately have a return value, set a fallback value for array to be an empty array. Here is an example of a function that has a fallback value:

Example Code

const myFunction = (string = "") => {

}



Check Your Code (Ctrl + Enter)

Reset



Navigated to Step 11