Step 10

Your select menu needs options for each of the food and exercise `fieldset` elements you created in the previous steps. Use the `option` element to create a new option for each `fieldset`. The `value` attribute of each option should be the `id` of the `fieldset`, and the text of each option should be the text of the `legend`.

Set the `Breakfast` option as the `selected` option.

```
<div class="controls">
  <span>
    <label for="entry-dropdown">Add food or exercise:</l
    <select id="entry-dropdown" name="options">
      <option value="breakfast" selected>Breakfast</opti
      <option value="lunch">Lunch</option>
      <option value="dinner">Dinner</option>
      <option value="snacks">Snacks</option>
      <option value="exercise">Exercise</option>
    </select>
    <button type="button" id="add-entry">Add Entry</butt
  </span>
</div>
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

Step 20

The current pattern will match the exact text `"hello"`, which is not the desired behavior. Instead, you want to search for `+`, `-`, or spaces. Replace the pattern in your `regex` variable with `\+-` to match plus and minus characters.

Note that you need to use the `backslash` `\` character to *escape* the `+` symbol because it has a special meaning in regular expressions.

```
9  function cleanInputString(str) {
10   const regex = /\+-/;
11 }
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

Step 30

The `e` in a number input can also be an uppercase `E`. Regex has a flag for this, however – the `i` flag, which stands for "insensitive".

```
/Hello/i
```

The following regex would match `hello`, `Hello`, `HELLO`, and even `hEllo` because of the `i` flag. This flag makes your pattern case-insensitive.

Add the `i` flag to your regex pattern.

```
14 function isInvalidInput(str) {
15   const regex = /e/i;
16 }
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

Step 40

Now you need to target the `.input-container` element within the element that has your `targetId`. Declare a new `targetInputContainer` variable, and assign it the value of `document.querySelector()`. Use concatenation to separate `targetId` and `'.input-container'` with a space, and pass that string to `querySelector()`.

```
19 function addEntry() {
20   const targetId = '#' + entryDropdown.value;
21   const targetInputContainer = document.querySelector(ta
22 }
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

Step 50

Finally, on a new line after your second `label`, create another `input` element. Give this one a `type` attribute set to `number`, a `min` attribute set to `0` (to ensure negative calories cannot be added), a `placeholder` attribute set to `Calories`, and an `id` attribute that matches the `for` attribute of your second `label` element.

```
19 function addEntry() {
20   const targetInputContainer = document.querySelector(`#
21   const entryNumber = targetInputContainer.querySelecto
22   const HTMLString = `
23   <label for="${entryDropdown.value}-${entryNumber}-name
24   <input type="text" id="${entryDropdown.value}-${entryN
25   <label for="${entryDropdown.value}-${entryNumber}-calo
26   <input
27     type="number"
28     min="0"
29     id="${entryDropdown.value}-${entryNumber}-calories"
30     placeholder="Calories"
31   />`;
32 }
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

Step 60

Remember that you wrote a function earlier to clean the user's input? You'll need to use that function here.

Update your `currVal` declaration to be the result of calling `cleanInputString` with `item.value`.

```
39     const currVal = cleanInputString(item.value);
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

Step 70

Using that same syntax, query your `number` inputs in the `#lunch` element and assign them to `lunchNumberInputs`.

```
35 function calculateCalories(e) {
36   e.preventDefault();
37   isError = false;
38
39   const breakfastNumberInputs = document.querySelectorAl
40   const lunchNumberInputs = document.querySelectorAll('#
41 }
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.

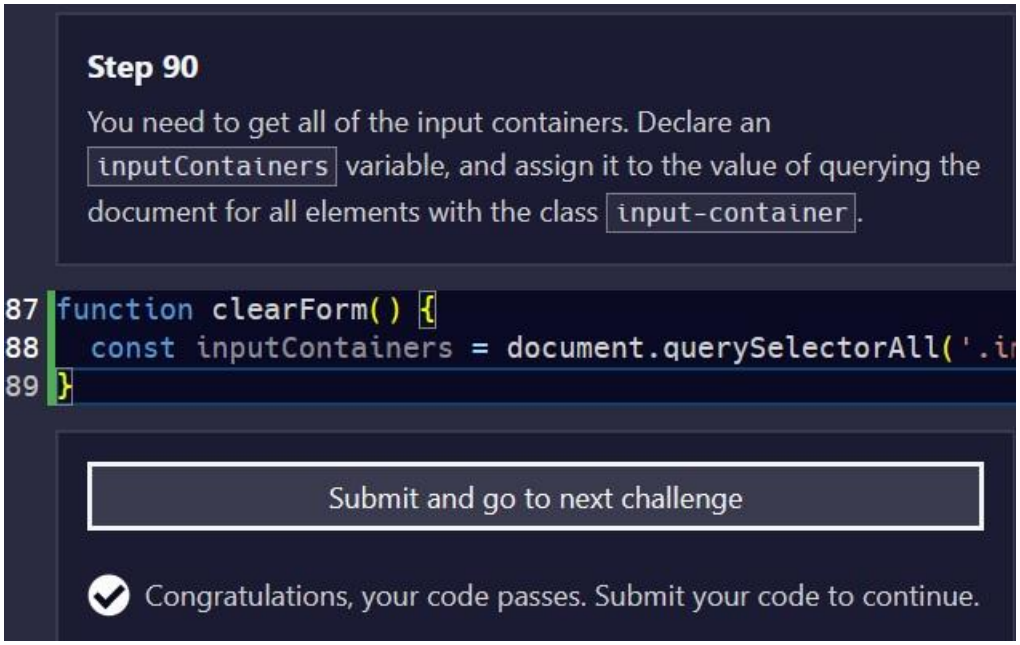
Step 80

You need to construct the HTML string that will be displayed in the `output` element. Start by assigning an empty template literal to the `innerHTML` property of the `output` element on a new line at the end of the function.

```
35 function calculateCalories(e) {
36   e.preventDefault();
37   isError = false;
38
39   const breakfastNumberInputs = document.querySelectorA
40   const lunchNumberInputs = document.querySelectorAll('
41   const dinnerNumberInputs = document.querySelectorAll(
42   const snacksNumberInputs = document.querySelectorAll(
43   const exerciseNumberInputs = document.querySelectorAl
44
45   const breakfastCalories = getCaloriesFromInputs(break
46   const lunchCalories = getCaloriesFromInputs(lunchNumb
47   const dinnerCalories = getCaloriesFromInputs(dinnerNu
48   const snacksCalories = getCaloriesFromInputs(snacksNu
49   const exerciseCalories = getCaloriesFromInputs(exerci
50   const budgetCalories = getCaloriesFromInputs([budgetN
51
52   if (isError) {
53     return;
54   }
55
56   const consumedCalories = breakfastCalories + lunchCal
57   const remainingCalories = budgetCalories - consumedCa
58   const surplusOrDeficit = remainingCalories < 0 ? 'Sur
59   output.innerHTML = ``;
60 }
```

Submit and go to next challenge

Congratulations, your code passes. Submit your code to continue.


Step 90
You need to get all of the input containers. Declare an inputContainers variable, and assign it to the value of querying the document for all elements with the class input-container.
87 function clearForm() {
88 const inputContainers = document.querySelectorAll('.i
89 }
Submit and go to next challenge
Congratulations, your code passes. Submit your code to continue.

Learn Form Validation by Building a Calorie Counter

Sometimes when you're coding a web application, you'll need to be able to accept input from a user. In this calorie counter project, you'll learn how to validate user input, perform calculations based on that input, and dynamically update your interface to display the results.

In this practice project, you'll learn basic regular expressions, template literals, the `addEventListener()` method, and more.