**Crea una calculadora con JavaScript**

**Note:** **React 18 has known incompatibilities with the tests for this project (see** [**issue**](https://github.com/freeCodeCamp/freeCodeCamp/issues/45922)**)**

**Objective:** Build an app that is functionally similar to this: <https://javascript-calculator.freecodecamp.rocks/>.

Fulfill the below user stories and get all of the tests to pass. Use whichever libraries or APIs you need. Give it your own personal style.

You can use any mix of HTML, JavaScript, CSS, Bootstrap, SASS, React, Redux, and jQuery to complete this project. You should use a frontend framework (like React for example) because this section is about learning frontend frameworks. Additional technologies not listed above are not recommended and using them is at your own risk. We are looking at supporting other frontend frameworks like Angular and Vue, but they are not currently supported. We will accept and try to fix all issue reports that use the suggested technology stack for this project. Happy coding!

**User Story #1:** My calculator should contain a clickable element containing an = (equal sign) with a corresponding id="equals".

**User Story #2:** My calculator should contain 10 clickable elements containing one number each from 0-9, with the following corresponding IDs: id="zero", id="one", id="two", id="three", id="four", id="five", id="six", id="seven", id="eight", and id="nine".

**User Story #3:** My calculator should contain 4 clickable elements each containing one of the 4 primary mathematical operators with the following corresponding IDs: id="add", id="subtract", id="multiply", id="divide".

**User Story #4:** My calculator should contain a clickable element containing a . (decimal point) symbol with a corresponding id="decimal".

**User Story #5:** My calculator should contain a clickable element with an id="clear".

**User Story #6:** My calculator should contain an element to display values with a corresponding id="display".

**User Story #7:** At any time, pressing the clear button clears the input and output values, and returns the calculator to its initialized state; 0 should be shown in the element with the id of display.

**User Story #8:** As I input numbers, I should be able to see my input in the element with the id of display.

**User Story #9:** In any order, I should be able to add, subtract, multiply and divide a chain of numbers of any length, and when I hit =, the correct result should be shown in the element with the id of display.

**User Story #10:** When inputting numbers, my calculator should not allow a number to begin with multiple zeros.

**User Story #11:** When the decimal element is clicked, a . should append to the currently displayed value; two . in one number should not be accepted.

**User Story #12:** I should be able to perform any operation (+, -, \*, /) on numbers containing decimal points.

**User Story #13:** If 2 or more operators are entered consecutively, the operation performed should be the last operator entered (excluding the negative (-) sign). For example, if 5 + \* 7 = is entered, the result should be 35 (i.e. 5 \* 7); if 5 \* - 5 = is entered, the result should be -25 (i.e. 5 \* (-5)).

**User Story #14:** Pressing an operator immediately following = should start a new calculation that operates on the result of the previous evaluation.

**User Story #15:** My calculator should have several decimal places of precision when it comes to rounding (note that there is no exact standard, but you should be able to handle calculations like 2 / 7 with reasonable precision to at least 4 decimal places).

**Note On Calculator Logic:** It should be noted that there are two main schools of thought on calculator input logic: *immediate execution logic* and *formula logic*. Our example utilizes formula logic and observes order of operation precedence, immediate execution does not. Either is acceptable, but please note that depending on which you choose, your calculator may yield different results than ours for certain equations (see below example). As long as your math can be verified by another production calculator, please do not consider this a bug.

**EXAMPLE:** 3 + 5 x 6 - 2 / 4 =

* **Lógica de ejecución inmediata:** 11.5
* **Fórmula/Expresión lógica:** 32.5

You can build your project by [using this CodePen template](https://codepen.io/pen?template=MJjpwO) and clicking Save to create your own pen. Or you can use this CDN link to run the tests in any environment you like: https://cdn.freecodecamp.org/testable-projects-fcc/v1/bundle.js

Once you're done, submit the URL to your working project with all its tests passing.