HTML, CSS, JavaScript

HTML

1. Create the file index.html with the elements needed for a calculator:

```
Element: div
Class: “calculator”
```

a. A text área to show the results

```
Element: input
Type: text
Class: "display"
Value: 0 (starting value)
Disabled (the user can not insert text)
```

b. Calculator keys

```
Element: div
Class: “keys”
```

i. Operation symbols: sum (+), subtract (-), multiply (x) e divide (÷)

```
Element: button
Type: button
Class: "operator"
Value: +  -  x  /  (according to the operation)
Content: +  -  &times;  &divide; (according to the operation)
```

ii. Numbers: 0 to 9

```
Element: button
Type: button
Value: 0 … 9 (according to the number)
Content: 0 … 9 (according to the number)
```

c. Decimal symbol (.)

```
Element: button
Type: button
Class: "decimal"
Value: .
Content: .
```

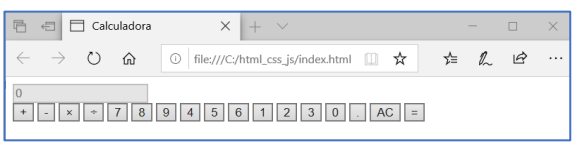
d. Clear (AC)

```
Element: button
Type: button
Class: "clear"
Value: clear
Content: AC
```

e. Equals (=)

```
Element: button
Type: button
Class: "equals"
Value: equals
Content: =
```

Expected result



CSS

1. Create the file style.css and define the following styles:

```
html {
  font-size: 62.5%;
  box-sizing: border-box;
}
*, *::before, *::after {
  margin: 0;
  padding: 0;
  box-sizing: inherit;
}
```

2. Write <link rel="stylesheet" href="estilo.css"> in the index.html file header
3. Define Styles for the classes:
    a. calculator

(border thickness) border: 1px
(border style) border-style: solid
(border color) border-color: cinzento
(roundy border line) border-radius: 5px;
(horizontal and vertical central positioning)

```
    position: absolute;
    top: 50%;
    left: 50%;
     transform: translate(-50%, -50%);
```

(wodh) width: 400px;

    b. display

(width) width: 100%;
(font size) font-size: 5rem;
(height) height: 80px;
(no border line) border: none;
(background color) background-color: #252525;
(color) color: #fff;
(texto alignment) text-align: right;
(padding to the right) padding-right: 20px;
(padding to the left) padding-left: 10px;

    c. operator

(color) color: #337cac;

    d. clear

(background color) background-color: #f0595f;

(border line color) border-color: #b0353a;

(color) color: #fff;

e. equals

(background color) background-color: #2e86c0;

(border line color) border-color: #337cac;

(color) color: #fff;

(height) height: 100%;

(grid areas used) grid-area: 2 / 4 / 6 / 5;

f. keys

(display in grid) display: grid;

(define how many columns are there in the grid) grid-template-columns: repeat(4, 1fr);

(define the gap between columns) grid-gap: 20px;

(define the padding) padding: 20px;

4. Define elemento styles:

a. button

height: 60px;

background-color: #fff;

border-radius: 3px;

border: 1px solid #c4c4c4;

background-color: transparent;

font-size: 2rem;

color: #333;

background-image: linear-gradient(to bottom,transparent,transparent 50%,rgba(0,0,0,.04));

box-shadow: inset 0 0 0 1px rgba(255,255,255,.05), inset 0 1px 0 0 rgba(255,255,255,.45), inset 0 -1px 0 0 rgba(255,255,255,.15), 0 1px 0 0 rgba(255,255,255,.15);

text-shadow: 0 1px rgba(255,255,255,.4);

5. Define behaviours (pseudoClass): hover over elements

a. Button (general)

background-color: #eaeaea;

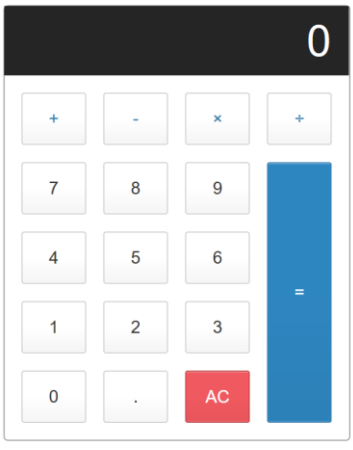
b. Clear button

background-color: #f17377;

c. Equals button

background-color: #4e9ed4;

Expected result



JavaScript

The calculator must allow performing basic arithmetic operations (e.g.: 10+13). Arithmetical operations are composed by 3 elements: the first operand (in the example, 10), the operator (in the example, +) and the second operand (in the example, 13). To use JavaScript to allow the calculator to perform the operations, follow the next steps:

1. Create a file named script.js.
2. Write <script src="script.js" type="text/javascript"></script> at the end of index.html body
3. Define the following object in JavaScript:

```
const calculator = {
  displayValue: '0',
  firstOperand: null,
  waitingForSecondOperand: false,
  operator: null,
};
```

4. Create the function that updates the value on the display based on the displayValue:

```
function updateDisplay() {
  const display = document.querySelector('.calculator-screen');
  display.value = calculator.displayValue;
}

updateDisplay();
```

5. Create the eventListener to capture the values of the pressed keys

```
let keys = document.querySelector('.keys');
keys.addEventListener('click', (event) => {
  let target = event.target;

  if (!target.matches('button')) {
    return;
  }

  if (target.classList.contains('operator')) {
    console.log('operator', target.value);
    return;
  }

  if (target.classList.contains('decimal')) {
    console.log('decimal', target.value);
    return;
  }

  if (target.classList.contains('limpar')) {
    console.log('key', target.value);
    return;
  }

  if (target.classList.contains('igual')) {
    console.log('key', target.value);
    return;
  }

  console.log('digit', target.value);
});
```

6. Verify on the console (Ctrl+Shift+I) that when the keys are pressed, the value is displayed

7. Create a function that allows the numbers pressed to appear on the display

```
function inputDigit(digit) {
 let displayValue = calculator.displayValue;
 // COMPLETE: if the current value is 0 replace by the digit. Otherwise, join the digit to the current value
 // …
}
```

8. Modify the eventListener (console.log('digit', target.value);) so that, when the numbers are pressed, instead of printing them to the console, it executes the function created on (7).
9. Force JavaScript to update the display.
10. Create a function that allows that pressing the decimal symbol changes the display (only if no decimal symbols have been inserted yet).

```
function inputDecimal(dot) {
 // Se o displayValue não contem nenhum sinal de decimal
 if (!calculator.displayValue.includes(dot)) {
  // COMPLETE: Join the signal to the content
  // …
 }
}
```

11. Change the eventListener and force JavaScript to update the display.
12. Create function (resetCalculator()) for the AC key.
13. Change the eventListener so that, when AC is pressed, the function implemented on (12) is called and the display is updated.
14. Create the function to handle pressing the operators. Change the eventListener so that, when the operators are pressed, the function implemented is called

```
function handleOperator(operator) {
  let firstOperand = calculator.firstOperand;
  let displayValue = calculator.displayValue;
  let operator = calculator.operator;

  let inputValue = parseFloat(displayValue);

  if (firstOperand === null) {
    calculator.firstOperand = inputValue;
  }

  calculator.waitingForSecondOperand = true;
  calculator.operator = operator;
}
```

15. At this moment, if we press 12+10, the display is not cleared after the '+'. The following is needed.

```
function inputDigit(digit) {
  let displayValue = calculator.displayValue;
  let waitingForSecondOperand = calculator.waitingForSecondOperand

  if (waitingForSecondOperand === true) {
    calculator.displayValue = digit;
    calculator.waitingForSecondOperand = false;
  } else {
        // Existing code
  }
}
```

16. Create a special object that allows the operation to be performed.

```
let fazerCalculo = {
  '+': (firstOperand, secondOperand) => firstOperand + secondOperand,
  // complete for -, * and /
  '=': (firstOperand, secondOperand) => secondOperand
};
```

17. When the user presses "=", the calculator must perform the operations and show the result. We can consider "=" as na operator. Update thefunction handleOperator(operator):

```
function handleOperator(operator) {
  let firstOperand = calculator.firstOperand;
  let displayValue = calculator.displayValue;
  let operator = calculator.operator;

  let inputValue = parseFloat(displayValue);

  if (firstOperand === null) {
    calculator.firstOperand = inputValue;
  } else if (operator) { // Se o operator já tiver sido definido anteriormente
    const result = fazerCalculo[operator](firstOperand, inputValue);
    // COMPLETE: force the result to be attributed to the displayValue
    // COMPLETE: update the display
    calculator.firstOperand = result; // faz com que os resultados possam ser acumulados
  }

  calculator.waitingForSecondOperand = true;
  calculator.operator = operator;
}
```