HTML, CSS, JavaScript

HTML

1. Criar um ficheiro index.html com os elementos necessários para uma calculadora:

```
Elemento: div
Classe: “calculadora”
```

a. Uma área de texto para mostrar o resultado

```
Elemento: input
Tipo: texto
Classe: "visor"
Valor: 0 (para definir o valor inicial do ecrã)
Disabled (definir que o utilizador não pode introduzir texto)
```

b. Teclas da calculadora

```
Elemento: div
Classe: “teclas”
```

i. Sinais de operações simples: soma (+), subtração (-), multiplicação (x) e divisão (÷)

```
Elemento: button
Tipo: button
Classe: "operador"
Valor: + - x / (conforme a operação em questão)
Conteúdo: + - &times; &divide; (conforme a operação em questão)
```

ii. Números: 0 a 9

```
Elemento: button
Tipo: button
Valor: 0 … 9 (conforme o número em questão)
Conteúdo: 0 … 9 (conforme o número em questão)
```

c. Sinal de decimal (.)

```
Elemento: button
Tipo: button
Classe: "decimal"
Valor: .
Conteúdo: .
```

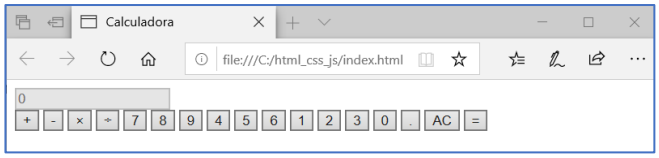
d. Limpar (AC)

```
Elemento: button
Tipo: button
Classe: "limpar"
Valor: limpar
Conteúdo: AC
```

e. Sinal igual (=)

```
Elemento: button
Tipo: button
Classe: "igual"
Valor: =
Conteúdo: =
```

Resultado esperado



CSS

1. Criar um ficheiro estilo.css que, inicialmente, deverá ter definidos os seguintes estilos:

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

2. Colocar a linha <link rel="stylesheet" href="estilo.css"> no head do ficheiro index.html.

3. Definir os estilos para as classes:

   a. calculadora

(espessura da linha de borda) border: 1px
(estilo da linha de borda) border-style: solid
(cor da linha de borda) border-color: cinzento
(linha de borda arredondada) border-radius: 5px;
(posicionamento absoluto a meio e ao centro)

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

(largura) width: 400px;

   b. visor

(largura) width: 100%;
(tamanho da letra) font-size: 5rem;
(altura) height: 80px;
(sem linha de fronteira) border: none;
(cor de fundo) background-color: #252525;
(cor) color: #fff;
(alinhamento do texto) text-align: right;
(intervalo de espaço à direita) padding-right: 20px;
(intervalo de espaço à esquerda) padding-left: 10px;

   c. operador

(cor) color: #337cac;

   d. limpar

(cor de fundo) background-color: #f0595f;
(cor da linha de fronteira) border-color: #b0353a;
(cor) color: #fff;

   e. igual

(cor de fundo) background-color: #2e86c0;
(cor da linha de fronteira) border-color: #337cac;
(cor) color: #fff;
(altura) height: 100%;
(zonas da grid que ocupa) grid-area: 2 / 4 / 6 / 5;

f. teclas

```
(definir que é para disport em grid) display: grid;
(definir quantas colunas temm a grid) grid-template-columns: repeat(4, 1fr);
(definir o interval entre as colunas) grid-gap: 20px;
(definir o espaçamento) padding: 20px;
```

4. Definir estilos de elementos:

a. botão

```
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. Definir comportamentos (pseudoclasses) de passar por cima (hover) dos elementos

a. Botão geral

```
background-color: #eaeaea;
```

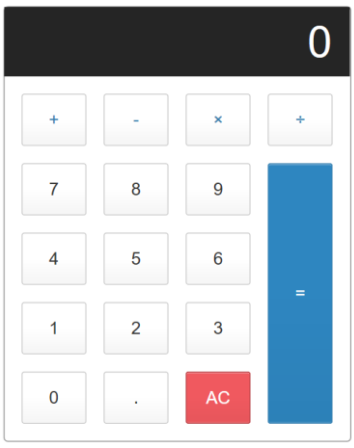
b. Botão de limpar

```
background-color: #f17377;
```

c. Botão igual

```
background-color: #4e9ed4;
```

Resultado esperado



JavaScript

A calculadora deve permitir fazer as operações aritméticas básicas como, por exemplo, 10+13. As operações aritméticas têm três componentes: o primeiro operando (no exemplo 10), o operador (no exemplo +) e o segundo operando. Para utilizar JavaScript para permitir à calculadora fazer os cálculos, seguir os seguintes passos:

1. Criar um ficheiro chamado script.js.

2. Colocar a linha <script src="script.js" type="text/javascript"></script> no **final** do body do documento index.html.

3. Definir o seguinte objeto em javascript:

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

4. Criar a função que atualiza o valor existente no visor da calculadora com base no displayValue do objeto anterior:

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

updateDisplay();
```

5. Criar o eventListener necessário para ir obtendo os valores das teclas em que se clica

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

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

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

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

```
  return;
 }

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

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

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

6. Verificar na consola (Ctrl+Shift+I) que quando se clica nas teclas da calculadora aparece a tecla clicada

7. Criar a função para que os cliques em dígitos sejam refletidos no visor da calculadora

```
function inputDigit(digit) {
 let displayValue = calculator.displayValue;
 // COMPLETAR:
        // Se o valor atual de displayValue for '0' substituir pelo dígito.
        // Se não, concatenar o dígito ao valor atual
 // …
}
```

8. Alterar a linha correspondente no eventListener (console.log('digit', target.value);)para que, quando se clica num dos dígitos, em vez de mostrar o dígito para a consola, execute a função criada na alínea anterior.

9. Obrigar o JavaScript a atualizar o visor.

10. Criar uma função para que o clique no sinal de decimal seja refletido no visor da calculadora (mas só se ainda não tiver sido introduzido nenhum sinal decimal).

```
function inputDecimal(dot) {
 // Se o displayValue não contem nenhum sinal de decimal
 if (!calculator.displayValue.includes(dot)) {
  // COMPLETAR: Concatenar o sinal ao conteúdo do displayValue
  // …
 }
}
```

11. Alterar a linha correspondente no eventListener e mandar o JavaScript atualizar o visor.

12. Criar a função necessária (resetCalculator()) para a tecla AC (limpar), que deve colocar a calculadora no estado inicial.

13. Alterar o eventListener para que, quando se clica na tecla AC, seja chamada essa função e de seguida se faça update ao valor do visor.

14. Criar a função para tratar o clique num dos operadores (no final, alterar o eventListener para que, quando se clica num dos operadores, seja chamada a função)

```
function handleOperator(operador) {
  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 = operador;
}
```

15. Neste momento, se tentarmos introduzir 12+10, o visor não faz reset depois do sinal +. É necessário atualizar a função inputDigit(digit) para que isso aconteça.

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

  if (waitingForSecondOperand === true) {
    calculator.displayValue = digit;
    calculator.waitingForSecondOperand = false;
  } else {
        // Código já existente
  }
}
```

16. Criar um objeto especial que permite efetuar o cálculo pedido.

```
let fazerCalculo = {
  '+': (firstOperand, secondOperand) => firstOperand + secondOperand,
  // completar para -, * e /
  '=': (firstOperand, secondOperand) => secondOperand
};
```

17. Quando o utilizador clica no “=”, a calculadora deve fazer o cálculo e mostrar o resultado. Podemos considerar que o = é também um operador. Atualizar a função handleOperator(operador):

```
function handleOperator(operador) {
  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 operador já tiver sido definido anteriormente
    const result = fazerCalculo[operator](firstOperand, inputValue);
    // COMPLETAR: fazer com que o result seja atribuído ao displayValue
    // COMPLETAR: atualizar o visor
    calculator.firstOperand = result; // faz com que os resultados possam ser acumulados
  }

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