script.js

```
class Calculator {
    constructor(previousOperandTextElement, currentOperandTextElement) {
      this.previousOperandTextElement = previousOperandTextElement
      this.currentOperandTextElement = currentOperandTextElement
      this.clear()
    }

    clear() {
      this.currentOperand = ''
      this.previousOperand = ''
      this.operation = undefined
    }

    delete() {
      this.currentOperand = this.currentOperand.toString().slice(0, -1)
    }

    appendNumber(number) {
      if (number === '.' && this.currentOperand.includes('.')) return
      this.currentOperand = this.currentOperand.toString() + number.toString()
    }

    chooseOperation(operation) {
      if (this.currentOperand === '') return
      if (this.previousOperand !== '') {
        this.compute()
      }
      this.operation = operation
      this.previousOperand = this.currentOperand
      this.currentOperand = ''
    }

    compute() {
      let computation
      const prev = parseFloat(this.previousOperand)
      const current = parseFloat(this.currentOperand)
      if (isNaN(prev) || isNaN(current)) return
      switch (this.operation) {
        case '+':
          computation = prev + current
          break
        case '-':
          computation = prev - current
          break
        case '*':
          computation = prev * current
          break
        case '÷':
```

```
          computation = prev / current
          break
        default:
          return
      }
      this.currentOperand = computation
      this.operation = undefined
      this.previousOperand = ''
    }

    getDisplayNumber(number) {
      const stringNumber = number.toString()
      const integerDigits = parseFloat(stringNumber.split('.')[0])
      const decimalDigits = stringNumber.split('.')[1]
      let integerDisplay
      if (isNaN(integerDigits)) {
        integerDisplay = ''
      } else {
        integerDisplay = integerDigits.toLocaleString('en', { maximumFractionDigits: 0 })
      }
      if (decimalDigits != null) {
        return `${integerDisplay}.${decimalDigits}`
      } else {
        return integerDisplay
      }
    }

    updateDisplay() {
      this.currentOperandTextElement.innerText =
        this.getDisplayNumber(this.currentOperand)
      if (this.operation != null) {
        this.previousOperandTextElement.innerText =
          `${this.getDisplayNumber(this.previousOperand)} ${this.operation}`
      } else {
        this.previousOperandTextElement.innerText = ''
      }
    }
  }


  const numberButtons = document.querySelectorAll('[data-number]')
  const operationButtons = document.querySelectorAll('[data-operation]')
  const equalsButton = document.querySelector('[data-equals]')
  const deleteButton = document.querySelector('[data-delete]')
  const allClearButton = document.querySelector('[data-all-clear]')
  const previousOperandTextElement = document.querySelector('[data-previous-operand]')
  const currentOperandTextElement = document.querySelector('[data-current-operand]')

  const calculator = new Calculator(previousOperandTextElement, currentOperandTextElement)

```

```
  numberButtons.forEach(button => {
    button.addEventListener('click', () => {
      calculator.appendNumber(button.innerText)
      calculator.updateDisplay()
    })
  })

  operationButtons.forEach(button => {
    button.addEventListener('click', () => {
      calculator.chooseOperation(button.innerText)
      calculator.updateDisplay()
    })
  })

  equalsButton.addEventListener('click', button => {
    calculator.compute()
    calculator.updateDisplay()
  })

  allClearButton.addEventListener('click', button => {
    calculator.clear()
    calculator.updateDisplay()
  })

  deleteButton.addEventListener('click', button => {
    calculator.delete()
    calculator.updateDisplay()
  })
```