Parte 1: Verificación de que todas hagan lo mismo

Cada una de las calculadoras realiza las mismas operaciones (suma, multiplicación y encontrar el número mayor) y tiene la misma interfaz con dos entradas de números y un botón para realizar el cálculo. Los resultados de las operaciones se muestran en el mismo formato.

- **Calculadora Clásica**: Usa una función tradicional en JavaScript para realizar las operaciones y actualizar los resultados en el DOM con `getElementById`.
- **Calculadora Moderna**: Utiliza un enfoque más moderno con funciones de flecha y el uso de `addEventListener` para manejar el evento del botón.
- **Calculadora Funcional**: Se enfoca en un estilo más funcional con funciones puras para obtener el número, establecer el texto y realizar las operaciones.
- **Calculadora con Clases**: Implementa una clase `Calculadora` para estructurar el código y manejar las operaciones.

En resumen, todas las calculadoras realizan las mismas funciones de suma, multiplicación y encontrar el mayor número, aunque usan diferentes enfoques de programación.

Parte 2: Análisis del código con IA

A continuación se presenta el código comentado de cada una de las versiones de la calculadora.

Calculadora Clásica

```
// Esta función se activa cuando se hace clic en el botón 'Calcular'
function calcular() {
    // Se obtienen los valores de los números ingresados
    const numero1 = Number(document.getElementById("numero1").value);
    const numero2 = Number(document.getElementById("numero2").value);

    // Se calculan la suma y el producto
    const suma = numero1 + numero2;
    const producto = numero1 * numero2;
    let mayor;

    // Se determina cuál es el número mayor
    if (numero1 > numero2) {
        mayor = numero1;
    } else if (numero2 > numero1) {
        mayor = numero2;
    } else {
        mayor = "Los números son iguales";
    }

    // Se muestran los resultados en el HTML
    document.getElementById("suma").textContent = suma;
    document.getElementById("producto").textContent = producto;
    document.getElementById("mayor").textContent = mayor;
}
```

Calculadora Moderna

```
// Se agrega un event listener al botón para ejecutar el cálculo cuando se hace clic
document.getElementById("calcularBtn").addEventListener("click", () => {
    // Se obtienen los números ingresados como valores numéricos
    const numero1 = Number(document.getElementById("numero1").value);
    const numero2 = Number(document.getElementById("numero2").value);

    // Se calculan la suma, el producto y el número mayor
    const suma = numero1 + numero2;
    const producto = numero1 * numero2;
    const mayor = numero1 === numero2 ? "Los números son iguales" :
Math.max(numero1, numero2);

    // Se muestran los resultados en el HTML
    document.getElementById("suma").textContent = suma;
    document.getElementById("producto").textContent = producto;
    document.getElementById("mayor").textContent = mayor;
});
```

Calculadora Funcional

```
// Función para obtener el valor numérico de un input por su id
const getNumber = (id) => Number(document.getElementById(id).value);

// Función para establecer el texto de un elemento por su id
const setText = (id, value) => document.getElementById(id).textContent = value;

// Función principal para realizar los cálculos
const calcular = () => {
    // Se obtienen los dos números ingresados
    const [numero1, numero2] = [getNumber("numero1"), getNumber("numero2")];

    // Se definen las operaciones de suma, producto y mayor usando funciones puras
    const operaciones = {
        suma: (a, b) => a + b,
        producto: (a, b) => a * b,
        mayor: (a, b) => (a === b ? "Los números son iguales" : Math.max(a, b))
    };

    // Se establecen los resultados en el HTML
    setText("suma", operaciones.suma(numero1, numero2));
    setText("producto", operaciones.producto(numero1, numero2));
    setText("mayor", operaciones.mayor(numero1, numero2));
};

// Se asocia la función calcular al botón de clic
document.getElementById("calcularBtn").addEventListener("click", calcular);
```

Calculadora con Clases

```
// Definición de la clase Calculadora
class Calculadora {
    // Constructor que recibe los ids de los elementos HTML
    constructor(numero1Id, numero2Id, sumaId, productoId, mayorId) {
        this.numero1Input = document.getElementById(numero1Id);
        this.numero2Input = document.getElementById(numero2Id);
        this.sumaOutput = document.getElementById(sumaId);
        this.productoOutput = document.getElementById(productoId);
        this.mayorOutput = document.getElementById(mayorId);
    }

    // Método para obtener el número ingresado
    obtenerNumero(input) {
        return Number(input.value);
    }

    // Método principal que ejecuta los cálculos
    calcular() {
        const numero1 = this.obtenerNumero(this.numero1Input);
        const numero2 = this.obtenerNumero(this.numero2Input);

        // Se llaman a los métodos para mostrar los resultados
        this.mostrarResultados(numero1, numero2);
    }

    // Método para mostrar los resultados de las operaciones
    mostrarResultados(numero1, numero2) {
        this.sumaOutput.textContent = numero1 + numero2;
        this.productoOutput.textContent = numero1 * numero2;
        this.mayorOutput.textContent = numero1 === numero2 ? "Los números son
iguales" : Math.max(numero1, numero2);
    }
```

```
}

// Se crea una instancia de la clase y se asocia el evento al botón
const calculadora = new Calculadora("numero1", "numero2", "suma", "producto",
"mayor");
document.getElementById("calcularBtn").addEventListener("click", () =>
calculadora.calcular());
```

Comparativa de las Versiones

Aquí se presenta una tabla comparativa entre las diferentes versiones de la calculadora en términos de facilidad de entender, limpieza del código y facilidad de mantenimiento/expansión.

Característica	Calculadora Clásica	Calculadora Moderna	Calculadora Funcional	Calculadora con Clases
Facilidad de entender	Moderada	Alta	Alta	Alta
Código limpio	Moderado	Alto	Alto	Medio
Mantenimiento/Futuro	Moderado	Alto	Alto	Alto

Sugerencia de Otros Enfoques

La IA sugiere que se podrían explorar más enfoques de implementación como:

- **Programación reactiva** usando bibliotecas como React.js, que permite un manejo más eficiente de las interfaces de usuario.

- **Uso de frameworks** como Vue.js o Angular para estructurar más fácilmente el código.

- **Uso de Web Workers** para hacer los cálculos en segundo plano si se tiene una gran cantidad de operaciones.

Parte 3: Conclusiones

1. ¿Qué estilo te parece más fácil y más difícil?

Más fácil: La Calculadora Moderna es, en mi opinión, la más fácil de entender. Utiliza un enfoque simple con funciones flecha, addEventListener, y un manejo directo de los elementos del DOM, lo que la hace muy clara para alguien que está comenzando a trabajar con JavaScript. Este estilo es adecuado para quien busca una programación estructurada pero simple, ideal para proyectos pequeños y rápidos.

Más difícil: La Calculadora con Clases es probablemente la más compleja de entender. Aunque las clases son una característica muy poderosa en la programación orientada a objetos, el hecho de tener que manejar objetos y métodos adicionales puede dificultar la comprensión al principio, sobre todo si no se tiene mucha experiencia con el concepto de clases en JavaScript.

2. ¿Es todo JavaScript? ¿Por qué tantas formas?

Sí, todas las versiones están escritas en JavaScript, pero cada una utiliza diferentes enfoques de programación. Las diversas formas existen porque JavaScript permite una gran flexibilidad, lo que da lugar a múltiples maneras de abordar un mismo problema. Por ejemplo:

La Calculadora Clásica utiliza funciones globales, lo cual es común en JavaScript tradicional. La Calculadora Moderna utiliza funciones de flecha y eventos para un código más limpio y modular. La Calculadora Funcional enfatiza el uso de funciones puras, lo que promueve un estilo más matemático y menos mutable. La Calculadora con Clases usa la programación orientada a objetos, lo que es útil para estructurar código más grande y reutilizable. Cada estilo tiene sus ventajas y se adapta a diferentes necesidades y niveles de experiencia.

3. ¿Qué versión de la calculadora es más adecuada para nosotros?

Para proyectos pequeños y simples, como una calculadora básica en una web, la Calculadora Moderna es la más adecuada. Es directa, fácil de entender, y eficiente. Permite realizar los cálculos sin complejidades adicionales, lo cual es perfecto para tareas simples.

Si se busca entender la programación orientada a objetos y se desea desarrollar aplicaciones más complejas, la Calculadora con Clases es más adecuada. Te permite aprender sobre clases, instancias y métodos, lo que es esencial cuando trabajas con aplicaciones más grandes y estructuradas.

4. ¿Qué versión de la calculadora es más adecuada para aprender?

La Calculadora Moderna es la mejor para aprender. Introduce conceptos modernos de JavaScript como funciones de flecha, addEventListener, y manejo del DOM de una manera clara. Además, es fácil de modificar y expandir, lo que permite experimentar mientras se aprende.

5. ¿Qué ventajas y desventajas encuentras en cada enfoque?

Calculadora Clásica:

Ventajas: Código simple, fácil de entender para principiantes. Utiliza funciones estándar de JavaScript sin depender de características más avanzadas.

Desventajas: Puede volverse desordenado y difícil de mantener a medida que crecen los proyectos, ya que las funciones globales pueden causar conflictos en el código.

Calculadora Moderna:

Ventajas: Código limpio, modular y con una buena separación de responsabilidades. Utiliza enfoques modernos que facilitan la reutilización y mantenimiento.

Desventajas: Aún puede ser difícil para un principiante entender completamente el uso de addEventListener y las funciones flecha.

Calculadora Funcional:

Ventajas: Enfoque funcional que promueve un código limpio y predecible. Funciones puras que no alteran el estado global.

Desventajas: Puede ser más difícil de entender para alguien que no está familiarizado con la programación funcional. Es más abstracto.

Calculadora con Clases:

Ventajas: Utiliza la programación orientada a objetos, lo que es útil para aplicaciones más grandes. Permite estructurar mejor el código y facilita la reutilización.

Desventajas: Requiere más conocimiento sobre clases y objetos, lo que puede ser un desafío para quienes están comenzando a aprender JavaScript.

6. ¿Qué versión o propuesta te parece más interesante?

La Calculadora Moderna es la más interesante para alguien que busca una forma rápida y eficiente de implementar una calculadora con un enfoque en la simplicidad. Combina lo mejor de la programación estructurada con un estilo moderno, sin complicaciones innecesarias. Si se busca una experiencia más

profunda y orientada a objetos, la Calculadora con Clases es igualmente interesante, ya que permite entender un paradigma de programación más complejo y preparado para proyectos más grandes.

En resumen, la elección de la versión depende del nivel de experiencia y los objetivos del aprendizaje o proyecto. Para un principiante, la Calculadora Moderna es ideal, mientras que para quienes buscan profundizar en conceptos más avanzados, la Calculadora con Clases es una excelente opción.