🧪 EJERCICIO 1: Cálculo de Aceleración desde Fuerzas (matriz de fuerzas + masa por objeto)

Descripción:

Construye una aplicación en JavaScript que:

- Permita al usuario **agregar múltiples objetos**.
- Cada objeto tendrá:
    - Un vector de fuerza F = [Fx, Fy, Fz]
    - Una masa m

Objetivo del ejercicio:

- Generar una **matriz de fuerzas**.
- Calcular una **matriz de aceleraciones** aplicando la **2ª Ley de Newton**:

$$a = \frac{F}{m}$$

Qué debes programar:

- Un formulario para ingresar Fx, Fy, Fz y m.
- Un botón "Agregar objeto" que actualice la matriz.
- Un botón "Calcular" que genere y muestre la matriz de aceleraciones.

```
// Datos iniciales
const fuerzas = [
  [10, 5, 0],    // Fuerza sobre el objeto 1
  [4, -3, 2],    // Fuerza sobre el objeto 2
  [0, 0, 6]      // Fuerza sobre el objeto 3
];

const masas = [2, 4, 3]; // Masa de cada objeto

// Instrucciones:
// 1. Recorre la matriz de fuerzas.
// 2. Para cada vector de fuerza y su masa correspondiente, calcula el vector de aceleración (F = m * a → a = F / m).
// 3. Almacena los resultados en una nueva matriz de aceleraciones.
```

🧪 EJERCICIO 2: Suma de fuerzas vectoriales y cálculo de aceleración total

Descripción:

Crea una app donde el usuario pueda:

- Agregar **múltiples fuerzas** aplicadas sobre un mismo objeto (como una matriz de vectores de fuerza).
- Ingresar la **masa** del objeto.

Objetivo del ejercicio:

- Sumar los vectores de fuerza:

  $$F_{\text{total}} = \sum \vec{F}_i$$

- Calcular la aceleración total:

  $$a = \frac{F_{\text{total}}}{m}$$

  Qué debes codificar:

- Inputs para ingresar fuerzas como [Fx, Fy] o [Fx, Fy, Fz].
- Una lista (matriz) de fuerzas que se puede ir llenando.
- Un campo para masa.
- Un botón "Calcular" que:
  - Sume todas las fuerzas.
  - Calcule la aceleración resultante.
  - Muestre los resultados.

```
// Datos iniciales: fuerzas aplicadas sobre el mismo objeto
const fuerzas = [
  [5, 0],     // Fuerza 1
  [-2, 3],    // Fuerza 2
  [1, -1]     // Fuerza 3
];

const masa = 2; // kg

// Instrucciones:
// 1. Suma todas las fuerzas para obtener la fuerza neta total.
// 2. Usa F = m * a para calcular la aceleración vectorial del objeto.
// 3. Imprime la fuerza total y la aceleración.
```

🧪 **EJERCICIO 3: Simulación de posición usando matrices y aceleración constante**

Descripción:

Simula el movimiento de un objeto con una aceleración constante, durante varios segundos.

Qué puedes permitir al usuario:

- Ingresar una **aceleración vectorial** [ax, ay] (o 3D).
- Ingresar una **velocidad inicial** [vx, vy].
- Ingresar una **posición inicial** [x0, y0].
- Definir el número de pasos de tiempo t (por ejemplo: 5 segundos).

Objetivo del ejercicio:

- Generar una **matriz de posiciones por tiempo**, usando la fórmula:

$$x = x_0 + v_0 \cdot t + \frac{1}{2} a \cdot t^2$$

- Mostrar una tabla con las posiciones en cada segundo.

```
// Datos iniciales
const aceleracion = [2, -1];    // m/s²
const velocidadInicial = [4, 0]; // m/s
const posicionInicial = [0, 0];  // m
const pasos = 5; // calcular posición para t = 1 a 5

// Instrucciones:
// 1. Usar la fórmula de movimiento rectilíneo uniformemente acelerado:
//    x = x0 + v0*t + 0.5*a*t²
// 2. Generar un array o matriz con la posición del objeto en cada segundo.
// 3. Imprimir la posición del objeto en cada tiempo t = 1, 2, 3, ..., 5.
```

💡 Consejo para empezar

Puedes usar arrays como estructuras de datos para vectores y matrices, por ejemplo:

javascript Copiar Editar

```javascript
let fuerzas = []; // matriz de vectores de fuerza
let masas = [];   // array de masas
```

Y usar `Array.map()` o `reduce()` para hacer cálculos vectoriales. Ejemplo:

javascript Copiar Editar

```javascript
const sumaFuerza = fuerzas.reduce((acc, fuerza) =>
  acc.map((val, i) => val + fuerza[i]), [0, 0, 0]);
```