Problema I (15 puntos)

Cambiar el contexto de `this`

Escribe una función llamada `saludar` que tome un parámetro `nombre` y devuelva un saludo personalizado. Luego, crea un objeto `persona` con una propiedad `nombre`. Utiliza el método `call()` para llamar a la función `saludar` con el contexto del objeto `persona`. Si lo deseas apoyate en la siguiente estructura:

Problema1.js > ...

```
1  function saludar() {
2    return `¡Hola, ${this.nombre}! ¿Cómo estás?`;
3  }
4  let persona = {
5    nombre: "Juan",
6  };
7  let mensajeSaludo = saludar.call(persona);
8
9  console.log(mensajeSaludo);
10
```

```
> function saludar() {
    return `¡Hola, ${this.nombre}! ¿Cómo estás?`;
  }
  let persona = {
    nombre: "Juan",
  };
  let mensajeSaludo = saludar.call(persona);

  console.log(mensajeSaludo);
¡Hola, Juan! ¿Cómo estás?
```

```
> mensajeSaludo
< '¡Hola, Juan! ¿Cómo estás?'
```

Problema II (15 puntos)
Cambiar el contexto de `this` en un método de objeto

Crea un objeto `auto` con una propiedad `marca` y un método `mostrarMarca` que devuelva un mensaje con la marca del auto. Luego, crea otro objeto `moto` con una propiedad `marca`. Utiliza el método `call()` para llamar al método `mostrarMarca` del objeto `auto` con el contexto del objeto `moto`.

JS Problemall.js > ...

```
let auto = {
  marca: "Toyota",
  mostrarMarca: function () {
    return `La marca del auto es ${this.marca}`;
  },
};

let moto = {
  marca: "Honda",
};

let mensajeMarcaMoto = auto.mostrarMarca.call(moto);

console.log(mensajeMarcaMoto);
```

```
> let auto = {
    marca: "Toyota",
    mostrarMarca: function () {
      return `La marca del auto es ${this.marca}`;
    },
  };

  let moto = {
    marca: "Honda",
  };

  let mensajeMarcaMoto = auto.mostrarMarca.call(moto);

  console.log(mensajeMarcaMoto);
  La marca del auto es Honda                                    VM202:14
< undefined
⚠ Error with Permissions-Policy header: Unrecognized feature: 'ch-ua-form-
  factor'.
> mensajeMarcaMoto
< 'La marca del auto es Honda'
```

Problema III (10 Puntos)

Crea dos objetos, `persona1` y `persona2`, cada uno con una propiedad `nombre`. Define una función `saludar` que devuelva un mensaje de saludo utilizando la propiedad `nombre`. Usa `call()` para invocar la función `saludar` con el contexto de `persona2`.

```
JS ProblemaIII.js > ...
1   function saludar() {
2     return `¡Hola, ${this.nombre}! ¿Cómo estás?`;
3   }
4
5   let persona1 = {
6     nombre: "Juan",
7   };
8
9   let persona2 = {
10    nombre: "María",
11  };
12
13  let mensajeSaludo1 = saludar.call(persona1);
14  let mensajeSaludo2 = saludar.call(persona2);
15
16  console.log(mensajeSaludo1);
17  console.log(mensajeSaludo2);
```

```
> function saludar() {
    return `¡Hola, ${this.nombre}! ¿Cómo estás?`;
  }

  let persona1 = {
    nombre: "Juan",
  };

  let persona2 = {
    nombre: "María",
  };

  let mensajeSaludo1 = saludar.call(persona1);
  let mensajeSaludo2 = saludar.call(persona2);

  console.log(mensajeSaludo1);
  console.log(mensajeSaludo2);
  ¡Hola, Juan! ¿Cómo estás?                                    VM774:16
  ¡Hola, María! ¿Cómo estás?                                   VM774:17
< undefined
⚠ Error with Permissions-Policy header: Unrecognized feature: 'ch-ua-form-
  factor'.
> mensajeSaludo1
< '¡Hola, Juan! ¿Cómo estás?'
> mensajeSaludo2
< '¡Hola, María! ¿Cómo estás?'
```

Problema IV (10 Puntos)
Crea un objeto `rectangulo` con propiedades `ancho` y `alto`, y un método `area` que calcule el área del rectángulo. Luego, crea un objeto `cuadrado` con una propiedad `lado`. Usa `call()` para invocar el método `area` del objeto `rectangulo` con el contexto del objeto `cuadrado`.

JS ProblemaIV.js > ...

```
let rectangulo = {
  ancho: 5,
  alto: 10,
  area: function () {
    return this.ancho * this.alto;
  },
};

const cuadrado = {
  lado: 7,
};

const areaCuadrado = rectangulo.area.call({
  ancho: cuadrado.lado,
  alto: cuadrado.lado,
});


console.log("El área del cuadrado es:", areaCuadrado);
```

```
> let rectangulo = {
    ancho: 5,
    alto: 10,
    area: function () {
      return this.ancho * this.alto;
    },
  };

  const cuadrado = {
    lado: 7,
  };

  const areaCuadrado = rectangulo.area.call({
    ancho: cuadrado.lado,
    alto: cuadrado.lado,
  });


  console.log("El área del cuadrado es:", areaCuadrado);
  El área del cuadrado es: 49
```

Problema V (15 puntos)

Crea dos objetos, `persona1` y `persona2`, cada uno con una propiedad `nombre`. Define una función `presentar` que devuelva un mensaje de presentación utilizando la propiedad `nombre`. Usa `apply()` para invocar la función `presentar` con el contexto de `persona2`.

JS ProblemaV.js > ...

```
function presentar() {
  return `Hola, soy ${this.nombre}. Mucho gusto.`;
}

let persona1 = {
  nombre: "Juan",
};

let persona2 = {
  nombre: "María",
};

let mensajePresentacion = presentar.apply(persona2);

console.log(mensajePresentacion);
```

```
▶ 704 Third-party cookie will be blocked in future Chrome versions as part of Privacy Sandbox.

> function presentar() {
    return `Hola, soy ${this.nombre}. Mucho gusto.`;
  }

  let persona1 = {
    nombre: "Juan",
  };

  let persona2 = {
    nombre: "María",
  };

  let mensajePresentacion = presentar.apply(persona2);

  console.log(mensajePresentacion);

  Hola, soy María. Mucho gusto.                                VM1243:15

< undefined

⚠ Error with Permissions-Policy header: Unrecognized feature: 'ch-ua-form-factor'.

> mensajePresentacion

< 'Hola, soy María. Mucho gusto.'
```

Problema VI (15 puntos)

Expansión de Objeto con Array Crea un objeto `libro` con propiedades `titulo` y `autor`. Define una función `agregarCapitulos` que tome un array de capítulos y los agregue como una nueva propiedad `capitulos` al objeto `libro`. Usa `apply()` para invocar la función `agregarCapitulos` con el contexto del objeto `libro` y un array de capítulos.


```
> let libro = {
      titulo: 'Harry Potter',
      autor: 'J. K. Rowling'
  };

  function agregarCapitulos(capitulos) {
      this.capitulos = capitulos;
  }

  let listaCapitulos = ['Harry Potter y la piedra filosofal', 'Harry Potter y
  la cámara secreta', 'Harry Potter y el prisionero de azkaban', 'Harry
  Potter y el cáliz de fuego'];

  agregarCapitulos.apply(libro, [listaCapitulos]);

  console.log(libro);
                                                                  VM1319:14
  ▶ {titulo: 'Harry Potter', autor: 'J. K. Rowling', capitulos: Array(4)}
< undefined
⚠ Error with Permissions-Policy header: Unrecognized feature: 'ch-ua-form-
  factor'.
> agregarCapitulos
< ƒ agregarCapitulos(capitulos) {
      this.capitulos = capitulos;
  }
> listaCapitulos
<   (4) ['Harry Potter y la piedra filosofal', 'Harry Potter y la cámara secr
  ▶ eta', 'Harry Potter y el prisionero de azkaban', 'Harry Potter y el cáliz
    de fuego']
```

Problema VII (20 puntos)

Actualización de Saldo en Cuenta Bancaria

Crea un objeto `cuentaBancaria` con propiedades `titular` y `saldo`. Define una función `actualizarSaldo` que tome un monto y lo sume al saldo actual de la cuenta. Usa `apply()` para invocar la función `actualizarSaldo` con el contexto del objeto `cuentaBancaria` y un array que contenga el monto a agregar.

ProblemaVII.js > ...

```
let cuentaBancaria = {
  titular: "Eduardo Colindres",
  saldo: 1000,
};

function actualizarSaldo(monto) {
  this.saldo += monto;
}

let montoAgregar = 500;

actualizarSaldo.apply(cuentaBancaria, [montoAgregar]);

console.log(cuentaBancaria);
```

```
> let cuentaBancaria = {
    titular: "Eduardo Colindres",
    saldo: 1000,
  };

  function actualizarSaldo(monto) {
    this.saldo += monto;
  }

  let montoAgregar = 500;

  actualizarSaldo.apply(cuentaBancaria, [montoAgregar]);

  console.log(cuentaBancaria);
  ▶ {titular: 'Eduardo Colindres', saldo: 1500}          VM1705:14
< undefined
```