12-11-2023

Estructuras definidas por el usuario en JavaScript

Tarea 04

ROBERTO RODRÍGUEZ JIMÉNEZ
roberto.rodjim.1@educa.jcyl.es

Contenido

Tarea online DWEC02

Título de la tarea: Tarea online para DWEC04
Unidad: DWEC04
Ciclo formativo y módulo: Desarrollo de Aplicaciones Web - Desarrollo web en entorno cliente
Curso académico: 2020/2021

¿Qué contenidos o resultados de aprendizaje trabajaremos?

Resultados de Aprendizaje

- ✓ RA4. Programa código para clientes Web analizando y utilizando estructuras definidas por el usuario.

Contenidos

Estructuras definidas por el usuario en JavaScript.

1. Arrays.
    1. Trabajando con Arrays.
2. Funciones.
    1. Trabajando con funciones.
3. Objetos.
    1. Trabajando con objetos.

1.- Descripción de la tarea

Caso práctico

Antonio ha avanzado mucho en su estudio de JavaScript, viendo los principales objetos con los que puede trabajar, sus métodos y propiedades.

Ha llegado el momento, de ver cómo puede organizar los datos que genera en su aplicación de JavaScript, de forma que le permita gestionarlos más eficientemente.

También verá cómo crear funciones, el alcance de las variables y cómo utilizarlas en su código. Y para terminar Juan, le va a explicar un poco más en detalle, cómo podrá crear nuevos objetos y asignarles propiedades y métodos en JavaScript.

Antonio ha comenzado a analizar más a fondo parte del trabajo que tiene que realizar, y comenta con su directora Ada, la posibilidad de hacer algunos bocetos de las innovaciones y mejoras que quiere aportar al proyecto. Ada está muy contenta con los progresos de Antonio y está deseando ver los bocetos.

¿Qué te pedimos que hagas?

Queremos hacer una aplicación en JavaScript para gestionar edificios con la información de la situación del edificio y de los propietarios de cada piso. Para ello queremos almacenar la siguiente información de cada edificio:

- calle.
- número.
- código postal.
- plantas del edificio (dentro de cada planta tendremos un número de puertas y para cada puerta almacenaremos el nombre del propietario).

Se pide:

- Crear un objeto que nos permita instanciar edificios. Cada vez que instanciemos un edificio le pasaremos la calle, número y código postal como parámetros. Se pide además crear los siguientes métodos para el objeto Edificio:
    - agregarPlantasYPuertas(numplantas, puertas) // Se le pasa el número de plantas que queremos crear en el piso y el número de puertas por planta. Cada vez que se llame a este método, añadirá el número de plantas y puertas indicadas en los parámetros, a las que ya están creadas en el edificio.
    - modificarNumero(numero) // Se le pasa el nuevo número del edificio para que lo actualice.
    - modificarCalle(calle) // Se le pasa el nuevo nombre de la calle para que lo actualice.
    - modificarCodigoPostal(codigo) // Se le pasa el nuevo número de código postal del edificio.
    - imprimeCalle // Devuelve el nombre de la calle del edificio.
    - imprimeNumero // Devuelve el número del edificio.
    - imprimeCodigoPostal // Devuelve el código postal del edificio.
    - agregarPropietario(nombre,planta,puerta) // Se le pasa un nombre de propietario, un número de planta y un número de puerta y lo asignará como propietario de ese piso.
    - imprimePlantas // Recorrerá el edificio e imprimirá todos los propietarios de cada puerta.
- Cada vez que se crea un edificio que muestre automáticamente un mensaje del estilo:

    - construido nuevo edificio en calle: xxxxxx, nº: xx, CP: xxxxx.
- Cada vez que se añada un propietario a un piso de un edificio que muestre un mensaje del estilo:
    - xxxxxxxx es ahora el propietario de la puerta x de la planta x.

Aquí se muestra un **ejemplo de lo que tendría que mostrar la aplicación**:

Trabajando con objetos Javascript:

Instanciamos 3 objetos edificioA, edificioB y edificioC con estos datos:

- Construido nuevo edificio en calle: Garcia Prieto, nº: 58, CP: 15706.
- Construido nuevo edificio en calle: Camino Caneiro, nº: 29, CP: 32004.
- Construido nuevo edificio en calle: San Clemente, nº: s/n, CP: 15705.

- El código postal del edificio A es: 15706.
- La calle del edificio C es: San Clemente.
- El edificio B está situado en la calle Camino Caneiro número 29.

Agregamos 2 plantas y 3 puertas por planta al edificio A...

Agregamos 4 propietarios al edificio A...

- Jose Antonio Lopez es ahora el propietario de la puerta 1 de la planta 1.
- Luisa Martinez es ahora el propietario de la puerta 2 de la planta 1.
- Marta Castellón es ahora el propietario de la puerta 3 de la planta 1.
- Antonio Pereira es ahora el propietario de la puerta 2 de la planta 2.

Listado de propietarios del edificio calle García Prieto número 58

- Propietario del piso 1 de la planta 1: Jose Antonio Lopez.
- Propietario del piso 2 de la planta 1: Luisa Martinez.
- Propietario del piso 3 de la planta 1: Marta Castellón.
- Propietario del piso 1 de la planta 2:
- Propietario del piso 2 de la planta 2: Antonio Pereira.
- Propietario del piso 3 de la planta 2:

Agregamos 1 planta más al edificio A...

Agregamos 1 propietario más al edificio A planta 3, puerta 2...

- Pedro Meijide es ahora el propietario de la puerta 2 de la planta 3.

Listado de propietarios del edificio calle García Prieto número 58

- Propietario del piso 1 de la planta 1: Jose Antonio Lopez.
- Propietario del piso 2 de la planta 1: Luisa Martinez.
- Propietario del piso 3 de la planta 1: Marta Castellón.
- Propietario del piso 1 de la planta 2:
- Propietario del piso 2 de la planta 2:
- Propietario del piso 1 de la planta 3:
- Propietario del piso 2 de la planta 3: Pedro Meijide.

2.- Información de interés

Recursos necesarios

Editor web para teclear el código de la aplicación y un navegador web para ejecutar y probar la aplicación.

Consejos y recomendaciones

Intenta crear todos los métodos dentro del objeto Edificio y utiliza los métodos programados para imprimir el valor de las propiedades que se piden en el enunciado.

3.- Evaluación de la tarea

Criterios de evaluación implicados

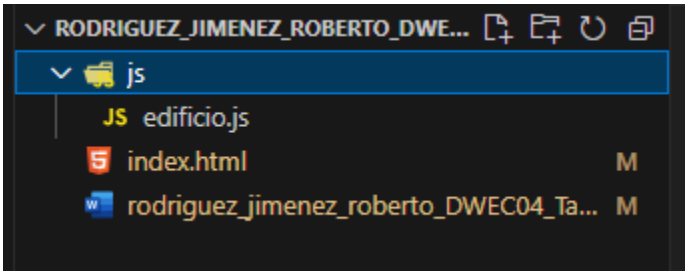
- a) Se han clasificado y utilizado las funciones predefinidas del lenguaje.
- b) Se han creado y utilizado funciones definidas por el usuario.
- c) Se han reconocido las características del lenguaje relativas a la creación y uso de arrays.
- d) Se han creado y utilizado arrays.
- e) Se han reconocido las características de orientación a objetos del lenguaje.
- f) Se ha creado código para definir la estructura de objetos.
- g) Se han creado métodos y propiedades.
- h) Se ha creado código que haga uso de objetos definidos por el usuario.
- i) Se ha depurado y documentado el código.

¿Cómo valoramos y puntuamos tu tarea?

Rúbrica de la tarea	
Creación del objeto Edificio.	1,5 puntos
Implmementación del método agregarPlantasYPuertas(numplantas, puertas)	2 puntos
Implementación de los métodos: modificarNumero(numero), modificarCalle(calle), modificarCodigoPostal(codigo), imprimeCalle(), imprimeNumero(), imprimeCodigoPostal()	0,5 puntos por método
Implementación del método agregarPropietario (nombre, planta, puerta)	1 punto
Implementación del método imprimePlantas()	1,5 puntos
Claridad y presentación del código del ejercicio, comentarios en el código y su indentación.	1 punto

RESOLUCIÓN

Estructura de la página



Sistema de archivos de la tarea

El código html se limita a una lista que se va llenando con las tareas que se van ejecutando.
El código JavaScript se ha separado en una hoja externa para hacerlo más legible. El archivo de encuentra en *js/edificio.js*.
Se ha eliminado la hoja de estilo externa porque apenas se aplican estilos.

```html
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>DWEC04 Tarea</title>
  </head>
  <body>
    <header style="text-align: center;">
      <h1>Gestión de edificios</h1>
    </header>
    <main>
      <ul id="tareas" style="list-style: none;"></ul>
    </main>
    <script src="js/edificio.js"></script>
  </body>
</html>
```

Código de index.html

```javascript
class Edificio {
  constructor(calle, numero, cp = 1000) {}
  agregarPlantasYPuertas(plantas = 1, puertas = 1) {}
  modificarNumero(numero) {}
  modificarCalle(calle) {}
  modificarCodigoPostal(codigo) {}
  imprimeCalle() {}
  imprimeNumero() {}
  imprimeCodigoPostal() {}
  agregarPropietario(nombre, planta, puerta) {}
  imprimePlantas() {}
  mostrarMensajeDeEdificioConstruido(edificio) {}
  mostrarMensajeDePropietario(planta, puerta) {}
}
```

Listado de funciones de edificio.js

La clase Edificio

Representa un edificio que tiene cuatro propiedades:

- La calle en la que está situado, de tipo *string*.
- El número del edificio, de tipo *string*.
  - El número del edificio se declara *string* dada la posibilidad de que el edificio no tenga número (s/n) o de que exista otro con el mismo número en la misma calle, por ejemplo n.º1 y n.º1 bis.
- El código postal es de tipo *number*.
- Las plantas son una colección puertas a las que se les asigna un propietario (*string*).
  - La estructura de las plantas es `plantas[planta][puerta] = "propietario"`.
  - Las plantas comienzan en 0, pues es lógico que la planta 1 sea la primera. De este modo, la planta 0 es la planta baja.

La clase no valida ni formatea datos, se limita a dar respuesta al enunciado del ejercicio.

Los nombres de las funciones de la clase se ajustan al enunciado del ejercicio.

El constructor

constructor(calle, número, cp)

Recibe como parámetros la calle, el número y el código postal. Podría recibir, también, las plantas y el número de puertas, pero se deja para hacerlo por separado.
Los tres parámetros son obligatorios.

Después de asignar los valores a sus propiedades, se muestra el mensaje de que se ha construido el edificio. Para generar el mensaje se ha creado una función que lo imprime, liberando al constructor de esta tarea.

```
  constructor(calle, numero, cp) {
    this._calle = calle;
    this._numero = numero;
    this._cp = cp;
    this._plantas = [];
    this.mostrarMensajeDeEdificioConstruido(this);
  }
```

Constructor de la clase Edificio

```
 mostrarMensajeDeEdificioConstruido(edificio) {
    document.write(`<li>Construido un nuevo edificio en la calle ${edificio._calle}, n.º
${edificio._numero}, CP: ${edificio._cp}</li>`);
  }
```

Función que muestra el mensaje al crear un edificio

Crear un objeto que nos permita instanciar edificios.

Cada vez que instanciemos un edificio le pasaremos la calle, número y código postal como parámetros.

Para crear el objeto se instancia Edificio, pasando al constructor la calle, el número y código postal.

```
let edificioA = new Edificio("Cebolla", 43, 99001);
let edificioB = new Edificio("Calabacín", 2, 99012);
let edificioC = new Edificio("Pepino", 22, 99002);
```

En el constructor se asignan los valores recibidos a las variables que se han creado como propiedades del objeto. La colección de plantas se inicia como un array vacío.

Los nombres de las propiedades se inician con un guion bajo "_" para diferenciarlas.

```
  constructor(calle, numero, cp) {
    this._calle = calle;
    this._numero = numero;
    this._cp = cp;
    this._plantas = [];

    this.mostrarMensajeDeEdificioConstruido(this);
  }
```

agregarPlantasYPuertas(numplantas, puertas)

Se le pasa el número de plantas que queremos crear en el piso y el número de puertas por planta. Cada vez que se llame a este método, añadirá el número de plantas y puertas indicadas en los parámetros, a las que ya están creadas en el edificio.

El método debe recibir el número de plantas y de puertas por planta. En principio debemos obtener las plantas que ya están creadas, si las hubiera, para poder añadir más.

Se inicia un primer bucle que crea cada planta y, dentro de este, se ejecuta otro para crear cada puerta. El bucle de las plantas se inicia en el valor siguiente que tenga la última planta registrada. El índice de las plantas indica la planta y el de las puertas, el número de puerta. El propietario que se asigna a cada puerta en un *String* en blanco.

```
  agregarPlantasYPuertas(plantas, puertas) {
    let plantasActuales = this._plantas.length;

    for (let i = plantasActuales; i < plantasActuales +
plantas; i++) {
      this._plantas.push(new Array(puertas));

      for (let j = 0; j < puertas; j++) {
        this._plantas[i][j] = "";
      }
    }
  }
```

modificarNumero(numero)
modificarCalle(calle)
modificarCodigoPostal(codigo)

Estos tres métodos trabajan de la misma forma: se accede desde la instancia de la clase a su método, pasándole el valor nuevo y se asigna a la propiedad.

```
edificioA.modificarCodigoPostal(88001);
edificioC.modificarCalle('Pepinillo');
edificioB.modificarNumero(79);
edificioB.modificarCalle('Calabaza');
```

```
modificarNumero(numero) {
  this._numero = numero;
}
modificarCalle(calle) {
  this._calle = calle;
}
modificarCodigoPostal(codigo) {
  this._cp = codigo;
}
```

imprimeCalle
imprimeNumero
imprimeCodigoPostal

Devuelve la calle, el número y el código postal.

Aunque los métodos se llaman "*Imprime*", realmente devuelve una cadena da texto con el valor de la propiedad. Los tres métodos funcionan del mismo modo.

```
// Imprimir una lista de propietarios
document.write(`<h2>Lista de propietarios de la calle ${edificioA.imprimeCalle()} número
${edificioA.imprimeNumero()} :</h2><ul>`);
document.write(edificioA.imprimePlantas());
document.write('</ul>');
```

```
imprimeCalle() {
  return this._calle;
}

imprimeNumero() {
  return this._numero;
}

imprimeCodigoPostal() {
  return this._cp;
}
```

agregarPropietario(nombre,planta,puerta)

Se le pasa un nombre de propietario, un número de planta y un número de puerta y lo asignará como propietario de ese piso.

Desde el objeto de la clase se llama al método pasándole los parámetros y en el método se realiza la asignación.

```
// Agregar propietarios
document.write('<h2>Agregar propietarios:</h2><ul>');
edificioA.agregarPropietario('Benito Boniato', 2, 2);
edificioA.agregarPropietario('Lorenza Coto',1,1);
edificioA.agregarPropietario('Aitor Tilla',0,1);
edificioA.agregarPropietario('Andrés Trozado',2,1);
edificioA.agregarPropietario('Lola Mento',2,0);
edificioA.agregarPropietario('Encarna Vales',1,2);
edificioA.agregarPropietario('Ana Tomía',0,2);
document.write('</ul>');
```

```
  agregarPropietario(nombre, planta, puerta) {
    this._plantas[planta][puerta] = nombre;
    document.write(this.mostrarMensajeDePropietario(planta, puerta));
  }
```

imprimePlantas

Recorrerá el edificio e imprimirá todos los propietarios de cada puerta.

Es un método sencillo: con dos bucles *for* se recorre la colección de plantas y puertas y se va formando el texto.

```
  imprimePlantas() {
    let texto = "";

    for (let planta in this._plantas) {
      texto += `<ul>  <li>Planta ${planta}  <ul>`;

      for (let puerta in this._plantas[planta]) {
        texto += `<li>Puerta ${puerta}: ${this._plantas[planta][puerta]}</li>`;
      }

      texto += `</ul> </li> </ul>`;
    }
    texto += "</li>";
    return texto;
  }
```

Cada vez que se crea un edificio que muestre automáticamente un mensaje del estilo:

construido nuevo edificio en calle: xxxxxx, nº: xx, CP: xxxxx.

En este caso, al ser el constructor, se delega la impresión del mensaje al método que genera el mensaje.

```
  constructor(calle, numero, cp) {
        . . .
    this.mostrarMensajeDeEdificioConstruido(this);
  }
```

```
  mostrarMensajeDeEdificioConstruido(edificio) {
    document.write(`<li>Construido un nuevo edificio en la calle ${edificio._calle}, n.º
${edificio._numero}, CP: ${edificio._cp}</li>`);
  }
```

Cada vez que se añada un propietario a un piso de un edificio que muestre un mensaje del estilo:

xxxxxxxx es ahora el propietario de la puerta x de la planta x.

El método recibe los datos y asigna el propietario a puerta de la planta indicada.
Después de hacer la asignación, se llama a mostrarMensajeDePropietario(planta, puerta) para que devuelva el mensaje en la página e imprimirlo.

```
  agregarPropietario(nombre, planta, puerta) {
    this._plantas[planta][puerta] = nombre;
    document.write(this.mostrarMensajeDePropietario(planta, puerta));
  }
```

```
  mostrarMensajeDePropietario(planta, puerta) {
    return `<li>${this._plantas[planta][puerta]} es ahora el propietario de la puerta ${puerta} de la
planta ${planta}</li>`;
  }
```

Código

index.html

```html
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>DWEC04 Tarea</title>
    <style>
      body{
        padding: 0 10px;
      }
      h2{
        font-size: 1rem;
        margin-bottom: 0;
      }
      ul{
        margin-top: 0;
      }
    </style>
  </head>
  <body>
    <header style="text-align: center;">
      <h1>Gestión de edificios</h1>
    </header>

    <script src="js/edificio.js"></script>
  </body>
</html>
```

edificio.js

```
/**
 * Clase que representa un edificio.
 *
 * Un Edificio tiene una calle, un número y un C.P. y una
 * colección de plantas.
 *
 * @author Roberto Rodríguez <roberto.rod.jim.1@educa.jcyl.es>
 * @class representa un edificio.
 */
class Edificio {

  /**
   * Constructor de la clase.
   *
   * @param {string} calle en la que se situa el edificio
   * @param {string} numero del edificio, string para poder llevar letras (s/n, 21bis, ...)
   * @param {number} cp del edificio
   */
  constructor(calle, numero, cp) {
    // Es una buena práctica comenzar el nombre del atributo con _
    /**
     * @type {string}
     */
    this._calle = calle;

    /**
     * @type {string}
     */
    this._numero = numero;

    /**
     * @type {number}
     */
    this._cp = cp;

    // Inicializar las plantas como un array vacío.
    // Una planta es una colección de puertas.
    this._plantas = [];

    // Imprime en el documento HTML el mensaje pasado como parámetro.
    this.mostrarMensajeDeEdificioConstruido(this);

  }

  /**
   * Agrega las plantas y las puertas por planta al edificio.
   *
```

```
 * Las plantas se suman a las ya creadas en el edificio.
 * La planta 0 es la planta baja, por lo tanto, un edificio
 * de 3 plantas tendrá: bajo, 1º y 2º.
 *
 * @param {number} plantas del edificio.
 * @param {number} puertas de cada planta.
 */
agregarPlantasYPuertas(plantas, puertas) {
  // El edificio puede tener a alguna planta
  let plantasActuales = this._plantas.length;

  // Recorrer las plantas del edificio que queremos agregar.
  // OJO! i contiene el índice de la planta pero, si ya hay
  // plantas, i se puede corresponder con plantas ya creadas.
  // Para evitar esto, i debe partir del último elemento de
  // de la lista de plantas.
  for (let i = plantasActuales; i < plantasActuales + plantas; i++) {
    // Cada planta contiene un array con las puertas.
    this._plantas.push(new Array(puertas));

    // Recorrer las puertas en cada planta.
    for (let j = 0; j < puertas; j++) {
      // A la planta se le asignan las puertas
      // A cada puerta se le asigna un propietario en blanco.
      this._plantas[i][j] = "";
    }
  }
}

/**
 * Modifica el número del edificio.
 *
 * @param {number} numero nuevo del edificio.
 */
modificarNumero(numero) {
  this._numero = numero;
}

/**
 * Actualiza el nombre de la calle del edificio.
 *
 * @param {string} calle nueva.
 */
modificarCalle(calle) {
  this._calle = calle;
}

/**
 * Modifica el CP del edificio.
 *
 * @param {number} codigo
 */
```

```
  modificarCodigoPostal(codigo) {
    this._cp = codigo;
  }

  /**
   * Devuelve el nombre de la calle del edificio.
   *
   * @returns el nombre de la calle.
   */
  imprimeCalle() {
    return this._calle;
  }

  /**
   * Devuelve el número del edificio.
   *
   * @returns el número del edificio.
   */
  imprimeNumero() {
    return this._numero;
  }

  /**
   * Devuelve el código postal del edificio.
   *
   * @returns el código postal.
   */
  imprimeCodigoPostal() {
    return this._cp;
  }

  /**
   * Asigna un propietario a una puerta de una planta.
   *
   * @param {string} nombre del propietario.
   * @param {number} planta en la que está la puerta.
   * @param {number} puerta del propietario.
   */
  agregarPropietario(nombre, planta, puerta) {

    // Asignar el propietario
    this._plantas[planta][puerta] = nombre;

    // Mostrar el mensaje
    document.write(this.mostrarMensajeDePropietario(planta, puerta));
  }

  /**
   * Recorre el edificio e imprime todos los propietarios de cada puerta.
   */
  imprimePlantas() {
    // Iniciamos el texto que se va a mostrar, creando una lista
```

```
    // para cada planta y otra para cada colección de puertas.
    let texto = "";

    // Recorrer las plantas
    for (let planta in this._plantas) {
      texto += `<ul>
        <li>Planta ${planta}
          <ul>`;

      // Recorrer cada puerta de la planta
      for (let puerta in this._plantas[planta]) {
        texto += `<li>Puerta ${puerta}: ${this._plantas[planta][puerta]}</li>`;
      }

      texto += `</ul>
          </li>
        </ul>`;
    }

    texto += "</li>";

    return texto;
  }

  /**
   * Cada vez que se añada un propietario a un piso de un edificio que muestre un mensaje.
   * @param { Edificio } edificio del que se muestran los datos.
   * @returns { string } mensaje de confirmación.
   */
  mostrarMensajeDeEdificioConstruido(edificio) {
    document.write(`<li>Construido un nuevo edificio en la calle ${edificio._calle}, n.º
${edificio._numero}, CP: ${edificio._cp}</li>`);
  }

/**
 * Cada vez que se añada un propietario a un piso de un edificio que muestre un mensaje.
 * @param {number} planta a la que petenece la puerta buscada.
 * @param {number} puerta buscada
 * @returns {string} mensaje que se imprimirá.
 */
  mostrarMensajeDePropietario(planta, puerta) {
    return `<li>${this._plantas[planta][puerta]} es ahora el propietario de la puerta ${puerta} de la
planta ${planta}</li>`;
  }


}
// Fin de la clase edificios ----------------------

// Declarar los edificios
document.write('<h2>Crear 3 edificios: A, B, C</h2><ul>');
let edificioA = new Edificio("Cebolla", 43, 99001);
```

```
let edificioB = new Edificio("Calabacín", 2, 99012);
let edificioC = new Edificio("Pepino", 22, 99002);
document.write('</ul>')

// Imprimir algunos datos de cada edificio
document.write('<h2>Algunos datos:</h2><ul>');
document.write(`<li>El código postal del edificioA es: ${edificioA.imprimeCodigoPostal()}</li>`);
document.write(`<li>La calle del edificio C es: ${edificioC.imprimeCalle()}</li>`);
document.write(`<li>El edificio B está situado en la calle ${edificioB.imprimeCalle()} número
${edificioB.imprimeNumero()}</li>`);
document.write('</ul>')

// Cambiar algunos datos de cada edificio
document.write('<h2>Cambiamos los datos:</h2><ul>');
edificioA.modificarCodigoPostal(88001);
edificioC.modificarCalle('Pepinillo');
edificioB.modificarNumero(79);
edificioB.modificarCalle('Calabaza');
document.write(`<li>El nuevo código postal del edificioA es: ${edificioA.imprimeCodigoPostal()}</li>`);
document.write(`<li>Ahora, la calle del edificio C es: ${edificioC.imprimeCalle()}</li>`);
document.write(`<li>La nueva situación del edificio B es la calle ${edificioB.imprimeCalle()} número
${edificioB.imprimeNumero()}</li>`);
document.write('</ul>');

// Agregar plantas y puertas
document.write(`<h2>Agregar plantas y puertas (planta baja es 0):</h2>
  <ul >
    <li>Edifico A.: 3 plantas y 3 puertas por planta.</li>
    <li>Edifico B.: 2 plantas y 4 puertas por planta.</li>
    <li>Edifico c.: 4 plantas y 2 puertas por planta.</li>
  </ul>`);
edificioA.agregarPlantasYPuertas(3,3);
edificioB.agregarPlantasYPuertas(2,4);
edificioC.agregarPlantasYPuertas(4,2);

// Agregar propietarios
document.write('<h2>Agregar propietarios:</h2><ul>');
edificioA.agregarPropietario('Benito Boniato', 2, 2);
edificioA.agregarPropietario('Lorenza Coto',1,1);
edificioA.agregarPropietario('Aitor Tilla',0,1);
edificioA.agregarPropietario('Andrés Trozado',2,1);
edificioA.agregarPropietario('Lola Mento',2,0);
edificioA.agregarPropietario('Encarna Vales',1,2);
edificioA.agregarPropietario('Ana Tomía',0,2);
document.write('</ul>');

// Imprimir una lista de propietarios
document.write(`<h2>Lista de propietarios de la calle ${edificioA.imprimeCalle()} número
${edificioA.imprimeNumero()} :</h2><ul>`);
document.write(edificioA.imprimePlantas());
document.write('</ul>');
```

```
// Agregar plantas
document.write('<h2>Agregar 2 plantas, con 2 puertas cada una, al edificio A.</h2>');
edificioA.agregarPlantasYPuertas(2,2);

// Agregar propietarios a las plantas nuevas
document.write('<h2>Agregar propietarios a las plantas nuevas:</h2><ul>');
edificioA.agregarPropietario('Matías Queroso', 3, 1);
edificioA.agregarPropietario('Igor Dito',4,0);
document.write('</ul>');

// Listar nuevamente las plantas
document.write('<h2>Así quedan las viviendas del edificio A:</h2><ul>');
document.write(edificioA.imprimePlantas());
document.write('</ul>');
```