

PROGRAMACION ORIENTADA A OBJETOS

ENTREGABLE 1



Alumno: Luis Enrique Mendoza Rubí

Matrícula: 13803551

Profesor: Saúl Santiago Rivera

Carrera: Ingeniería en Software y Redes

Ciclo: 25-1

INTRODUCCIÓN

En este proyecto diseñé 3 programas con las especificaciones que nos solicitó en el entregable numero uno. Las tecnologías que utilicé son para desarrollo web, y específicamente utilicé JavaScript que es uno de los lenguajes orientados a objetos más utilizados de la actualidad. Como lo vimos en nuestra clase síncrona, la POO es un estilo de programación que organiza el código en torno a objetos, que son representaciones de entidades del mundo real o conceptos, como personas, autos, productos, etc. Cada objeto tiene:

- **Atributos:** Propiedades que describen al objeto. Ejemplo: un auto tiene color, marca, modelo.
- **Métodos:** Comportamientos o funciones que el objeto puede realizar. Ejemplo: un auto puede arrancar, frenar o acelerar.

JavaScript, aunque originalmente no estaba diseñado para ser un lenguaje orientado a objetos de manera clásica, ha evolucionado para soportar la POO de manera robusta. A partir de ES6 (ECMAScript 2015), se introdujo la sintaxis de clases, que hace que trabajar con objetos sea más familiar para quienes vienen de lenguajes como Java o C#.

¿Cómo funciona la POO en JavaScript?

1. **Clases:** Son plantillas para crear objetos. Definen los atributos (propiedades) y los métodos (funciones) que los objetos de esa clase tendrán.
2. **Objetos:** Instancias de una clase. Cada objeto tiene su propio conjunto de atributos.
3. **Constructores:** Son funciones especiales dentro de una clase que se ejecutan cuando un objeto es creado. Generalmente, se usan para inicializar las propiedades del objeto.

DESARROLLO

Quería comentarle profesor, adicionalmente a las evidencias en pdf que le estoy anexando en este documento, sería un gran gusto que ingrese a mi perfil de GitHub donde podrá corroborar mi autoría y el funcionamiento de las aplicaciones. Diseñé los tres programas teniendo en mente que los datos puedan ser ingresados por el usuario, por lo que usted podrá verificar mis procedimientos con los datos solicitados de cada programa. Le comparto el link del repositorio donde guardé este proyecto: <https://github.com/LuisoRubi/entregable1>

Me ayudaría mucho recibir cualquier feedback, comentario o sugerencia de alguien con la experiencia que tiene usted. Sin más por el momento le comparto mi desarrollo:

Ejercicio 1.

Código:

// Ejercicio 1: Plantear una clase Alumno con atributos de nombre, apellido, edad, direccion. Lo verificaremos con un programa para inscribir alumnos.

```
class Alumno {
    constructor(nombre, apellido, edad, direccion) {
        this.nombre = nombre;
        this.apellido = apellido;
        this.edad = edad;
        this.direccion = direccion;
    }

    // Método para mostrar los detalles del alumno
    mostrarAlumnx() {
        return `Alumnx: ${this.nombre} ${this.apellido}, Edad: ${this.edad}, Dirección:
        ${this.direccion}`;
    }
}
```

```
}
```

// Función para inscribir un alumno

```
function inscribirAlumno() {
```

```
    // Obtener los valores ingresados por el usuario
```

```
    const nombre = document.getElementById("nombre").value;
```

```
    const apellido = document.getElementById("apellido").value;
```

```
    const edad = document.getElementById("edad").value;
```

```
    const direccion = document.getElementById("direccion").value;
```

```
    // Crear un nuevo objeto Alumno con los datos
```

```
    const nuevoAlumno = new Alumno(nombre, apellido, edad, direccion);
```

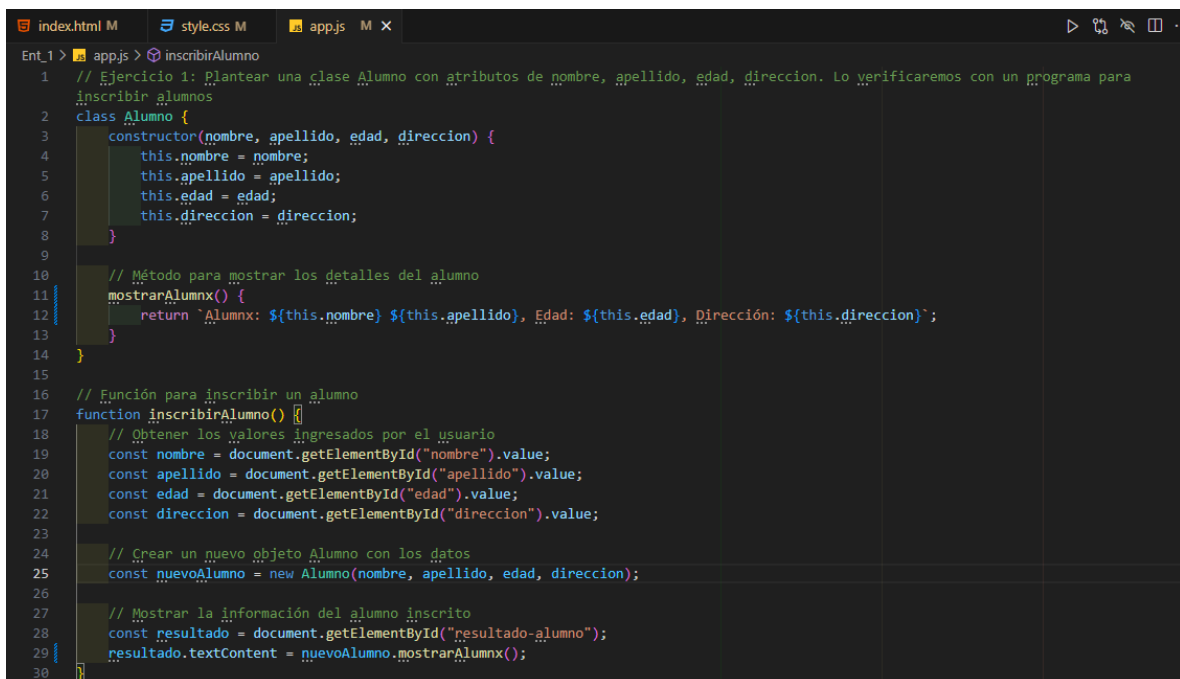
```
    // Mostrar la información del alumno inscrito
```

```
    const resultado = document.getElementById("resultado-alumno");
```

```
    resultado.textContent = nuevoAlumno.mostrarAlumnx();
```


```
}
```

Captura de pantalla del código:



```
Ent_1 > app.js > inscribirAlumno
1 // Ejercicio 1: Plantear una clase Alumno con atributos de nombre, apellido, edad, direccion. Lo verificaremos con un programa para
  inscribir alumnos
2 class Alumno {
3     constructor(nombre, apellido, edad, direccion) {
4         this.nombre = nombre;
5         this.apellido = apellido;
6         this.edad = edad;
7         this.direccion = direccion;
8     }
9
10    // Método para mostrar los detalles del alumno
11    mostrarAlumnx() {
12        return `Alumnx: ${this.nombre} ${this.apellido}, Edad: ${this.edad}, Dirección: ${this.direccion}`;
13    }
14 }
15
16 // Función para inscribir un alumno
17 function inscribirAlumno() {
18     // Obtener los valores ingresados por el usuario
19     const nombre = document.getElementById("nombre").value;
20     const apellido = document.getElementById("apellido").value;
21     const edad = document.getElementById("edad").value;
22     const direccion = document.getElementById("direccion").value;
23
24     // Crear un nuevo objeto Alumno con los datos
25     const nuevoAlumno = new Alumno(nombre, apellido, edad, direccion);
26
27     // Mostrar la información del alumno inscrito
28     const resultado = document.getElementById("resultado-alumno");
29     resultado.textContent = nuevoAlumno.mostrarAlumnx();
30 }
```

Captura de pantalla de los resultados:

**Luiso Rubí**
software engineer

Hola profesor Saúl Santiago Rivera. Mi nombre es Luis Enrique Mendoza Rubí, estudiante de la carrera Ingeniería en Software y Redes del séptimo cuatrimestre.

A continuación le comparto mis ejercicios resueltos para el primer entregable. Las tecnologías que estoy utilizando son: VS Code como IDE, HTML, CSS Y JavaScript para la solución de los ejercicios.

Ejercicio 1:

Plantear una clase llamada Alumno y definir como atributos su nombre, edad, apellido y dirección. En el constructor realizar la carga de datos con el método this. Esto se verificará con un programa básico para inscribir alumnos. Por favor ingrese los datos del nuevx alumnx:

Nombre:

Apellido:

Edad:

Dirección:

Alumnx: Pedrito Martínez, Edad: 25, Dirección: Miraflores 34, Naucalpan de Juárez, México

Ejercicio 2:

Código:

// Ejercicio 2: Verificar si una fecha corresponde a Navidad

```
function verificarNavidad() {  
    const fechaIngresada = document.getElementById("fecha").value;  
    const fecha = new Date(fechaIngresada);  
    const resultado = document.getElementById("resultado-navidad");  
  
    if (fecha.getMonth() === 11 && fecha.getDate() === 25) {  
        resultado.textContent = "¡Es Navidad! Felicidades. 😊✅🎅";  
    } else {  
        resultado.textContent = "No es Navidad, lo siento 😞❌🎅";  
    }  
}
```

Captura de pantalla del código:

```
31
32 // Ejercicio 2: Verificar si una fecha corresponde a Navidad
33 function verificarNavidad() {
34     const fechaIngresada = document.getElementById("fecha").value;
35     const fecha = new Date(fechaIngresada);
36     const resultado = document.getElementById("resultado-navidad");
37
38     if (fecha.getMonth() === 11 && fecha.getDate() === 25) {
39         resultado.textContent = "¡Es Navidad! Felicidades. 😊✔️🎅";
40     } else {
41         resultado.textContent = "No es Navidad, lo siento 😞❌🎅";
42     }
43 }
44
```

Captura de pantalla de los resultados:

Ejercicio 2:

Realizar un programa que pida cargar una fecha cualquiera, luego verificar si dicha fecha corresponde a Navidad.

Ingrese una fecha con el formato AAAA/MM/DD:

¡Es Navidad! Felicidades. 😊✔️🎅

Ejercicio 2:

Realizar un programa que pida cargar una fecha cualquiera, luego verificar si dicha fecha corresponde a Navidad.

Ingrese una fecha con el formato AAAA/MM/DD:

No es Navidad, lo siento 😞❌🎅

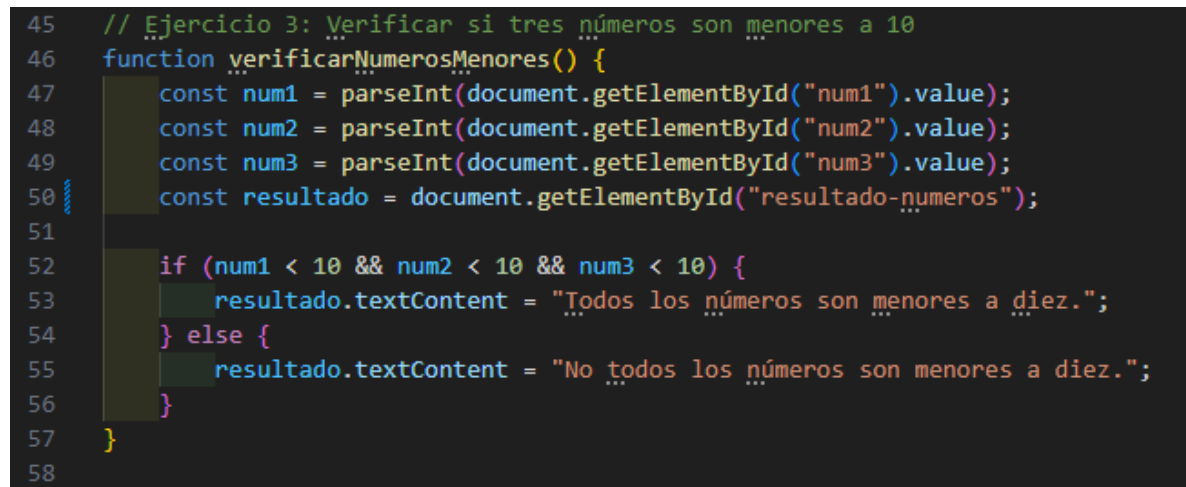
Ejercicio 3:

Código:

// Ejercicio 3: Verificar si tres números son menores a 10

```
function verificarNumerosMenores() {  
    const num1 = parseInt(document.getElementById("num1").value);  
    const num2 = parseInt(document.getElementById("num2").value);  
    const num3 = parseInt(document.getElementById("num3").value);  
    const resultado = document.getElementById("resultado-numeros");  
  
    if (num1 < 10 && num2 < 10 && num3 < 10) {  
        resultado.textContent = "Todos los números son menores a diez.";  
    } else {  
        resultado.textContent = "No todos los números son menores a diez.";  
    }  
}
```

Captura de pantalla del código:



```
45 // Ejercicio 3: Verificar si tres números son menores a 10  
46 function verificarNumerosMenores() {  
47     const num1 = parseInt(document.getElementById("num1").value);  
48     const num2 = parseInt(document.getElementById("num2").value);  
49     const num3 = parseInt(document.getElementById("num3").value);  
50     const resultado = document.getElementById("resultado-numeros");  
51  
52     if (num1 < 10 && num2 < 10 && num3 < 10) {  
53         resultado.textContent = "Todos los números son menores a diez.";  
54     } else {  
55         resultado.textContent = "No todos los números son menores a diez.";  
56     }  
57 }  
58
```

Captura de pantalla del resultado:

Ejercicio 3:

Se ingresan por teclado tres números, si todos los valores ingresados son menores a 10, imprimir en pantalla la leyenda "Todos los números son menores a diez".

Número 1:

Número 2:

Número 3:

Todos los números son menores a diez.

Ejercicio 3:

Se ingresan por teclado tres números, si todos los valores ingresados son menores a 10, imprimir en pantalla la leyenda "Todos los números son menores a diez".

Número 1:

Número 2:

Número 3:

No todos los números son menores a diez.

CONCLUSIONES

Los ejercicios que realicé en este entregable nos ayudan a reforzar conocimientos específicos del uso de clases, objetos, métodos y atributos además de complementar con el uso de funciones. Adicionalmente me ayudó a trabajar mis conocimientos en el área del desarrollo web con tecnologías como HTML y CSS. Comprender el funcionamiento de la POO en este tipo de entorno es primordial ya que la salida de los resultados y el mismo procesamiento de los problemas es mucho más legible con el uso de estas tecnologías.

La POO organiza el código alrededor de objetos, y JavaScript permite trabajar con clases, herencia, y objetos de manera eficiente. Aprender estos conceptos es clave para convertirnos en buenos desarrolladores, ya que nos permitirá escribir código más organizado, limpio y fácil de mantener.

REFERENCIAS BIBLIOGRÁFICAS

García, J. (2019). Fundamentos de la programación orientada a objetos. Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología, 22, 67-78.

Deitel, P. J., & Deitel, H. M. (2009). Java: Cómo programar (8.^a ed.). Pearson Educación.

Mozilla Developer Network. (n.d.). Introducción a la programación orientada a objetos en JavaScript. Recuperado de <https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects>

Zavala, L. (2016). JavaScript y jQuery: Interactividad en la web (1.^a ed.). Anaya Multimedia.