Evaluación 1 Optativa de JavaScript

En todos los casos el estudiante debe:

* Entregar el código de su respuesta mediante el enlace del repositorio de su GitHub
* Hacer uso de clases y funciones de JavaScript. Es opcional el uso de conceptos de POO como Herencia y Polimorfismo, y el estudiante puede utilizarlos si lo considera necesario.

El estudiante puede asumir a su gusto todo lo que no esté explícito en el enunciado.

1. **Gestor de Libros**

Crea una clase "Libro" que tenga propiedades como "título", "autor" y "año de publicación". Luego, implementa funciones que permitan agregar, editar y eliminar libros en una lista de libros. Además, desarrolla otra función que permita buscar libros por autor.

1. **Registro de Empleados**

Diseña una clase "Empleado" con propiedades como "nombre", "cargo" y "salario". Luego, desarrolla funciones que permitan agregar, editar y eliminar empleados en un registro de empleados. Además, crea una función que calcule el salario promedio de los empleados.

1. **Gestor de Proyectos**

Implementa una clase "Proyecto" que contenga propiedades como "nombre", "descripción" y "fecha de inicio". A continuación, crea funciones que posibiliten agregar, editar y eliminar proyectos en una lista de proyectos. Además, desarrolla una función que muestre el número total de proyectos en el registro.

1. **Inventario de Productos**

Crea una clase "Producto" con propiedades como "nombre", "precio" y "cantidad en stock". Luego, implementa funciones que permitan agregar, editar y eliminar productos en un inventario de productos. Además, crea una función que calcule el valor total del inventario.

1. **Gestor de Tareas de Proyecto**

Diseña una clase "Tarea" que tenga propiedades como "nombre", "prioridad" y "estado" (pendiente, en progreso, completada). Luego, desarrolla funciones que permitan agregar, editar y eliminar tareas en una lista de tareas de proyecto. Además, crea una función que muestre todas las tareas pendientes por prioridad.

1. **Gestor de Clientes**

Crea una clase "Cliente" que contenga propiedades como "nombre", "correo electrónico" y "número de teléfono". Implementa funciones que permitan agregar, editar y eliminar clientes en una lista de clientes. Además, desarrolla una función que permita buscar clientes por correo electrónico.

1. **Registro de Películas**

Diseña una clase "Película" con propiedades como "título", "género" y "año de lanzamiento". Luego, crea funciones que permitan agregar, editar y eliminar películas en un registro de películas. Además, implementa una función que calcule la cantidad total de películas en el registro.

1. **Gestor de Cuentas de Usuario**

Implementa una clase "CuentaUsuario" que tenga propiedades como "nombre de usuario", "contraseña" y "correo electrónico". A continuación, desarrolla funciones que permitan agregar, editar y eliminar cuentas de usuario en una lista. Además, crea una función que verifique la fortaleza de las contraseñas.

1. **Control de Inventario de Libros**

Crea una clase "Libro" con propiedades como "título", "autor" y "cantidad en stock". Luego, implementa funciones que permitan agregar, editar y eliminar libros en un control de inventario de libros. Además, desarrolla una función que actualice automáticamente la cantidad en stock cuando se venden libros.

1. **Gestor de Notas Escolares**

Diseña una clase "Estudiante" con propiedades como "nombre", "materias" y "calificaciones". Implementa funciones que permitan agregar, editar y eliminar estudiantes en un registro de estudiantes. Además, crea una función que calcule el promedio de calificaciones de un estudiante en todas las materias.

1. **Registro de Eventos**

Implementa una clase "Evento" que tenga propiedades como "nombre del evento", "fecha" y "ubicación". Desarrolla funciones que permitan agregar, editar y eliminar eventos en un registro de eventos. Además, crea una función que liste todos los eventos por fecha.

1. **Gestor de Proveedores de Servicios**

Crea una clase "Proveedor" con propiedades como "nombre", "servicios ofrecidos" y "ubicación". Luego, implementa funciones que permitan agregar, editar y eliminar proveedores en un registro de proveedores. Además, desarrolla una función que muestre los proveedores que ofrecen un servicio específico.

1. **Control de Gastos Personales**

Diseña una clase "Gasto" que tenga propiedades como "categoría", "monto" y "fecha". Implementa funciones que permitan agregar, editar y eliminar gastos en un registro de gastos personales. Además, crea una función que calcule el total de gastos en una categoría específica.

1. **Gestor de Proyectos de Desarrollo**

Implementa una clase "Proyecto" que contenga propiedades como "nombre", "descripción" y "estado" (en curso, finalizado). A continuación, crea funciones que posibiliten agregar, editar y eliminar proyectos de desarrollo en una lista. Además, desarrolla una función que muestre todos los proyectos en curso.

1. **Registro de Pedidos en Línea**

Crea una clase "Pedido" con propiedades como "producto", "cantidad" y "cliente". Luego, implementa funciones que permitan agregar, editar y eliminar pedidos en un registro de pedidos en línea. Además, crea una función que calcule el valor total de pedidos de un cliente específico.

1. **Control de Tareas de Mantenimiento**

Diseña una clase "TareaMantenimiento" que tenga propiedades como "equipo", "descripción" y "fecha de vencimiento". Implementa funciones que permitan agregar, editar y eliminar tareas de mantenimiento en un control de tareas. Además, desarrolla una función que muestre las tareas pendientes por equipo.

1. **Gestor de Proveedores de Suministros**

Implementa una clase "ProveedorSuministros" con propiedades como "nombre", "productos suministrados" y "ubicación". A continuación, crea funciones que permitan agregar, editar y eliminar proveedores de suministros en un registro. Además, crea una función que muestre los proveedores que suministran un producto específico.

1. **Sistema de Reservas de Viajes**

Crea una clase "Viaje" con propiedades como "destino", "fecha de salida" y "precio". Luego, implementa funciones que permitan agregar, editar y eliminar reservas de viajes en un sistema de reservas. Además, desarrolla una función que calcule el costo total de un viaje con múltiples reservas.

1. **Control de Exámenes Estudiantiles**

Diseña una clase "Examen" que tenga propiedades como "materia", "puntaje" y "fecha de examen". Implementa funciones que permitan agregar, editar y eliminar exámenes en un control de exámenes estudiantiles. Además, crea una función que calcule el promedio de puntajes por materia.

1. **Gestor de Almacenes**

Implementa una clase "ProductoAlmacén" que tenga propiedades como "nombre", "cantidad en stock" y "ubicación en el almacén". Desarrolla funciones que permitan agregar, editar y eliminar productos en un almacén. Además, crea una función que muestre la ubicación de un producto específico.

1. **Registro de Proyectos de Construcción**

Crea una clase "ProyectoConstrucción" con propiedades como "nombre", "ubicación" y "presupuesto". Implementa funciones que permitan agregar, editar y eliminar proyectos de construcción en un registro. Además, desarrolla una función que muestre los proyectos dentro de un rango de presupuesto específico.

1. **Gestor de Vehículos**

Diseña una clase "Vehículo" con propiedades como "marca", "modelo" y "año". Luego, crea funciones que permitan agregar, editar y eliminar vehículos en un registro de vehículos. Además, implementa una función que muestre todos los vehículos de una marca específica.

1. **Control de Actividades Deportivas**

Implementa una clase "ActividadDeportiva" que tenga propiedades como "nombre", "fecha" y "participantes". Desarrolla funciones que permitan agregar, editar y eliminar actividades deportivas en un control. Además, crea una función que muestre las actividades con el mayor número de participantes.

1. **Registro de Mascotas**

Crea una clase "Mascota" con propiedades como "nombre", "tipo" y "edad". Luego, implementa funciones que permitan agregar, editar y eliminar mascotas en un registro de mascotas. Además, desarrolla una función que clasifique las mascotas por tipo.

1. **Gestor de Restaurantes**

Diseña una clase "Restaurante" con propiedades como "nombre", "ubicación" y "tipo de cocina". Implementa funciones que permitan agregar, editar y eliminar restaurantes en un gestor de restaurantes. Además, crea una función que muestre todos los restaurantes de un tipo de cocina específico.

1. **Control de Horarios de Clases**

Implementa una clase "Clase" que tenga propiedades como "materia", "día de la semana" y "hora de inicio". Desarrolla funciones que permitan agregar, editar y eliminar clases en un control de horarios. Además, crea una función que muestre las clases programadas para un día específico.

1. **Gestor de Recetas de Cocina**

Crea una clase "Receta" con propiedades como "nombre", "ingredientes" y "instrucciones de preparación". Luego, implementa funciones que permitan agregar, editar y eliminar recetas en un gestor de recetas de cocina. Además, desarrolla una función que permita buscar recetas por ingredientes.

1. **Control de Actividades de Proyectos**

Diseña una clase "ActividadProyecto" que tenga propiedades como "nombre", "estado" y "fecha de inicio". Implementa funciones que permitan agregar, editar y eliminar actividades de proyectos en un control. Además, crea una función que muestre todas las actividades en curso.

1. **Registro de Exposiciones de Arte**

Implementa una clase "ExposiciónArte" que contenga propiedades como "nombre", "artista" y "fecha de exhibición". Desarrolla funciones que permitan agregar, editar y eliminar exposiciones de arte en un registro. Además, desarrolla una función que muestre las exposiciones de un artista específico.

1. **Gestor de Alumnos en un Curso**

Crea una clase "AlumnoCurso" con propiedades como "nombre", "calificaciones" y "asistencia". Luego, implementa funciones que permitan agregar, editar y eliminar alumnos en un registro de alumnos en un curso. Además, desarrolla una función que calcule el promedio de calificaciones de todos los alumnos.

1. **Registro de Proyectos de Investigación**

Diseña una clase "ProyectoInvestigación" con propiedades como "título", "investigadores" y "fecha de inicio". Implementa funciones que permitan agregar, editar y eliminar proyectos de investigación en un registro. Además, crea una función que muestre los proyectos en los que participa un investigador específico.

1. **Gestor de Tareas de Marketing**

Implementa una clase "TareaMarketing" que tenga propiedades como "nombre", "fecha de vencimiento" y "responsable". Desarrolla funciones que permitan agregar, editar y eliminar tareas de marketing en un control. Además, crea una función que liste las tareas pendientes por fecha de vencimiento.

1. **Control de Compras en Línea**

Crea una clase "Compra" con propiedades como "producto", "cantidad" y "precio unitario". Luego, implementa funciones que permitan agregar, editar y eliminar compras en un registro de compras en línea. Además, desarrolla una función que calcule el total gastado en compras.

1. **Gestor de Entrenamientos Deportivos**

Diseña una clase "EntrenamientoDeportivo" con propiedades como "nombre", "duración" y "tipo de ejercicio". Implementa funciones que permitan agregar, editar y eliminar entrenamientos deportivos en un gestor de entrenamientos. Además, crea una función que muestre los entrenamientos por tipo de ejercicio.