Prueba - Programación avanzada en JavaScript

En esta prueba final, evaluaremos tu comprensión y habilidades en los conceptos avanzados de JavaScript cubiertos durante el módulo.

Lee todo el documento antes de comenzar el desarrollo individual, para asegurarte de tener el máximo de puntaje y enfocar bien los esfuerzos.

// Tiempo asociado: 2 horas cronológicas.

Descripción

"TechFlow", una startup de tecnología, está desarrollando "ProjectPulse", una plataforma de gestión de proyectos para equipos de desarrollo de software.
Tu tarea es implementar funcionalidades clave en JavaScript para esta plataforma, incluyendo:

- Manejo de datos de proyectos y tareas
- Análisis y filtrado de tareas
- Simulación de operaciones asíncronas
- Sistema de notificaciones en tiempo real

Al final de este documento encontrarás algunos ejemplos de implementación que te **ayudarán a desarrollar los requerimientos de la prueba**.

Requerimientos

1. Gestión de Proyectos y Tareas **(3 puntos)**

    - Crea una estructura de datos para representar proyectos y tareas.
        - Cada proyecto debe tener un ID, nombre, fecha de inicio, y un array de tareas.
        - Cada tarea debe tener un ID, descripción, estado (pendiente, en progreso, completada) y fecha límite.

      Nota: Utiliza como referencia la **"Presentación - Introducción a Objetos"**.

    - Implementa una función que permita añadir nuevas tareas a un proyecto.

    - Desarrolla una función que utilice métodos de array (map, filter, reduce) para generar un resumen del proyecto mostrando el número de tareas en cada estado.

    - Crea una función que ordene las tareas de un proyecto por fecha límite utilizando el método sort de JavaScript.

2. Análisis Avanzado de Tareas **(3 puntos)**

    - Crea una función de orden superior **filtrarTareasProyecto** que tome una función de filtrado como argumento y la aplique a la lista de tareas de un proyecto.

    - Implementa una función **calcularTiempoRestante** que utilice el método reduce para calcular el número total de días que faltan para completar todas las tareas pendientes de un proyecto.

    - Desarrolla una función obtenerTareasCriticas que identifique y retorne las tareas que están a menos de 3 días de su fecha límite y aún no están completadas.

3. Sincronización y Actualizaciones en Tiempo Real **(4 puntos)**

    - Desarrolla una función cargarDetallesProyecto que simule una llamada asíncrona a una API para cargar los detalles de un proyecto.
      Utiliza Promises o async/await.

- Crea una función actualizarEstadoTarea que simule la actualización del estado de una tarea en el servidor y maneje tanto el caso de éxito como el de error.

- Implementa un sistema simple de notificacionesTareas que permita a diferentes partes del código "escuchar" cuando se completa una tarea.

¡Mucho éxito! 😊

Consideraciones y recomendaciones

- Proporciona tu código JavaScript en un solo archivo, bien comentado y organizado.

- Incluye ejemplos de uso para cada funcionalidad implementada.

Ejemplos de implementación

Añadir nuevos elementos a un arreglo de objetos

```javascript
let personas = [
  { id: 1, nombre: "Ana", edad: 28 },
  { id: 2, nombre: "Carlos", edad: 32 },
];

// 1. Método push (imperativo)
personas.push({ id: 3, nombre: "Elena", edad: 25 });

console.log("\nDespués de usar push:");
console.log(personas);
```

Ejemplo de filtrado a partir de una condición.

```javascript
personas_filtradas = personas.map((p) => console.log(p.nombre == "Ana"));
```

Ejemplo de orden de datos mediante una función.

```javascript
const compararPorEdad = (a, b) => a.edad - b.edad;
const personasOrdenadasConFuncion = [...personas].sort(compararPorEdad);
console.log(personasOrdenadasConFuncion);
```

Ejemplo de filtrado de información a partir de una función de orden superior

```javascript
// Lista de personas
const personas = [
  { nombre: "Ana", edad: 28, ciudad: "Madrid" },
  { nombre: "Carlos", edad: 32, ciudad: "Barcelona" },
  { nombre: "Elena", edad: 25, ciudad: "Valencia" },
  { nombre: "David", edad: 35, ciudad: "Madrid" },
  { nombre: "Beatriz", edad: 27, ciudad: "Barcelona" },
];

// Función de orden superior para filtrar personas
```

```
const filtrarPersonas = (personas, funcionFiltro) => personas.filter(funcionFiltro);

// Ejemplo de función de filtrado
const mayoresDe30 = (persona) => persona.edad > 30;

console.log(filtrarPersonas(personas, mayoresDe30));
```

Ejemplo de solicitud de información a una API

JavaScript

```
// Función para obtener un usuario de la API JSONPlaceholder
const obtenerUsuario = async (id) => {
  try {
    const response = await fetch(`https://jsonplaceholder.typicode.com/users/${id}`);
    if (!response.ok) {
      throw new Error(`HTTP error! status: ${response.status}`);
    }
    const usuario = await response.json();
    console.log(usuario);
  } catch (error) {
    console.error("Error al obtener el usuario:", error.message);
     throw error; // Re-lanzamos el error para manejarlo en la función que llama
  }
};

obtenerUsuario(1);
```