Primer Parcial Mayo 2023

Para evaluar este examen se tendrá en cuenta tanto el correcto funcionamiento de los ejercicios, como la eficiencia de los mismos, por esa razón un ejercicio podrá obtener la máxima nota siempre y cuando tenga en cuenta ambas premisas.

Los ejercicios deben estar resueltos en una página HTML para demostrar su correcto funcionamiento, pudiéndose utilizar la consola, además se debe utilizar una codificación estricta de JavaScript. **Si un ejercicio no está demostrado, el ejercicio se puntuará con un máximo de la mitad de la nota de dicho ejercicio.**

En cada ejercicio se deberá utilizar los métodos de los diferentes objetos del API de JavaScript para su resolución. En el caso de implementar funcionalidad que ya exista en el API, el ejercicio se verá penalizado por dicha circunstancia.

Por último, no se pueden presentar errores en las pruebas de los ejercicios. Errores de sintaxis o no captura de excepciones implicarán la reducción de la nota o la pérdida total de la puntuación.

Instrucciones de entrega

Crea una carpeta con tu nombre, ejemplo “LizanoMontalvoPablo”, y dentro de ella crea una carpeta por cada ejercicio, ejemplo “ejercicio1”, “ejercicio2”, etc, las cuales deberán contener la resolución de cada ejercicio. Empaqueta la carpeta principal en un ZIP y súbelo a Delphos.

Si lo ves necesario, explica el ejercicio en un párrafo de la página HTML que hayas creado.

1. Ejercicio único

La tienda de electrodomésticos **SportClubMaestre** necesita una aplicación para gestionar los miembros del club.

En la siguiente tabla describimos la estructura del objeto **Member** que representará los miembros del club. La clase debe ser **abstracta**.

Descripción del objeto Member	
Propiedad o método	**Descripción**
Member(nif, firstname, lastname, born)	Constructor
id	Se trata de un String compuesto por: - Las dos iniciales del jugador. - El año de creación. - Secuencia de objetos creados. Todos tienen que estar separados por guiones, todo en mayúsculas. El siguiente ejemplo representaría el identificador para “Pablo Lizano” creado en 2023 y el quinto miembro inscrito ese año: **PL-2023-5**
nif	NIF del miembro. Deberá estar validado con expresión regular.
firstname	Nombre.

lastname	Apellido.
born	Fecha de nacimiento
creationDate	Fecha de creación del objeto.
toString()	Método que transforma el objeto en un String.

Tabla 1 Objeto Member

Las clases hijas de Member especializan los jugadores en dos tipos:

- **BasketballPlayer**.

Descripción del objeto: Television	
Propiedad o método	**Descripción**
height	Altura en centímetros.

Tabla 2 Objeto Television

- **SoccerPlayer**.

Descripción del objeto: Fridge	
Propiedad o método	**Descripción**
position	Posición del jugador.

Tabla 3 Objeto Fridge

El segundo tipo objeto a implementar será **SportClubMaestre** que nos permite gestionar la aplicación. Este objeto debe contener la estructura para almacenar los miembros del club de forma privada y garantizando el acceso seguro. Los métodos que debe implementar y su funcionalidad quedan recogidos en la siguiente tabla. **El objeto debe ser único**.

Método	**Descripción**
SportClubMaestre()	Constructor del objeto.
insert(Member)	Permite añadir un nuevo miembro en la aplicación. Debemos chequear que no exista un miembro creado previamente con el mismo NIF. El método tiene la peculiaridad de que debe permitir añadir más de un electrodoméstico de forma simultánea. El método debe poder encadenarse.
delete(id)	Nos permite eliminar un miembro de la aplicación a partir de su identificador. Si el id no está registrada debemos comunicarlo. El método debe poder encadenarse.
toString(function)	Por defecto, el método muestra todos los objetos registrados en forma de String, y ordenados por el identificador.

	Si el método recibe una función, ésta debe ser utilizada para ordenar el resultado de la salida.
filter(function)	Devolvemos un iterador con los objetos que cumplan la condición configurada en la función. Los objetos deben ser devueltos según los vayamos encontrando. No será válido obtener crear un array para luego obtener el iterador en base al array.

Tabla 4 Descripción del objeto SportClubMaestre

Además, el objeto SportClubMaestre debe ser un **objeto iterable e iterador**, en el que devuelva los objetos Member.

Nota: deberemos gestionar con excepciones todos los puntos del código donde detectemos algún dato no válido o circunstancia que no nos permita finalizar la ejecución del método. Las excepciones serán valoradas como parte de la implementación de los métodos.

Puntuación:

Las funcionalidades deben estar testeadas para poder ser puntuadas. En el caso de no ser testeadas en una función, la funcionalidad será valorada con un 0.

Funcionalidad	**Puntuación**
Implementación de la estructura de objetos. - Las propiedades deben ser implementados mediante campos privados. - En Member las propiedades id, nif y creationDate debe ser de solo lectura. - Los métodos que puedan ser heredados deben ser implementados en el prototipo de los objetos.	1
Función generador de identificadores. Debes mantener la secuencia de identificadores por cada año, además de devolver el siguiente identificador disponible. Deberás elegir el lugar idóneo para albergar la función.	1,5
Validación del NIF.	0,5
Insertar objetos Member en la app. Inserta al menos 3 objetos de cada subclase.	1
Borrar objetos Member en la app.	0,5
toString de la app	0,5
Permitir ordenar la cadena devuelta de toString() con diferentes funciones.	1
Filtrado.	1
Muestra los miembros mayores de edad que sean BasketballPlayers	1,5
SportClubMaestre iterable e iterador. Demuestra que es iterable mediante un bucle.	1
Singleton	0,5
Total	**10 puntos**