

DWEC - Examen 2 de diciembre de 2022

Contenido

Instrucciones	1
Ejercicios RA4	2
Ejercicio 1	4
Ejercicio 2	4
Ejercicio 3	4
Ejercicios RA2 y RA3	5
Ejercicio 4	5
Ejercicio 5	5
Ejercicio 6	5

Instrucciones

Lee atentamente las siguientes indicaciones.

Te **recomiendo encarecidamente** que leas el ejercicio completo antes de hacer los distintos apartados indicados, ya que te dará ideas de cómo plantear la solución.

Se te ha facilitado una estructura de carpetas con todos los documentos.

Los documentos **HTML** solo podrás modificarlos para indicar tu nombre y apellidos donde se indica (borra el texto "**Nombre y apellidos**" y pon tus datos) y para añadir las referencias a los archivos JS que corresponda.

NO AÑADAS LIBRERÍAS. Si quieres usar alguna función extra añádela en el JS que corresponda.

Para la realización de este examen dispones de **las tres horas de clase**, y el método de entrega es a través de Moodle. Modifica el nombre de la carpeta a:

DWEC.Ex2.apellido.nombre

Y comprímela a un archivo ZIP, RAR o 7z con el mismo nombre

Por ejemplo, para Antonio Sierra, el archivo comprimido (y el de la carpeta) será:

DWEC.Ex2.sierra.antonio


ES IMPORTANTE QUE, a la hora de entregar, guardéis los cambios, le deis a **Enviar tarea** luego confirméis que queréis enviar el trabajo para su evaluación (ver la imagen siguiente)

☒ Confirmando que este trabajo es de elaboración propia, excepto aquellas partes en las que haya reconocido la autoría de la obra o parte de ella a otras personas.

¿Está seguro que quiere enviar su trabajo para que sea evaluado? Una vez enviado ya no podrá realizar modificaciones.

Continuar

Cancelar

En este formulario hay campos obligatorios .

Y confirmar que sale el estado de entrega como **enviado para calificar**:

Estado de la entrega

Enviado para calificar

Si no tienes claro algo del enunciado, pregunta. Para la corrección de este examen será **obligatorio** usar los nombres de funciones y parámetros que se indican.

La puntuación será sobre 10 en cada agrupación de RA y la de cada ejercicio es la siguiente:

RA	2 y 3			4		
Ejercicio	1	2	3	4	5	6
Puntuación Máxima	3	4	3	3	3	4

Se valorará todo el código, pero en caso de que alguna función o elemento no funcione como se indica se usará el **modificador x0,6** al valor de éste.

Ejercicios RA4

Para no complicarte la vida, en los tres ejercicios de clases que contiene este examen solo tendrás que trabajar con dos clases. Siempre las mismas. Éstas son: Trabajador y Directivo; de tal forma que Directivo es una clase que hereda de Trabajador.

El funcionamiento de estas clases siempre será el mismo:

<u>CLASE</u>	<u>TRABAJADOR</u>		
<u>Propiedades</u>			
Nombre	Tipo	Visibilidad	Descripción
_nombreCompleto	String	privado	Defecto: “Sin nombre ni apellidos”
_salario	Number	privado	Defecto: 0
<u>Setters y Getters</u>			
Nombre	Entrada	Salida	Descripción
setNombreCompleto	nombre:string		Si se verifica el <i>nombre</i> recibido con <i>verificarTexto</i> se modificará el valor de la propiedad privada <i>nombreCompleto</i> , de lo contrario no hará nada.
getNombreCompleto		string	Devolverá la propiedad privada <i>nombre</i>
getSalario		número	Devolverá la propiedad privada <i>salario</i>
<u>Métodos</u>			

Nombre	Entrada	Salida	Descripción
toString		cadena	Devuelve una cadena de texto con los datos internos de la instancia siguiendo el siguiente formato: "Trabajador: <i>_nombreCompleto</i> ; Cobra <i>_salario €</i> ", siendo las palabras en cursiva las propiedades del Empleado.
aumentarSalario	aumento:número		Si el <i>aumento</i> recibido es verificado con <i>verificarSalario</i> se aumentará la propiedad privada <i>salario</i> en dicha cantidad.
verificarSalario	salario:número	bool	Si el <i>salario</i> recibido es mayor que cero devolverá true, si es cero o menos devolverá false.
verificarTexto	Texto:string	bool	Si el <i>texto</i> recibido tiene una longitud mínima de 6 caracteres y, además, contiene al menos un espacio devolverá true, en cualquier otro caso false.

CLASE <u>DIRECTIVO [hereda de TRABAJADOR]</u>			
<u>Propiedades</u>			
Nombre	Tipo	Visibilidad	Descripción
_cargo	String	privado	Defecto: "Escogido a dedo"
<u>Setters y Getters</u>			
Nombre	Entrada	Salida	Descripción
setCargo	cargo:string		Si se verifica el <i>cargo</i> recibido con <i>verificarTexto</i> se modificará el valor de la propiedad privada <i>cargo</i> , de lo contrario no hará nada.
getCargo		string	Devolverá la propiedad privada <i>cargo</i>
<u>Métodos</u>			

Nombre	Entrada	Salida	Descripción
toString		cadena	Llamando a la clase madre, devuelve una cadena de texto con el formato "Directivo: <i>_nombreCompleto</i> ; Cobra <i>_salario</i> €; Puesto: <i>_cargo</i> ", siendo las palabras en cursiva las propiedades del directivo.
aumentarSalario	aumento:número		Si el <i>aumento</i> recibido es verificado con <i>verificarSalario</i> se aumentará la propiedad privada <i>salario</i> en dicha cantidad multiplicada por 2.

Ejercicio 1

Desarrolla mediante funciones constructoras la clase Trabajador. Todos los métodos, setters y getters serán funciones externas asociadas a un nombre de método en la **función constructora**.

Dentro de la carpeta *ejercicio1* tienes el html y la carpeta con el archivo específico JS donde desarrollar la clase. Referencia la clase en el html. Vincula en el HTML el JS para poder hacer pruebas.

Ejercicio 2

Desarrolla las clases Trabajador y Directivo mediante **prototipos**.

Dentro de la carpeta *ejercicio2* tienes el html y la carpeta con los archivos específicos JS donde desarrollar las clases. Referencia las clases en el html. Vincula en el HTML los JS para poder hacer pruebas.

Ejercicio 3

Supongamos que quisiese simular una clase parecida a las anteriores mediante **objetos literales**, implementando los datos que me ha devuelto una consulta de una app externa. Estos datos vienen en un array de dos dimensiones que es, básicamente, un array con subarrays que tienen el siguiente formato:

[nombreCompleto, salario, cargo]

Le puedes encontrar en el archivo *ejercicio3.js* con el nombre *datosImportados*.

Usa un bucle, ya que el array puede cambiar de tamaño, para implementar todos los trabajadores del array mediante objetos literales y guardarlo en el array que se te facilita *datosConvertidos*.

Si algún elemento es *undefined* no lo implementes en el objeto literal correspondiente.

Vincula en el HTML el JS para poder hacer pruebas.

Ejercicios RA2 y RA3

Ejercicio 4

Dentro del archivo JS del ejercicio tienes un único array, denominado *array4*, que contiene una serie de valores.

Implementa la siguiente función sort:

Función	Recibe	Devuelve	Descripción
Ordenar	a,b	number	Se usará para ordenar el array en función del tipo de dato, dado prioridad a los string y en caso de ser números a aquellos divisibles por 3. En caso de que los datos sean del mismo tipo y/o divisibles por 3 se ordenará ascendentemente.

Ejercicio 5

Crea la siguiente función

Nombre	generarMatriz
Argumentos de entrada	Dimensio1, Dimension2 Además, sin ser definidos, podrá recibir otros dos argumentos. Todos ellos se espera que sean números enteros mayores o iguales a 1. Por lo que no hay que verificar nada. Además, el usuario introducirá siempre 2 o 4 argumentos. Nunca introducirá 1, 3, 5 o más o ningún argumento.
Salida	Devolverá la matriz que genere
Funcionamiento	Esta función generará un array de dos dimensiones, de tamaño dimension1xdimension2, en el que cada uno de los elementos dentro de la matriz será un número entero aleatorio, y de existir el tercer y cuarto argumento será un número aleatorio entre el tercer y cuarto argumento indicados.

Ejercicio 6

En el HTML vas a encontrar una división (*divSalida*) donde mostrarás todos los mensajes necesarios.

En el archivo JS encontrarás el manejador del load, pero completamente limpio. Tienes libertad absoluta para modificar este load y añadir las funciones que necesites.

Funcionamiento de la aplicación

En la división de salida deberá aparecer una cuenta atrás desde un número obtenido aleatoriamente que estará medio segundo visible y medio segundo invisible, y cada vez que llegue a 0 volverá a sacar un número aleatorio y repetir el proceso.

Siempre que se genere un número aleatorio éste deberá estar entre los valores 2 y 5, ambos inclusive. Cuando lo genere saca dicho valor por la consola.