Examen práctico UD03. Arrays, Funciones y Objetos definidos por el usuario

En este examen se pueden usar todos los recursos vistos hasta la fecha en las unidades 02 y 03.

**Se pide: Valoración del apartado**

1. Definir un objeto Vehículo con las siguientes propiedades: **(1 punto)**

* tipo (que podrá tener los valores:

b (si es una bicicleta),

c (si es un coche),

B (si es un bus),

C (si es un camión)).

* valor (que almacenará el valor en € del vehículo)
* años (que almacenará los años de antigüedad del vehículo)
* propietario (almacenará el nombre del propietario)

Definirlo con argumentos, pero inicializarlos teniendo en cuenta que puede o no recibir el valor inicial al crear un objeto de este tipo. Si no se recibiese algún argumento el objeto creado se inicializará con los siguientes datos por defecto: “c”, 1000, 5, “Lucas”. **(0,5 puntos)**

1. y con los siguientes métodos:

* todos aquellos que sean necesarios para leer sus propiedades. **(0,5 puntos)**
* todos aquellos que sean necesarios para escribir sus propiedades. **(0,5 puntos)**
* uno que me permita devolver el valor que tendrá un vehículo dentro de X años. Este método recibe el % de depreciación por año y el nº de años (X) y devuelve el nuevo valor calculado. Ejemplo: **(1 punto)**

Si un coche vale 1000€ y se deprecia un 10% anualmente, dentro de 5 años valdrá:

Año 1: 1000-10%\*1000=900

Año 2: 900-10%\*900=810

Año 3: 810-10%\*810=729

Año 4: 729-10%\*729=656,1

Año 5: 656,1-10%\*656,1=590,49

Este método devolverá (en el ejemplo 590,49 **(0,5 puntos)**

1. Crear un array literal de 3 elementos donde cada elemento es un objeto de tipo vehículo. **(0,5 puntos)**

Los datos los puedes inventar.

1. Crear un array literal paralelo al anterior donde almacenas los % de depreciación que tiene cada uno de los vehículos. **(0,5 puntos)**
2. Desarrolla una función que reciba ambos arrays, el nº de años y devuelva una cadena de texto con la información a mostrar. Dicha información será un listado del valor futuro de cada vehículo al cabo del nº años recibido como argumento. **(1 punto)**
3. El programa deberá solicitar al usuario el nº de años sobre el que se quiere calcular la depreciación y mostrar mediante un único alert los datos de los vehículos acompañados del valor final una vez aplicada la depreciación (cada vehículo en una línea) **(0,5 puntos)**

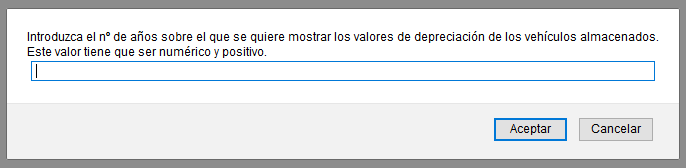
**TOTAL 6,5 puntos**

1. La correcta estructuración del código **(0,75 puntos)**
2. Los comentarios de variables, bloques de código, funciones **(0,75 puntos**)
3. El enlace del JavaScript en la cabecera del documento **(0, 5 puntos)**
4. El control de errores en la introducción de datos (nº de años) **(0,75 puntos)**
5. La redacción de los mensajes informativos en la entrada de datos y salida de información **(0,75 puntos)**

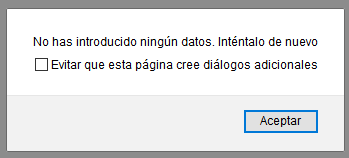
**TOTAL 3,5 puntos**

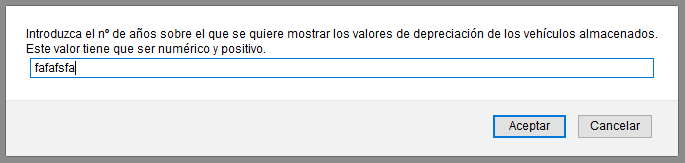
**SUMA TOTAL 10 puntos**

Copias de pantalla de la introducción de datos errónea y mensajes resultantes:

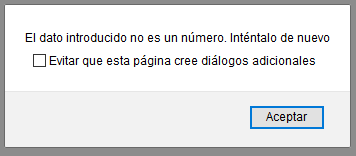


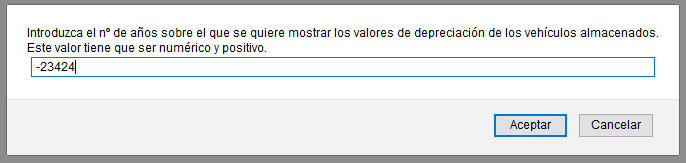
Si se deja en blanco:



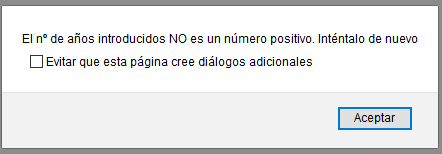


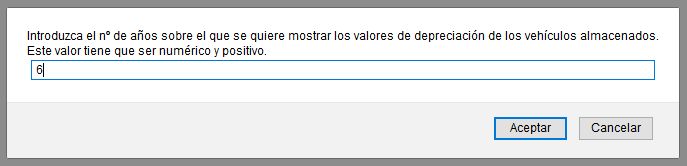
Si se introducen letras:



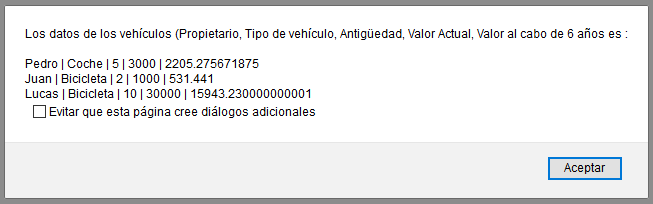


Si se introducen números menores o iguales que 0:





Si se introduce un número positivo:



Hoja de datos para el cálculo con los datos de entrada reflejados en la pantalla anterior y unos % de depreciación de 5, 10 y 10 respectivamente:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **5** | **10** | **10** | **%Depreciación** |
| **Años** | ***3000*** | ***1000*** | ***30000*** | ***Valor Inicial*** |
| 1 | 2850 | 900 | 27000 |  |
| 2 | 2707,5 | 810 | 24300 |  |
| 3 | 2572,125 | 729 | 21870 |  |
| 4 | 2443,51875 | 656,1 | 19683 |  |
| 5 | 2321,34281 | 590,49 | 17714,7 |  |
| 6 | 2205,27567 | 531,441 | 15943,23 |  |
| 7 | 2095,01189 | 478,2969 | 14348,907 |  |
| 8 | 1990,26129 | 430,46721 | 12914,0163 |  |
| 9 | 1890,74823 | 387,420489 | 11622,6147 |  |
| 10 | 1796,21082 | 348,67844 | 10460,3532 |  |
| 11 | 1706,40028 | 313,810596 | 9414,31788 |  |
| 12 | 1621,08026 | 282,429536 | 8472,88609 |  |
| 13 | 1540,02625 | 254,186583 | 7625,59748 |  |
| 14 | 1463,02494 | 228,767925 | 6863,03774 |  |
| 15 | 1389,87369 | 205,891132 | 6176,73396 |  |
| 16 | 1320,38001 | 185,302019 | 5559,06057 |  |
| 17 | 1254,36101 | 166,771817 | 5003,15451 |  |
| 18 | 1191,64296 | 150,094635 | 4502,83906 |  |
| 19 | 1132,06081 | 135,085172 | 4052,55515 |  |
| 20 | 1075,45777 | 121,576655 | 3647,29964 |  |
| 21 | 1021,68488 | 109,418989 | 3282,56967 |  |
| 22 | 970,600635 | 98,4770902 | 2954,31271 |  |
| 23 | 922,070603 | 88,6293812 | 2658,88144 |  |
| 24 | 875,967073 | 79,7664431 | 2392,99329 |  |
| 25 | 832,168719 | 71,7897988 | 2153,69396 |  |