JavaScript – Ejercicios Desarrollando algoritmos 3

Antonio Cebrián

JavaScript – Ejercicios – Desarrollando algoritmos 3

CODE SPACE ACADEMY

**** Información previa para realizar los ejercicios: ****

Todo el trabajo deberá ser realizado en un archivo llamado algorithms3.js, y deberá estar subido al repo training-javascript, de manera pública.

Se recomienda realizar una buena frecuencia de commits, incluyendo siempre mensajes lo más aclaratorios posibles del estado del código en el momento de la confirmación.

En estos ejercicios se debe incluir todo lo aprendido hasta el momento, es decir, estructuras de control, bucles, arrays, objetos y funciones.

En la medida de lo posible tratar de escribir funciones de un único propósito, es decir, una función para resolver un problema específico. Por ejemplo, en lugar de tener una función que genere números aleatorios y los guarde en un array, tener una función específicamente para generar los números. El guardado de números dentro de un array, pudiera ir dentro de otra función.

Antes de comenzar los ejercicios, te recomiendo que crees una cuenta personal de twitter. A continuación accede a la siguiente web y ¡acepta el ***challenge!*** → ***https://www.100daysofcode.com/***

JavaScript – Ejercicios – Desarrollando algoritmos 3

1. **Ejercicio**
   Escribe un programa que pregunte al usuario los límites máximo y mínimo, y genere un array de 20 números aleatorios entre esos valores, ambos incluidos. Luego mostrará el valor más alto y el más bajo dentro del array, con el siguiente formato (por consola):
   - **min value**: nnn
   - **max value**: mmm

2. **Ejercicio**
   Escribe un programa que recoge la hora del sistema, y al cargar la página muestra un saludo (alerta) en función de la hora, teniendo en cuenta los siguientes rangos:
   - **Entre las 6:00 y las 11:59 → “¡Buenos días!”**
   - **Entre las 12:00 y las 20:59 → “¡Buenas tardes!”**
   - **Entre las 21:00 y las 05:59 → “¡Buenas noches!”**

JavaScript – Ejercicios – Desarrollando algoritmos 3

3. **Ejercicio**
   Escribe un programa que genere 10 códigos hexadecimales aleatorios, y escriba en la consola "Hello World!" del color aleatorio generado

4. **Ejercicio**
   Para verificar el DNI, se divide el número entre 23 y el resto se sustituye por una letra que se determina por la siguiente tabla:

RESTO	0	1	2	3	4	5	6	7	8	9	10	11
LETRA	T	R	W	A	G	M	Y	F	P	D	X	B

RESTO	12	13	14	15	16	17	18	19	20	21	22
LETRA	N	J	Z	S	Q	V	H	L	C	K	E

Escribe un programa que almacena un DNI dado por el usuario, y verifica si es un DNI válido, lanzando una alerta "Valid DNI" o "The data entered is wrong".

** **Bonus**: añade un programa que genera DNI válidos aleatorios. **

JavaScript – Ejercicios – Desarrollando algoritmos 3

5. **Ejercicio**

   Escribe un programa que genera matrículas aleatorias, sabiendo que:

   - **Una matrícula válida debe estar compuesta por 4 números y 3 letras.**
   - *Los números pueden ir desde 0000 hasta 9999*
   - *Las letras pueden ir desde BBB hasta ZZZ*
   - *No se permiten vocales*
   - *No se permiten las consonantes O ni Q*

   El número de matrículas aleatorias será a petición del usuario.

6. **Ejercicio**

   Recopila 20 citas aleatorias de una temática a tu elección, y almacénalas en un array. A continuación, investiga las funciones setInterval y setTimeout para generar una cita aleatoria, que se mostrará en consola, cada 10 segundos, durante 2 minutos.

JavaScript – Ejercicios – Desarrollando algoritmos 3

CODE SPACE ACADEMY

7. **Ejercicio**
   Escribe un programa que genera 100 números aleatorios, entre 0 y 500, y los almacena en un array. A continuación filtra todos los números impares, ordenando los pares de mayor a menor

8. **Ejercicio**
   Escribe un programa para jugar a la carta más alta. Para el juego se utilizará la baraja de poker, por lo que:
   - **habrá cuatro arrays, uno por cada palo: (clubs, hearts, spades, diamonds)**
   - dentro de cada array habrán las siguientes cartas, cuyo valor está ordenado de mayor a menor: ACE, KING, QUEEN JACK, 10, 9, 8, 7, 6, 5, 4, 3, 2.

   Se juega contra la banca, con un saldo inicial de 500 €. En cada mano se preguntará al usuario cuanto quiere apostar.

   Si la apuesta del usuario supera su saldo, se le notificará y se le pedirá que apueste de nuevo. Si la apuesta es válida, se generarán 2 cartas aleatorias, se mostrarán por consola, y se indicará una alerta "***You win!***", "***You lose***" o "***Draw***", según corresponda. Continúa en la siguiente página →

JavaScript – Ejercicios – Desarrollando algoritmos

Se sumará o restará el saldo en función del resultado.

Al final de cada mano se le preguntará si quiere seguir jugando, siendo "***y***" la opción para seguir, y "***n***" la opción para retirarse.

En caso de perder todo el saldo, la partida finaliza. En caso de retirarse, se calcula si ha habido beneficios o pérdidas, y se indica mediante una alerta: "**Betting benefits:** XXX €", mostrando las pérdidas con un número negativo, y las ganacias con un número positivo. A continuación, otra alerta con el saldo total: **"Total balance:** YYY €**"**

JavaScript – Ejercicios – Desarrollando algoritmos 3

CODE SPACE ACADEMY

9. **Ejercicio**

   El cifrado César es un tipo de cifrado por sustitución en el que una letra en el texto original es reemplazada por otra letra que se encuentra un número fijo de posiciones más adelante en el alfabeto.

   Por ejemplo, con un desplazamiento de 3, la A sería sustituida por la D (situada 3 lugares a la derecha de la A), la B sería reemplazada por la E, etc. Este método debe su nombre a Julio César, que lo usaba para comunicarse con sus generales.

   Escribe un programa con una función que recibe 2 parámetros, el primero para indicar si hay que cifrar o descifrar, y el segundo bien texto en claro, o bien el texto codificado mediante el cifrado Cesar. El programa mostrará por consola el string resultante, codificado o no, según corresponda.

*** **Bonus:** *es posible que te hayas quedado con ganas de más. Si es así magnífico. Una vez has completado este entrenamiento, crea cuenta en https://www.hackerrank.com/ ¡y* ***happy coding!***

JavaScript – Ejercicios Desarrollando algoritmos 3

Antonio Cebrián