UT4. Programación con funciones, arrays y objetos definidos por el usuario

BOLETÍN DE EJERCICIOS

FUNCIONES

1. Hacer un programa que compruebe si un número es perfecto. Deberá implementarse una función esPerfecto(numero) que devuelva true si lo es.
2. Diseñar un programa que compruebe si un texto contiene sólo caracteres de nuestro alfabeto. Deberá implementarse una función esAlfabetoEspaniol(texto) que devuelva true si lo es.
3. Realizar un programa que calcule el número de cifras de un número. Deberá implementarse una función numCifras(*numero*) que devuelva el número de cifras del mismo. Utiliza el operador spread.
4. Realizar una función que pase una cantidad de Mbyte, Kbytes y bytes a bytes. Probadla en una página.
5. Hacer un programa para generar el siguiente primo a uno dado. Deben usarse funciones.
6. Hacer un programa que sume todos los parámetros pasados como argumentos de entrada en la llamada. El número de argumentos de entrada es desconocido.
7. Escribe una función que reciba como argumento de entrada un array de números y averigue utilizando métodos del objeto Math cuál es el menor y el mayor de ellos.
8. Realizar la función esPalindromo(cadena) de manera recursiva.

ARRAYS

9. Escribe todas las funciones en ES5 y con la notación de función flecha de ES6.

a. Escribe una función llamada **elMenor** que acepte un número variable de parámetros y devuelva el parámetro de menor valor pasado a la función.

b. Escribe una función llamada **colocaEnMedio** que acepte como parámetros dos arrays y devuelva el primer array con todos los valores del segundo array colocados en el centro del primer array.

Ejemplos:

```
placeInMiddle([1,2,6,7],[3,4,5]) // [1,2,3,4,5,6,7]
placeInMiddle([1],[3,4,5]) // [3,4,5,1]
placeInMiddle([1,6],[2,3,4,5]) // [1,2,3,4,5,6]
placeInMiddle([],[2,3,4,5]) // [2,3,4,5]
```

c. Escribe una función llamada **uneArrays** que acepte un número variable de parámetros (cada uno de ellos será un array) y devuelva un nuevo array con todos los parámetros concatenados.

Ejemplos:

```
joinArrays([1], [2], [3]); // [1,2,3]
joinArrays([1,2,3],[4,5,6],[7,8,9]) //[1,2,3,4,5,6,7,8,9]
```

d. Escribe una función llamada sumaArgPares que sume todos los argumentos pares que se pasen a la función.

Ejemplos:

```
sumEvenArgs(1,2,3,4) // 6
sumEvenArgs(1,2,6) // 8
```

10. **Refactoriza el siguiente código usando <u>funciones flecha</u>.**

a. Asegúrate de que la función se llama tripleAndFilter.

```
function tripleAndFilter(arr){
        return arr.map(function(value){
              return value * 3;
        }).filter(function(value){
              return value % 5 === 0;
        })
}
```

b. Asegúrate de que la función se llama doubleOddNumbers.

```
function doubleOddNumbers(arr){
    return arr.filter(function(val){
         return val % 2 !== 0;
    }).map(function(val){
         return val *2;
    })
}
```

c. Asegúrate de que la función se llama bar

```
  function bar(){
        let txt = '';
        for(let i in arguments){
              txt += arguments[i];
        }
        return txt;
  }
```

d.Observando los ejemplos anteriores, repite el ejercicio 9.d, utilizando la programación funcional

11. Dada una cadena leída por teclado, realizar un programa que extraiga los números que aparecen en dicha secuencia e imprima por pantalla dichos números y su suma.

12. Hallar los primeros N primos mediante el algoritmo de Criba de Eratóstenes.

13. Realizar una función que rellene un matriz de orden N de número aleatorios.

14. Realizar un programa que permita introducir 2 matrices (hasta tamaño 3x3), y nos de la opción de sumarlas o multiplicarlas. El programa imprimirá las dos matrices y la matriz resultante (si la hubiera).

15. Averiguar cúal es el número que más y el (o los) que menos se repite(n) en un array.

16. Implementar el algoritmo de ordenación QuickSort.

17. Realizar un script que tome una serie de palabras ingresadas por el usuario (separadas por coma) y almacena esas palabras en un array. Posteriormente, manipule el array para mostrar en una nueva ventana los siguientes datos:

    a. La primera palabra ingresada por el usuario

    b. La última palabra ingresada por el usuario

    c. El número de palabras presentes en el array

    d. Todas las palabras ordenadas alfabéticamente

18. Resolver el problema del cambio (devolución mínima de monedas y billetes) utilizando arrays, evitando la duplicidad de estructuras de control alternativo.

19. Hacer un programa en el que el usuario que introduzca, nombre, apellidos, DNI y fecha de nacimiento separado por comas. Esta entrada de datos se repetirá hasta que el usuario introduzca la cadena vacía. El programa debe guardar los datos en un array bidimensional.

20. Implementar funciones para el ejercicio anterior para imprimir los datos y para buscar una persona por apellidos, por DNI o por edad. ¿cómo podríamos optimizar la búsqueda?

21. Añade al ejercicio anterior las siguientes funciones, utiliza además las funciones creadas anteriormente (utiliza los métodos ya implementados de Array y funciones flecha):

    a. **mayorEdad:** filtrará del array los usuarios mayores de edad e imprimirá sus datos en una nueva ventana.

    b. **menorEdad:** filtrará del array los usuarios menores de edad e imprimirá en una

nueva ventana los días y/o años que le quedan para su mayoría de edad.

c. **modificaDatos:** solicitará qué datos modificar, el dato por el que se va a modificar y el dni del usuario.

d. **eliminaUsuario:** elimina un usuario del array solicitando su dni. Además solicita confirmación antes de eliminar.

MAP y SET

22. Utiliza un map almacenar información sobre módulos impartidos en el IES de la siguiente manera: ("DWECL", “Desarrollo Web en Entorno Cliente”). Añade la información con posterioridad a la creación de la estructura.

    a. Muestra cuántos módulos hay almacenados
    b. Muestra el contenido de la estructura
    c. Devuelve las abreviaturas de todos los módulos guardados
    d. Devuelve el nombre completo de todos los módulos
    e. Consulta si está el módulo "DAW"
    f. Si está, elimínalo.
    g. Ordena alfabéticamente el map según las abreviaturas de los módulos

23. En este ejercicio, de cada módulo se desea guardar su nombre, duración y alumnos matriculados (módulo, duración, numAlumnos). Utiliza la estructura que sea más conveniente.
    a. Comprueba si existe en tu estructura el módulo “DWS” (Servidor) y si es así devuelve el número de alumnos matriculados en dicho módulo.
    b. Calcula el número total de alumnos matriculados en todos los módulos

24. Escribe una función a la que se le pase como parámetro un array y devuelva ese mismo array después de eliminar los elementos repetidos.

OBJETOS DEFINIDOS POR EL USUARIO

25. Crear un objeto Punto con dos coordenadas (x,y) y un método para averiguar el cuadrante en el que está.

26. Crear un objeto Rectángulo con un constructor a partir de dos objetos Punto, con métodos para hallar el perímetro del mismo y su área. Añade también dos métodos para calcular la base y la altura del rectángulo a partir de los puntos.

27. Implementar el ejercicio 20 y 21(sólo imprimir los datos), usando objetos. (P.e. clase Persona)

28. Crear un clase Alumno con su nombre, DNI, ... (objeto Persona), curso y notas de cada módulo. Crear su constructor y un método para imprimir un Alumno, otro que devuelva la nota media y otro para obtener su mejor nota y el nombre del módulo con esa nota (puede tener la misma nota en varios módulos).

29. Crear un objeto Aula que contenga los alumnos de un aula y tenga los siguientes métodos:

    a. buscar un alumno del aula por DNI.

    b. ordenar por nota para un alumno en particular

    c. ordenar el array de alumnos por apellido

    d. imprimir los alumnos de un aula.

30. Usando una implementación de objetos ES6 para guardar la sesión de calificación de un piloto con los siguientes atributos:

```
piloto; // Objeto piloto, contendrá su nombre y escudería.

tiempo; // Contendrá los ms de la mejor vuelta
```

Y teniendo un array de sesiones de calificación, usando sort(); escribir el código necesario para ordenar el array de calificación por:

A. Tiempos.
B. Nombre de piloto.

Añade una función para añadir al array una sesión de calificación nueva, **en caso de que no exista una sesión para ese piloto y en caso de exista, si el tiempo el nuevo tiempo es menor, se modificará el tiempo en la sesión que ya existe en el array** y otra para eliminar del array (hay que comprobar que existe, crea una función para ello).

Crea un archivo para la clase Piloto.