T4. Programación con funciones, arrays y objetos definidos por el usuario

Basado en parte en los ejercicios de C de mi compañera de Mª Asunción Criado.

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.

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 fracciones y de como resultado la fracción simplificada.

7. Diseñar un programa que calcule el número combinatorio $\binom{n}{k}$. Usad funciones y recursividad para el factorial.

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

8. Realizar la función esPalindromo(cadena) de manera recursiva.

9. Realiza un algoritmo recursivo en JavaScript que desordene una palabra pasada como parámetro para crear un ***anagrama*** de la misma: `crearAnagrama("gregoryHouse")` devolverá por ejemplo “Hugeegosorry”

10. [**codeFEST**] Un virus tiene dos formas de propagación:
    a. Salto a la célula sgte.
    b. Salta a la célula sgte a la sgte.

Desarrollar un algoritmo que calcule las combinaciones de propagación posibles para que el virus llegue a una distancia determinada, suponiendo que sólo se transmite en una dimensión y hacia la derecha.
Ej. Para llegar a distancia 3, lo patrones posibles son “ab”, “ba” y “aaa”.

ARRAYS

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. Realizar un programa lee un texto y lo imprime en clave. Escribirlo en clave no es más que sumar una cantidad al número en código ASCII que corresponde a cada carácter. Esta cantidad o clave se ha de teclear en cada ejecución del programa.

17. Implementar el algoritmo de ordenación QuickSort.

18. 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 arrays.

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

20. 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 alfabeticamente

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

OBJETOS DEFINIDOS POR EL USUARIO

20. [B] Crear un objeto Punto con dos coordenadas (x,y) y un método para averiguar el cuadrante en el que está. Realizar otro método para averiguar la posición relativa de dos puntos.

21. Crear un objeto Rectángulo con un constructor a partir de dos puntos, con métodos para hallar el perímetro del mismo y su área.

22. Resolver el ejercicio 6 mediante objetos.

23. Implementar el ejercicio 18 y 19 usando mediante objetos. (P.e. el objeto Persona)

24. Crear un objeto Alumno con su nombre, DNI, ... (objeto Persona), curso, notas de cada módulo, y nota media. Crear su constructor y una función para imprimir un Alumno.

25. Crear un objeto Aula que contenga los alumnos de un aula y tenga un método de ordenación por nota basado en Qsort y otro para imprimir los alumnos de un aula.

26. Usando una implementación de objetos para guardar la sesión de calificación de un piloto como la siguiente:

```
function SesionCalificacion (){
      this.piloto;      //  Objeto  piloto,  contendrá  su  nombre  y
escudería.
      this.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.

NOTA: Es aconsejable programar un generador de sesiones y un método de imprimir sesiones para probar que las ordenaciones se realizan correctamente.

27. Crear un fichero cola.js que contenga el Tipo Abstracto de Datos (T.A.D.) cola y estos métodos:

a. Cola() (constructor),
b. push(dato) -> Añade dato a la cola.
c. pop() -> Devuelve el primer dato de la cola y lo elimina.

28. Crear un fichero arbolbinario.js que contenga el Tipo Abstracto de Datos (T.A.D.) árbol binario y estos métodos:

a. ArbolBinario(dato) (constructor),
b. AniadeHijoIzq(dato, datoNuevo)
c. AniadeHijoDcho(dato, datoNuevo)
d. ImprimeArbol() (Requerirá recorrer el árbol)

28. Crear un fichero piladoble.js que contenga el Tipo Abstracto de Datos (T.A.D.) PilaDoble y estos métodos:
    a. **`PilaDoble`**`()` -> Constructor.
    b. **`aniadeAPila1`**`(dato)` -> Añade el dato a la pila 1.
    c. **`aniadeAPila2`**`(dato)` -> Añade el dato a la pila 2.
    d. **`extrae`**`()` -> Devuelve y elimina el primer dato de una de las pilas. La sgte. operación de extracción se hará de la otra pila, salvo que esté vacía. Si ambas pilas están vacías se devolverá undefined.

29. Crear un fichero monticulo.js que contenga el Tipo Abstracto de Datos (T.A.D.) montículo (binary heap) y estos métodos:
    a. Monticulo()
    b. insertar(dato) -> Inserta dato en el montículo.
    c. borrarRaiz() -> Devuelve el dato de la raíz y la elimina reconstruyendo el montículo.