Examen JS/NodeJS desarrolla.software
Primavera 2021

1. Defina los tipos de datos **primitivos** de JavaScript. Declare una variable para cada uno de ellos.

2. Defina los tipos de datos **complejos** de JavaScript. Declare una variable para cada uno de ellos.

3. Defina una función llamada **MostrarMensaje**. Dicha función toma de parámetro 3 valores. Un código del tipo String, un número y un texto.
   La idea de esta función es enviar los 3 valores en el siguiente orden: **(code, number, description)** y mostrar un mensaje de la siguiente manera en la consola, pero usando los parámetros en lugar de estos 3 valores de ejemplo:

   user-not-found: 404
   The requested user could not be found

4. Escriba el bloque de código necesario para mostrar los siguientes resultados:
    a. Se tienen tres datos: nombreUsuario, tipoUsuario y operación.
       Dados los 3 valores, comprobar lo siguiente:
        i. Si el tipo de usuario es **'admin'**, mostrar en consola **"Permisos otorgados para: nombreUsuario"** para cualquier tipo de operación.
        ii. Si el tipo de usuario es **'colaborador'**, mostrar en consola **"Permisos otorgados"** solo si el tipo de operación es **"Lectura", "Edición" y "Mover"**. Si el tipo de operación es **"Creación"** ó **"Eliminar"**, mostrar el mensaje **"Permisos denegados para: nombreUsuario"**.
        iii. Si el tipo de usuario es **'regular'** y la operación es **"Lectura"**, mostrar **"Permisos otorgados para: nombreUsuario"**, si no, mostrar **"Permisos denegados para: nombreUsuario"**.

5. Dada una lista de números llamada **numbers**, recorrerla y mostrar en consola el número y si es par o impar. Ejemplo de salida en consola:

*Número: **6** es **par***
*Número: **7** es **impar***

6. **Declare un objeto** que represente la información de una transacción bancaria.
Una transacción bancaria se describe por tener el **número de cuenta que paga**, el **número de cuenta de destino**, la cantidad de **dinero transferido**, el nombre del **titular de la cuenta que paga**, el nombre del **titular de la cuenta destino** y la **fecha/hora de operación**.

7. Cree una **lista que represente compañías** de venta de dispositivos móviles. Mínimo cada compañía debe tener la siguiente información: **nombre de la compañía, país de origen, modelos principales** que vende (Mínimo 2 datos en este campo. Tip: es un array).

8. Dada la siguiente lista: **['A', 'B', 'C']**, añada después de **'C'** la letra **'D'**. Lo debe añadir de las dos posibles formas de hacerlo: **Por índice** y **con la función** que permite agregar elementos al final de una lista.

9. De la **lista de compañía del ejercicio 7**, tome el **último elemento** y muestre en la consola su información. **NO DEBE USAR EL ÍNDICE EXACTO**. Utilice lógica para determinar siempre el último valor de la lista. Por ejemplo, si en este momento hay 4 compañías, el último es 3, pero **NO debe poner companies[3]**, busque una forma diferente de obtener el índice 3.
   Esta información debe venir detallada. Ejemplo:

   Nombre de la compañía
   País: país de origen
   Modelos: Modelo A, Modelo B, Modelo C, etc.

10. Cree una función llamada **GenerarObjeto** que reciba de parámetro 3 datos: **nombre, apellidos y edad**.
    La función debe **retornar un objeto** con la siguiente estructura:

```
{
    name: ----,
    lastName: ----,
    age: ----,
}
```

11. Cree un **servidor** básico de **express**, con un **único endpoint** llamado **'/hello'**. Dicho endpoint será el método **HTTP – GET** y regresará un mensaje diciendo **"Hola!"**

12. Escriba el bloque de código necesario para hacer que el archivo **utils.js exporte un módulo** de **NodeJS**. Dicho módulo debe exportar lo siguiente:
    a. Una función llamada **"checkYear"** que recibirá de parámetro un **número** que represente un año y retornará true solo si el año es mayor o igual a 1995, y false si es menor a 1995.

13. Escriba el código necesario para **crear una cookie** en el navegador del cliente que se llama **SPGG** con el valor **"desarrolla.software"**.

14. Defina la diferencia entre: **query, params y body** en las peticiones **HTTP de express**. Ponga un ejemplo sencillo de cada uno (O sea, un endpoint y qué datos se le pasan a esa llamada, **NO CÓDIGO JS**, solo la llamada como si se hiciera desde PostMan)

15. Escriba la diferencia de los operadores de **"igualdad"** y de **"identidad"** en JS junto con un ejemplo. Hacer lo mismo con los operadores de **"no igual"** y **"no idéntico"**.

16. Escriba el código necesario para seleccionar un posible caso de un valor. El valor que se pasa es un código de número y tiene los posibles valores:
    a. Si el código es 15, mostrar "Error de escritura".
    b. Si el código es 22, mostrar "Archivo no encontrado".
    c. Si el código es 320, mostrar "No se puede acceder a la unidad".
    d. Si el código es 7890, mostrar "El archivo es demasiado grande para la unidad".
    e. Si el código es 871, mostrar "El archivo está corrupto".
    f. Si el código es 12, mostrar "El archivo está bloqueado".

17. Escriba el código para un **ciclo** que vaya **sumando de 2 en 2** a la variable **"contador"**. Este ciclo termina cuando el **contador es mayor o igual a 500**.

18. Escriba cómo buscar un dato en el **esquema** de **Mongoose**: **Company**. Debe buscar un solo elemento en la colección **companies**, donde el **identificador** de la compañía sea **"Softtek Servicios Corporativos S.A. de C.V"**. (Tip: Esto es muy similar a lo de User.findOne…).

19. Escriba qué significa **CRUD** y su equivalente de cada sigla en **métodos HTTP**.