	UNIVERSIDAD AUTÓNOMA "TOMÁS FRÍAS" **CARRERA DE INGENIERÍA DE SISTEMAS**	
	ESTUDIANTE:	
MATERIA: Seminario de Sistemas (SIS-719)		
DOCENTE: Ing. Ditmar Castro		**PRACTICA N.º:** 1
AUXILIAR: Univ. Edson Olmedo Copa		**GRUPO:** 1

PROBLEMAS DE PROGRAMACION JAVASCRIPT

INSTRUCCIONES:

- Crear un repositorio en GitHub con su nombre completo y al final agregue “P1Aux”
- Presentar el link a su repositorio en formato PDF con la cabecera de este documento al inicio de su tarea, llenando su nombre completo y al lado su CI.
- Agregar un comentario con el numero de la tarea realizada para cada una de las preguntas.

PRESENTACION.

Debe subir el archivo PDF a la plataforma Virtual de la Universidad (Moodle.uatf).

DESARROLLO DE LA PRACTICA.

Resolver los siguientes ejercicios:

1. Realiza un script que pida la edad y si es mayor de 18 años indica que ya puede conducir.
2. Realiza un script que pida números hasta que se pulse “cancelar”. Si no es un número deberá indicarse con un «alert» y seguir pidiendo. Al salir con “cancelar” deberá indicarse la suma total de los números introducidos.
3. Realiza un script que pida por teclado 3 edades y 3 nombres e indique el nombre del mayor.
4. Genera 3 números aleatorios entre 1 y 99 pero que no se repita ninguno.
5. Realiza un script que cuente el número de vocales que tiene un texto.
6. Pedimos una cadena de texto que sabemos que puede contener paréntesis. Realiza un script que extraiga la cadena que se encuentra entre los paréntesis. Ejemplo: Si escribimoe el texto “Hola (que) tal” se mostrará “que”. Si no existe el signo “(“ mostrará una cadena vacía y si existe el signo “(“pero no el signo “)” mostrará desde el primer paréntesis hasta el final.
7. Realiza un script que pida una cadena de texto y la devuelva al revés. Es decir, si tecleo “hola que tal” deberá mostrar “lat euq aloh”.
8. Realiza un script que muestre la serie de fibonacci hasta un número entre 1 y 100 pedido por teclado.
9. Realiza un juego de Piedra Papel Tijera contra el Pc.
10. Crea una función que reciba 2 parámetros, precio e iva, y devuelva el precio con iva incluido. Si no recibe el iva, aplicará el 21 por ciento por defecto.
11. Crea una función validar() para validar la entrada de datos de un formulario
    a. Para evitar que se envíe un formulario varias veces conviene deshabilitar el botón de envío tras enviarlo una vez. Escribe un script para gestionar el envío del formulario:
    b. Deshabilita el botón "Enviar"
    c. Cambia el mensaje que muestra el botón de "Enviar" a "Enviando…"
    d. Cuando se haya enviado genera una página nueva indicando que se ha enviado correctamente y muestra la información que se ha enviado.
12. Realiza un script que muestre un reloj en pantalla con fecha y hora y que se actualice cada segundo. Función setTimeout().
13. Programa una función que dada una fecha valida determine cuantos años han pasado hasta el día de hoy.
14. Programa una función que dada una cadena de texto cuente el numero de vocales y consonantes.
15. Programa una función que valide que un texto sea un nombre valido. Entrada(‘edson’) devolverá ‘verdadero’.
16. Programa una función que dado un arreglo de elementos, elimine los duplicados:
    entrada([‘5’, ’25’, ‘10’, ‘s’, ‘5’, ‘a’, ‘a’]) devolverá ([‘5’, ‘25’, ‘10’, ‘s’, ‘a’]).

17. Programa una función que dado un array de números devuelva un objeto con dos arreglos el primero con los números pares y en el segundo con los números impares.

18. **Ayuda al Elfo a listar los regalos.**

Te ha llegado una carta ✉ con todos los regalos que debes preparar. El tema es que es una cadena de texto y es muy difícil de leer 😱. ¡Menos mal que han puesto cada regalo separado por espacio! (aunque **ten cuidado**, porque al ser niños, igual han colado más espacios de la cuenta)
Encima nos hemos dado cuenta que algunas palabras vienen con un _ delante de la palabra, por ejemplo _playstation, que significa que **está tachado y no se tiene que contar.**
Transforma el texto a un objeto que contenga el nombre de cada regalo y las veces que aparece. Por ejemplo, si tenemos el texto:

```
const carta = 'bici coche balón _playstation bici coche peluche'
```

Al ejecutar el método debería devolver lo siguiente:

```
const regalos = listGifts(carta)

console.log(regalos)
/*
{
  bici: 2,
  coche: 2,
  balón: 1,
  peluche: 1
}
*/
```

Ten en cuenta que los tests pueden ser más exhaustivos... 😝 **¡Cuidado con contar espacios vacíos!**

19. **Buscando en el almacen:**

Mi amigo Dani está trabajando en una tienda y con la llegada de las navidades tiene el almacén hecho un desastre y no encuentra nada.
Vamos a crear una función que recibe dos parámetros: un objeto que define el almacén y el producto que buscamos.
La función debe devolver un booleano que indique si se encuentra el string como valor en algún nivel del objeto.
Veamos unos ejemplos:

```
const almacen = {
  'estanteria1': {
    'cajon1': {
      'producto1': 'coca-cola',
      'producto2': 'fanta',
      'producto3': 'sprite'
    }
  },
  'estanteria2': {
    'cajon1': 'vacio',
    'cajon2': {
      'producto1': 'pantalones',
      'producto2': 'camiseta' // <- ¡Está aquí!
    }
  }
}

contains(almacen, 'camiseta') // true
```

```
const otroAlmacen = {
  'baul': {
    'fondo': {
      'objeto': 'cd-rom',
      'otro-objeto': 'disquette',
      'otra-cosa': 'mando'
    }
  }
}

contains(otroAlmacen, 'gameboy') // false
```

Ten en cuenta que la tienda es enorme. Tiene diferentes almacenes y, como has visto en los ejemplos, cada uno puede tener diferentes organizaciones. **Lo importante es buscar que el producto está en los almacenes.**

20. **En busca del reno perdido.**

¡Hemos perdido a un reno y falta poco más de una semana para Navidad! 😱
Lo peor es que son tantos que no sabemos cuál es el que nos falta... ¡Qué lío! A ver, Elfon Musk ha hecho inventario y nos pasa un array con los ids de cada reno.
👍 Lo bueno: los ids son números que pueden ir del 0 al 100, no están repetidos y sólo se ha perdido un reno.
👎 Lo malo: la lista está desordenada y podría faltar el último...
Necesitamos una función que al pasarle la lista de ids de renos nos diga inmediatamente cuál es el que falta:

```
missingReindeer([0, 2, 3]) // -> 1
missingReindeer([5, 6, 1, 2, 3, 7, 0]) // -> 4
missingReindeer([0, 1]) // -> 2 (¡es el último el que falta!)
missingReindeer([3, 0, 1]) // -> 2
missingReindeer([9, 2, 3, 5, 6, 4, 7, 0, 1]) // -> 8
missingReindeer([0]) // -> 1 (¡es el último el que falta!)
```

Parece fácil con una complejidad de O(n)... ¿crees que podrías hacerlo mejor?