Normas generales para la práctica

Condiciones de entrega

- Se dispone de 3 sesiones para realizar la práctica. Se entregarán en la fecha indicada. No se admitirán ejercicios entregados después de esa fecha.
- La entrega de todas las actividades se hará a través de GitHub y Aules.
- En GitHub, al repositorio LM subirás un directorio que deberá nombrarse con el nombre y primer apellido del alumno seguido de la frase “-práctica1-UT5”. El nombre y los apellidos deben ir separados por un guión. En aules el enlace a ese directorio del repositorio.

Condiciones de corrección

- Las actividades se deben realizar con un editor (Visual Studio Code por ejemplo)
- Se deben entregar los ficheros .html y .js que se generen.
- Si se detecta copia en alguna actividad se suspenderá automáticamente la unidad de didáctica a todos los alumnos implicados.
- Si se detecta copia de alguna página web de internet u otro recurso, automáticamente se suspenderá la actividad copiada.

Calificación

- Existen tres actividades. Todas tienen la misma puntuación.
- Las actividades se puntuarán dentro del apartado de procedimientos que es un 10% de la nota de la unidad.

Ejercicio 1. Funciones y parámetros

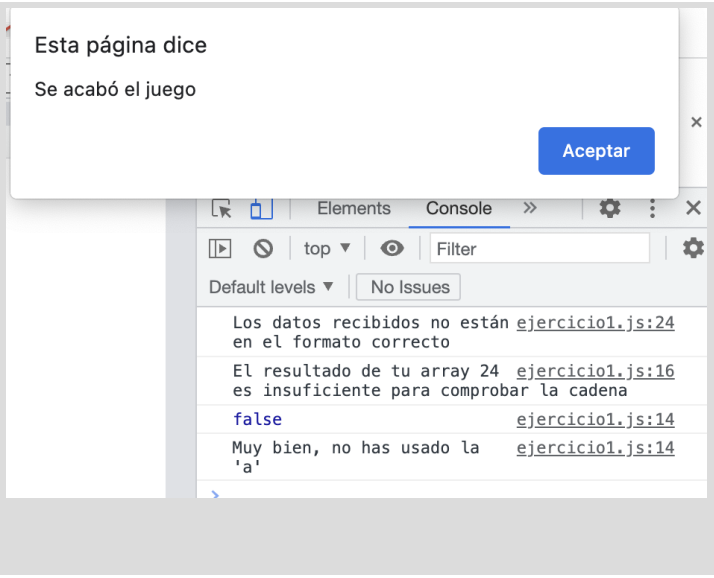
Descripción

- Crea una función que reciba una cadena,un booleano, una función y un array numérico. Si el tipo de algún parámetro no es el esperado debes imprimir un error (suponemos que si nos pasan un array sí que en todas sus posiciones habrá un dato numérico). Si es lo esperado, si el valor del booleano es true, recorre el array con la estructura foreach. Si el producto de todos los ítems en mayor a 100 entonces comprueba si en la cadena hay alguna “a”. Si hay alguna a saca por pantalla un mensaje diciendo que la “a” no está permitida. Si no la hay, saca por pantalla un mensaje diciendo, “Muy bien no has usado la ‘a’”. En caso de que el producto del array fuera menor de 100 informa al usuario: “El resultado de tu array es insuficiente para comprobar la cadena”. Si el valor del booleano es false entonces ejecuta la función recibida por parámetro.

Ejemplos de llamadas

```
Ejercicio1()
Ejercicio1("hola mundo",true,[1,2,3,4],()=>{alert("Se acabó el juego")})
Ejercicio1("Sí",true,[10,20,30,40],()=>{alert("Se acabó el juego")})
Ejercicio1("Sí",false,[10,20,30,40],()=>{alert("Se acabó el juego")})
```

Salida



.

Ejercicio 2. Arrays

Descripción

- Crea una función llamada verAsignaturas. Esta función va a recibir un número indeterminado de alumnos. De cada alumno va a recibir un array. En ese array estará almacenado el nombre, el curso y las asignaturas de las que está matriculado (una asignatura en cada posición). Saca por pantalla el nombre del alumno – el curso – asignaturas: y el nombre de las asignaturas separadas por un /. Si el número de datos de alumnos es 0 debes mostrar la cadena “No hay datos para mostrar”.Debes usar el operador rest, la desestructuración de arrays y el código lo más compacto posible.

Ejemplos de llamadas

```
Ejercicio2(["Sara", "DAMA", "Programación", "ED"],["Martín", "DAMB", "Programación", "LM", "ED", "BBDD", "FOL", "SI"],["Emma", "ASIR","ISO","BBD
Ejercicio2(["Álvaro","Semi","BBDD"])
Ejercicio2()
```

Salida

```
Sara-DAMA-asignaturas:Programación/ED                    ejercicio2.js:10
Martín-DAMB-asignaturas:Programación/LM/ED/BBDD/FOL/SI   ejercicio2.js:10
Emma-ASIR-asignaturas:ISO/BBDD/LM                        ejercicio2.js:10
Álvaro-Semi-asignaturas:BBDD                             ejercicio2.js:10
No hay datos para mostrar                                ejercicio2.js:13
>
```

.

Ejercicio 3. Arrays y funciones. Foreach

Descripción

- Crea dos arrays con nombres de personas. Busca si los nombres del primer array están todos en el segundo, si no hay ninguno o si hay alguno. Usa foreach.¿Se te ocurre otra solución?

Ejemplos de llamadas

```
/*Array original*/
Ejercicio4( ["Elisabet Ricart Monreal","María Del Carmen Sedano-Rocamora","Roldán Alvarado","Leocadio de Pera","Isaac Talavera Luna"], ["Elisabe
/*Ninguno incluído*/
Ejercicio4(["Lina Armida Machado Iglesias","Apolonia Santiago Buendía","Poncio Cobo Herrera","Rafaela Seco Cañas","Fulgencio Alarcón Lloret"
],["Elisabet Ricart Monreal","María Del Carmen Sedano-Rocamora","Roldán Alvarado","Leocadio de Pera","Isaac Talavera Luna"])
/*Alguno incluido*/
Ejercicio4(["Elisabet Ricart Monreal","Poncio Cobo Herrera","Isaac Talavera Luna"],["Elisabet Ricart Monreal","María Del Carmen Sedano-Rocamora"
```

Salida

```
Personas: Elisabet Ricart Monreal/María Del Carmen Sedano-Rocamora/Roldán Alvarado/Leocadio de Pera/Isaac Talavera Luna
Array donde buscar: Elisabet Ricart Monreal/María Del Carmen Sedano-Rocamora/Roldán Alvarado/Leocadio de Pera/Isaac Talavera Luna
Todos están incluidos
Personas: Lina Armida Machado Iglesias/Apolonia Santiago Buendía/Poncio Cobo Herrera/Rafaela Seco Cañas/Fulgencio Alarcón Lloret
Array donde buscar: Elisabet Ricart Monreal/María Del Carmen Sedano-Rocamora/Roldán Alvarado/Leocadio de Pera/Isaac Talavera Luna
No hay ninguno incluido
Personas: Elisabet Ricart Monreal/Poncio Cobo Herrera/Isaac Talavera Luna
Array donde buscar: Elisabet Ricart Monreal/María Del Carmen Sedano-Rocamora/Roldán Alvarado/Leocadio de Pera/Isaac Talavera Luna
Alguno está incluido
```

Ejercicio 4. Arrays y funciones.Map

Descripción

Teniendo un array con el nombre y el curso de cada alumno, crea un array nuevo al que incorpores asignaturas a las que esté matriculados (aunque no sea lo más óptimo, pregúntalas con prompt separadas por -).

Ejemplos de array

```
let alumnos = [
    { nombre: 'Juan', edad: 25},
    { nombre: 'Ana', edad: 30},
    { nombre: 'Pedro', edad: 40}
  ]
```

Salida