PEC 1: Conceptualización e investigación de usuarios

NOTA: Los programas deben tener en las primeras líneas la asignación de las variables que contengan los valores de los enunciados, de manera que cambiando estos valores los resultados del programa cuando se ejecute sean correctos en cualquier caso. Los resultados deben mostrarse por consola.

2. Actividad 2

Comprobad si estos fragmentos de código *JavaScript* muestran un mensaje de error cuando se ejecutarse. Decid qué mensaje muestran y por qué:

Resolución del ejercicio:

1.

```
var x;

try {
x = y - 1;
}
catch(err) { console.log(err.name); }
```

1.

```
var x; // Se define la variable X

try {

x = y - 1; // Se intenta ejecutar esta operación pero con una variable no definida (y) lo que genera un error

}

catch(err) { console.log(err.name); } // El catch(err) capturael error generado y console.log(err.name) muestra el nombre del error en consola
```

```
var x; // Se define la variable X

try {

x = y - 1; // Se intenta ejecutar esta operación pero con una variable no definida (y) lo que genera un error

}

catch(err) { console.log(err.name); } // El catch(err) capturael error generado y console.log(err.name) muestra el nombre del error en consola
```

```
ReferenceError
```

var x; // Se define la variable X

try {

x = y - 1; /*Se intenta ejecutar esta operación, pero con una variable no definida (y), lo que genera un error*/

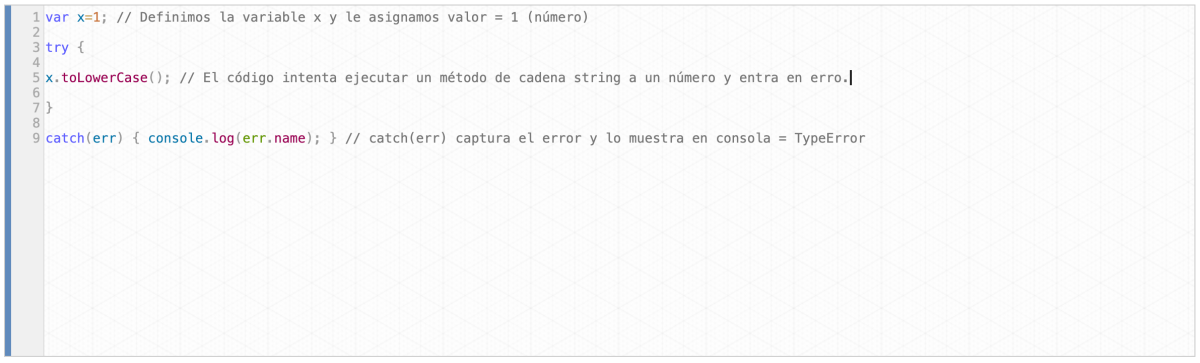
}

catch(err) { console.log(err.name); } /*El catch(err) captura el error generado y console.log(err.name) muestra el nombre del error en consola*/

2.

2.
```
var x=1;
try {
x.toLowerCase();
}
catch(err) { console.log(err.name); }
```

```
1  var x=1; // Definimos la variable x y le asignamos valor = 1 (número)
2
3  try {
4
5  x.toLowerCase(); // El código intenta ejecutar un método de cadena string a un número y entra en erro.
6
7  }
8
9  catch(err) { console.log(err.name); } // catch(err) captura el error y lo muestra en consola = TypeError
```



var x=1; // Definimos la variable x y le asignamos valor = 1 (número)

try {

x.toLowerCase(); // El código intenta ejecutar un método de cadena string a un número y entra en error

}

catch(err) { console.log(err.name); } // catch(err) captura el error y lo muestra en consola = TypeError

3.

```
3.
   var x=0.9999;
   try {
   console.log(x.toFixed(1000));
   }
   catch(err) { console.log(err.name); }
```

3.

```
var x=0.9999; // Definimos la Variable X y le asignamos el valor 0.999

try {

console.log(x.toFixed(1000)); /*El código intenta ejecutar y mostrar en pantalla el valor de X pero ajustando los decimales a un valor muy amplio
                               ¿Qué dice la documentación?
                               El método toFixed() formatea un número usando notación de punto fijo.
                               Sintaxis: numObj.toFixed([digitos])
                               El número de digitos que aparecen después del punto decimal; este puede ser un valor entre 0 y 20
                               Si digits es demasiado pequeño o demasiado grande. Los valores entre 0 y 20, inclusive, no causarán un error
                               tipo RangeError
                              */
}

catch(err) { console.log(err.name); } // catch (err) captura el error y lo muestra en cosonla
```

```
var x=0.9999; // Definimos la Variable X y le asignamos el valor 0.999

try {

console.log(x.toFixed(1000)); /*El código intenta ejecutar y mostrar en pantalla el valor de X pero ajustando los decimales a un valor muy amplio
                               ¿Qué dice la documentación?
                               El método toFixed() formatea un número usando notación de punto fijo.
                               Sintaxis: numObj.toFixed([digitos])
                               El número de digitos que aparecen después del punto decimal; este puede ser un valor entre 0 y 20
                               Si digits es demasiado pequeño o demasiado grande. Los valores entre 0 y 20, inclusive, no causarán un error tipo Ra
                               */
}

catch(err) { console.log(err.name); } // catch (err) captura el error y lo muestra en cosonla
```

```
RangeError
```

var x=0.9999; // Definimos la variable X y le asignamos el valor 0.999

try {

console.log(x.toFixed(1000)); /*El código intenta ejecutar y mostrar en pantalla el valor de X pero ajustando los decimales a un valor muy amplio.
¿Qué dice la documentación?
El método toFixed() formatea un número usando notación de punto fijo.
Sintaxis: numObj.toFixed([digitos])
El número de digitos que aparecen después del punto decimal; este puede ser un valor entre 0 y 20
Si digits es demasiado pequeño o demasiado grande. Los valores entre 0 y 20, inclusive, no causarán un error tipo RangeError*/

}

catch(err) { console.log(err.name); } // catch (err) captura el error y lo muestra en consola

4.

```
4. try {
    varr x=1;
   }
   catch(err) { console.log(err.name); }
```

```
try {

varr x=1; // Se intenta definir la variable x pero utilizando mal la palabra reservada "var" que tiene una "r" extra

}
catch(err) { console.log(err.name); } // El catch(err) captura el error y err.name muestra "Unexpected identifier 'x'"
```


```
Unexpected identifier 'x'
```

try {

varr x=1; // Se intenta definir la variable x pero utilizando mal la palabra reservada "var" que tiene una "r" extra

}

catch(err) { console.log(err.name); } /*El catch(err) captura el error y err.name muestra "Unexpected identifier 'x'"*/