Contenido de este curso

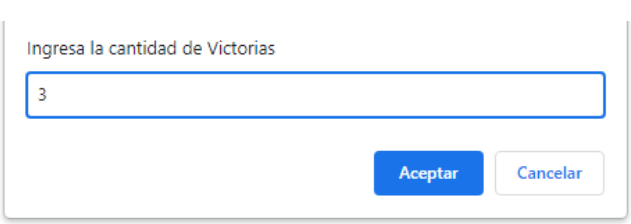
- Condicionales
- Iteraciones (ciclos While y Ciclos For)
- Interacción con el usuario
  - Conectando HTML con JavaScript
  - Botones y Cajas de Texto
- Arrays o Listas
- Chrome será nuestro compilador
- Sublime nuestro editor de texto

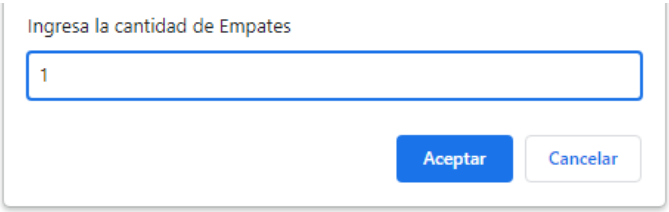
En Java Script debemos tomar en cuenta cuando solicitamos al usuario y guardamos información ya que cuando utilizamos **prompt** (se guarda como string) y JS lo interpreta de la siguiente manera: texto * numero = "multiplica" / texto + numero = "concatena"

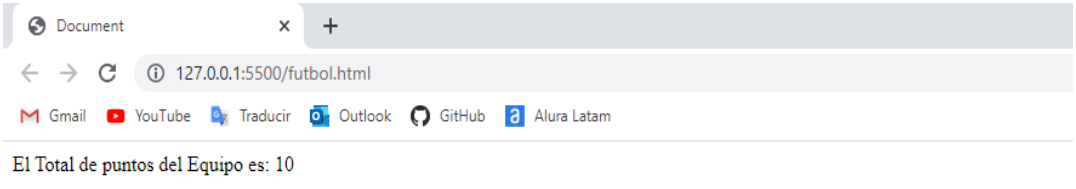
```
<body>
    <div>
        <script>
            function saltarLinea() {
                document.write("<br><br>");
            }

            function imprimir(frase) {
                document.write(frase);
                saltarLinea();
            }

            var victorias = prompt("Ingresa la cantidad de Victorias");
            var empates = prompt("Ingresa la cantidad de Empates");

            // texto * numero = "multiplica"  /  texto + numero =  "concatena"
            // (victorias * 3)       + empates  = resultado
            //      "3"   * 3  => 9  +   "1"    =   "91"

            var puntosTotal = (victorias * 3) + empates;

            imprimir("El Total de puntos del Equipo es: " + puntosTotal);

            imprimir(("10"*2)); // texto * numero = "multiplica"
            imprimir(("10"+2)); // texto + numero =  "concatena"

        </script>
    </div>
</body>
```

ParseInt() .- permite cambiar variables de texto a numéricas.

```
<body>
    <!-- <div>⋯
    <div>
        <script>
            function saltarLinea() {
                document.write("<br><br>");
            }

            function imprimir(frase) {
                document.write(frase);
                saltarLinea();
            }

            var victorias = parseInt(prompt("Ingresa la cantidad de Victorias"));
            var empates = parseInt(prompt("Ingresa la cantidad de Empates"));

            var puntosTotal = (victorias * 3) + empates;

            imprimir("El Total de puntos del Equipo es: " + puntosTotal);

        </script>
    </div>
</body>
```







Total de invitados de la fiesta

Nuestro amigo Jorge creó un programa simple para calcular el total de invitados de su fiesta, el programa tiene que sumar el total de invitados normales con el total de invitados VIPs. Ciertamente podría usar la calculadora, pero le pareció relevante ejercer su conocimiento para diseñar un programa. Entonces, Jorge escribió lo siguiente:

```
<meta charset="UTF-8">

<script>

    function saltarLinea() {
        document.write("<br>");
    }

    function imprimir(frase) {
        document.write(frase);
        saltarLinea();
    }

    var invitados = prompt("Número de invitados");
    var vips = prompt("Número de invitados VIP's");

    var total = invitados + vips;

    imprimir("El total de invitados es " + total);

</script>
```

Observa que usa la función prompt() para leer la cantidad de invitados e invitados vips, por lo tanto está realizando la concatenación de dos números leídos por el teclado, no está realizando la suma.

¿Cuál de las opciones de abajo tiene el código correctamente para calcular el total de invitados?

A

```
<meta charset="UTF-8">

    <script>
        function saltarLinea() {
            document.write("<br>");
        }

        function imprimir(frase) {
            document.write(frase);
            saltarLinea();
        }

        var invitados = parseInt(prompt("Número de invitados"));
        var vips = prompt("Número de invitados VIP's");

        var total = invitados + vips;

        imprimir("El total de invitados es " + total);

    </script>
```

B

```
<meta charset="UTF-8">

    <script>
        function saltarLinea() {
            document.write("<br>");
        }

        function imprimir(frase) {
            document.write(frase);
            saltarLinea();
        }

        var invitados = prompt("Número de invitados");
        var vips = prompt("Número de invitados VIP's");

        var total = parseInt(invitados + vips);
        imprimir("El total de invitados es " + total);

    </script>
```

C

```
<body>
    <h3>Total de Invitados</h3>
    <div>
        <meta charset="UTF-8">

        <script>
            function saltarLinea() {
                document.write("<br>");
            }

            function imprimir(frase) {
                document.write(frase);
                saltarLinea();
            }

            var invitados = parseInt(prompt("Número de invitados"));
            var vips = parseInt(prompt("Número de invitados VIP's"));

            var total = invitados + vips;

            imprimir("El total de invitados es " + total);

        </script>
    </div>
</body>
```

¡Correcto! Como la función `prompt` siempre transforma todo lo que digitamos en el formato del texto, o sea, como un `string`, es una buena práctica siempre convertir el valor digitado para el numero cuando nuestra intención es leer un número. Analicemos esta instrucción: `var invitados = parseInt(prompt("Número de invitados"));`
La función prompt retorna lo que digitamos como `string` y ese retorno es pasado para `parseInt`. Y es esta función que recibe un `string` y la convierte en un número. Ahora, como tenemos dos números vinculados, no sucederá más una concatenación, y sí una suma.

D

```
<meta charset="UTF-8">

  <script>
      function saltarLinea() {
          document.write("<br>");
      }

      function imprimir(frase) {
          document.write(frase);
          saltarLinea();
      }

      var invitados = prompt(parseInt("Número de invitados"));
      var vips = prompt(parseInt("Número de invitados VIP's"));

      var total = invitados + vips;

      imprimir("El total de invitados es " + total);

  </script>
```

Trabajando con Condiciones

If

Ejecuta una sentencia si una condición especificada es evaluada como verdadera.
Si la condición es evaluada como falsa, otra sentencia puede ser ejecutada.

```
<body>
    <div>
        <script>
            function saltarLinea() {
                document.write("<br><br>");
            }

            function imprimir(frase) {
                document.write(frase);
                saltarLinea();
            }

            var victorias = parseInt(prompt("Ingresa la cantidad de Victorias"));
            var empates = parseInt(prompt("Ingresa la cantidad de Empates"));

            var puntosTotal = (victorias * 3) + empates;

            imprimir("El Total de puntos del Equipo es: " + puntosTotal);

            if (puntosTotal > 28){
                imprimir("El equipo esta mejor que el año pasado");
            }
            if (puntosTotal < 28){
                imprimir("El equipo esta peor que el año pasado");
            }
            if(puntosTotal == 28){
                imprimir("El equipo esta igual que el año pasado");
            }
        </script>
    </div>
</body>
```

Aplicamos **If** (condición) para comparar las condiciones.

Analiza la condición **If** y de ser cierta la defina como **True** pasando a mostrar el mensaje indicado.

En caso de ser **False** pasa la siguiente condición **If** y compara nuevamente.

¿Será que entra cumpliendo las condiciones?

Tenemos la declaración de la siguiente variable:

```
var numero = 10;
```

Ahora, tenemos las siguientes condiciones:

A)

```
if(numero >= 2 && numero < 12) {
    alert("ENTRÓ!");
}
```

COPIA EL CÓDIGO

B)

```
if(numero >= 2 &&  < 12) {
    alert("ENTRÓ!");
}
```

COPIA EL CÓDIGO

C)

```
if(numero >= 10) {
    alert("ENTRÓ!");
}
```

COPIA EL CÓDIGO

¿Cuál de las opciones de abajo fueron verdaderas y se mostró la alerta ENTRÓ!?

A. A y B

B. Solo C

✓ A y C

¡Correcto! Ambas cumplen la condición.

D. C y B

```
<body>
    <h3>Programa Cálculo IMC</h3>
    <div>
        <script>
            function saltoLinea(){
                document.write("<br><br>");
            }

            function imrpimir(frase){
                document.write(frase);
                saltoLinea();
            }

            function calcularImc(peso,altura){

                return (peso / (altura * altura));
            }

            var nombre = prompt("Ingrese su Nombre");
            var pesoInformado = prompt(nombre + ", Ingrese su Peso");
            var alruraInformado = prompt(nombre + ", Ingrese su altura");

            imcCalculado=calcularImc(pesoInformado,alruraInformado);
            imrpimir(nombre + " el IMC calculado es: " + imcCalculado);

            if (imcCalculado < 18.5){
                imrpimir("IMC abajo de lo Recomendado");
            }
            if (imcCalculado >=18.5){

                if (imcCalculado < 25){

                    imrpimir("IMC esta dentro del intervalo Normal");
                }
            }
            if (imcCalculado >=25){

                if (imcCalculado < 30){

                    imrpimir("IMC considerado como Sobrepeso");
                }
                }
            if (imcCalculado >=30){
                imrpimir("IMC considerado como Obesidad");
            }


        </script>
    </div>
</body>
```

En este ejercicio existe un poco más de complicación. De igual manera:

Aplicamos la condición **If** para comparar cada condición y dentro de cada If analiza la condición una vez más y de ser cierta la defina como **True** pasando a mostrar el mensaje indicado.
En caso de ser **False** pasa la siguiente condición **If** y compara nuevamente.

Math.**random**. se utiliza para devolver un número aleatorio predeterminado (Ejm: 0, 45 – 0,56) entre 0 y 1. Entonces multiplicamos por 10 para que nos devuelva un número aleatorio **(entre** 0 y 10**)**. Todo esto con la ayuda de Math.**round** que nos devuelve números enteros.

```
<body>
    <h3>Programa Juego Adivinar un Número</h3>
    <div>
        <script>
            function saltoLinea(){
                document.write("<br><br>");
            }

            function imrpimir(frase){
                document.write(frase);
                saltoLinea();
            }

            var numeroPensado = Math.round(Math.random() * 10);
            var numeroLanzado= parseInt(prompt("Ingrese un número entre 0 - 10 "));

            if(numeroPensado == numeroLanzado){
                imrpimir("Usted Acerto");
            }
            if (numeroPensado != numeroLanzado){
                imrpimir("Usted Erro, el número pensado era: " + numeroPensado);
            }

        </script>
    </div>
</body>
```

Else (si no)

La cláusula **else** "no obligatoria" sirve para indicar instrucciones a realizar en caso de no cumplirse la condición. JS admite escribir un else y dejarlo vacío: else { }. El else vacío se interpreta como que contemplamos el caso, pero no hacemos nada en respuesta a él.

Para mejorar aún nuestro código asignamos la instrucción **else**.
Si no ocurre la instrucción **If** ejecutara directamente la instrucción **else** en este caso (**imprimir**).

```
var numeroPensado = Math.round(Math.random() * 10);
var numeroLanzado= parseInt(prompt("Ingrese un número entre 0 - 10 "));

if(numeroPensado == numeroLanzado){
    imrpimir("Usted Acerto");
}
else {
    imrpimir("Usted Erro, el número pensado era: " + numeroPensado);
}
```

Debemos analizar cuando utilizamos **else**. En este caso tenemos 2 condiciones que nos resulta practico aplicarlo. En el ejercicio anterior (IMC Tabla) que se realizo no resulta factible aplicarlo ya que teníamos 4 condiciones que realizaba el **If** y por ende no resultaría factible aplicar **else.**

La condición if y else

¿Cuál de las opciones de abajo usa correctamente la condición `if` y `else`?

A

```
var contrasenha = prompt("Entra con tu contraseña");

if(contrasenha == "micontrasenha123") {
    alert("Acceso liberado!");
else {
    alert("Acceso negado!");
}
```

✓

```
var contrasenha = prompt("Entra con tu contraseña");

if(contrasenha == "micontrasenha123") {
    alert("Acceso liberado!");
} else {
    alert("Acceso negado!");
}
```

¡Correcto! La condición `if` necesita recibir el resultado de una operación lógica dentro de los paréntesis.

```
if(contrasenha == "micontrasenha123")
```

COPIA EL CÓDIGO

Además, es dentro del bloque que indicamos cuáles instrucciones se ejecutarán si la operación lógica fuera `true`:

```
if(contrasenha == "micontrasenha123") {
    alert("Acceso liberado!");
}
```

COPIA EL CÓDIGO

Observa que tenemos dos llaves que comienzan y finalizan el bloque `if`. La instrucción `else` viene justo después del bloque `if`:

```
if(contrasenha == "micontrasenha123") {
    alert("Acceso liberado!");
} else
```

COPIA EL CÓDIGO

Así como la instrucción `if`, la instrucción `else` también tiene su bloque con las instrucciones que deseamos ejecutar si la condición `if` recibe `false`:

```
if(contrasenha == "micontrasenha123") {
    alert("Acceso liberado!");
} else {
    alert("Acceso negado!");
}
```

COPIA EL CÓDIGO

C

```
var contrasenha = prompt("Entra con tu contraseña");

if contrasenha == "micontrasenha123" {
    alert("Acceso liberado!");
} else {
    alert("Acceso negado!");
}
```

¿Será qué puedo conducir?

Este programa, de acuerdo con la respuesta del usuario, debe mostrar el mensaje "Puedes conducir" o "No puedes conducir". Recuerda que puede conducir solo quien tiene 18 años o más y también quien tiene una licencia de conducir.

```
<body>
    <script>
        function saltarLinea() {
            document.write("<br>");
        }

        function imprimir(frase) {
            document.write(frase);
            saltarLinea();
        }

        var edad = parseInt(prompt("¿Cuál es tu edad?"));
        var tieneLicencia = prompt("¿Tienes licencia? Responde S o N");

        if ((edad >= 18) && (tieneLicencia == "S")) {
            imprimir("Puedes conducir");
        }
        else {
            imprimir("No puedes conducir");
        }
    </script>
</body>
```

Desafíos:

1 – Crea una función de lotería que reciba un número `n` y sortee un número entre `0` a `n`, retornando ese valor. De esta forma, en vez de escribir `var numeroPensado = Math.round(Math.random()*n);`, escribirás `var numeroPensado = sorteo(n);`. Realiza esa modificación, creando una nueva función y utilízala de forma correcta.

2 – Hacer que tu juego exhiba, cuando el usuario falle el intento, si el número lanzado era mayor o menor al número pensando por el programa.

Desafío N°1

```
<body>
    <h3>Juego de Lóteria</h3>
    <script>
        function saltarLinea(){
            document.write("<br><br>");
        }
        function imprimir(frase){
            document.write(frase);
            saltarLinea();
        }
         function sorteo(n) {
                return Math.round(Math.random()* n);
         }

         var numeroPensado = sorteo(10);
         var numeroLanzado = parseInt(prompt("Ingrese un número entre 0-10"))

         if (numeroPensado == numeroLanzado){
            imprimir("Felicidades, Usted acertó");
         }
         else {
            imprimir("Usted perdio, el número sorteado era el: " + numeroPensado);
         }

    </script>
```

Desafío N°2

```
<body>
    <h3>Juego de N intentos</h3>
     <script>
      function saltarLinea(){
        document.write("<br><br>");
      }
      function imprimir(frase){
        document.write(frase);
        saltarLinea();
      }

      var numeroPenzado = Math.round(Math.random() * 10);
      var numeroLanzado = parseInt(prompt("Ingrese un número entre 0 - 10"));

      if(numeroPenzado == numeroLanzado){

            imprimir("Felicidades, Usted Acerto");
      }
      else {
          if (numeroLanzado > numeroPenzado ){

              imprimir("Usted erro, el número " + numeroLanzado +" ingresado es mayor al número " + numeroPenzado  + " pensado.");
          }
          if (numeroLanzado < numeroPenzado ){

              imprimir("Usted erro, el número " + numeroLanzado +" ingresado es menor al número " + numeroPenzado  + " pensado.");
          }
      }

    </script>
</body>
```

Lo que aprendimos

- Convertir string en número usando `parseInt`.
- Trabajar con condiciones `if` y `else`.
- Usar fórmulas matemáticas como `Math.random` y `Math.round`.

While

Crea un bucle que ejecuta una sentencia especificada mientras cierta condición se evalúe como verdadera. Dicha condición es evaluada antes de ejecutar la sentencia.

```
<body>
    <h3>Programa - Año Mundial de la FiFa</h3>
    <script>
        function saltarLinea(){
            document.write("<br><br>");
        }
        function imprimir(frase){
            document.write(frase);
            saltarLinea();
        }

        var anioMundial = 1930;

        while (anioMundial <= 2022){
            imprimir("Hubo un Mundial de la FIFA en el año: " + anioMundial);
            // De ahí procede a sumar + cada 4 años se dio cada mundial
            anioMundial = anioMundial +4;
        }
        alert("FIN");
    </script>
</body>
```

Realizaremos lo siguiente:
Que sucede si el usuario quiere saber cuántos mundiales se realizaran hasta x año. (Ejm: 2050)

```
<body>
    <h3>Programa - Año Mundial de la FiFa</h3>
    <script>
        function saltarLinea(){
            document.write("<br><br>");
        }
        function imprimir(frase){
            document.write(frase);
            saltarLinea();
        }

        var anioMundial = 1930;
        var aniolimite =  parseInt(prompt("Ingrese el año limite para calcular"));

        while (anioMundial <= aniolimite){
            imprimir("Hubo un Mundial de la FIFA en el año: " + anioMundial);//muestra el
            // De ahí procede a sumar + cada 4 años se dio cada mundial
            anioMundial = anioMundial +4;
        }
        imprimir(" ******** FIN ******** ");
    </script>
</body>
```

✓ **En While** Las iteraciones son utilizadas para no tener que repetir el código un millón de veces.

☞ Mucho cuidado con la condición ya que si no consigo salir de la condición “nunca va a parar” y queda iniciada siempre.

Dejando muy claro que erró

Daniel siempre fue muy exagerado. En su versión particular del juego de adivinanzas, cuando el usuario falla el intento de acertar, se muestran diez advertencias de que ha cometido un error en la pantalla.

Aquí está el código de Daniel:

```
    <meta charset="UTF-8">
<script>
    function saltarLinea() {

        document.write("<br>");
    }
    function imprimir(frase) {

        document.write(frase);
        saltarLinea();
    }
    function sortearNumero() {

        return Math.round(Math.random() * 100);
    }

    var numeroPensado = sortearNumero();
    var numeroLanzado = parseInt(prompt("Ingrese un número entre 0-10: "));

    if(numeroLanzado  == numeroPensado) {

        imprimir("Uau! Vos acertaste, pues yo pensé en el número " + numeroPensado);
    }
    else {

        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
        imprimir("¡Se acabó, vos erraste!");
    }
</script>
```

Ten en cuenta que las diez instrucciones que se muestran son las mismas.

Ayuda a Daniel marcando la opción que contiene otra forma más eficiente de mostrar el mensaje ("¡Se acabó, vos erraste!") ¡10 veces! De esa manera, si quiere alterar el mensaje tendrá que cambiar el texto en un solo lugar.

A

```
<body>
    <meta charset="UTF-8">
    <script>

        // funciones omitidas saltarLinea, imprimir, sortearNumero

        var numeroPensado = sortearNumero();

        var numeroLanzado = parseInt(prompt("Ingrese un número entre 0-10: "));

        if(numeroLanzado  == numeroPensado) {

            imprimir("Uau! Vos acertaste, pues yo pensé en el número " + numeroPensado);
        } else {
            var contador = 1;

            while(contador < 10) {

                    imprimir("¡Se acabó, vos erraste!");
                    contador = contador +1;
                }
            }
    </script>
```

Al ejecutar el código se imprime 9 veces el mensaje (“¡Se acabó, vos erraste”)

B

```
<body>
    <h3>Juego de N intentos</h3>
        <script>
            function saltarLinea(){
                document.write("<br><br>");
            }
            function imprimir(frase){
                document.write(frase);
                saltarLinea();
            }
            function sortearNumero(){
                return Math.round(Math.random()* 10);
            }

            var numeroPensado = sortearNumero();
            var numeroLanzado = parseInt(prompt("Ingrese un número entre 0 - 10: "));

            if (numeroLanzado == numeroPensado){
                imprimir("Uau! Vos acertaste, pues yo pensé en el número " + numeroPensado);
            }
            else{
                var contador = 1;

                while (contador <= 10){
                    imprimir("¡Se acabó, vos erraste!");
                    contador = contador +1;
                }
            }

        </script>
</body>
```

¡Correcto! Recuerda que el `while` repetirá TODAS las instrucciones que estén dentro de su bloque `{ }` cuando la condición pasada sea `true`. Observa que el `while` recibe en sus paréntesis `()` la expresión `contador <= 10`.
Cuando el navegador interpreta la línea del `while` preguntará: ¿El valor de la variable contador es menor que o igual a 10? Como variable contador recibe el valor 1 en su inicialización el resultado de la expresión `contador <= 10` será `true`. Siendo así, el `while` estará autorizado para ejecutar la instrucción dentro de su bloque, en ese caso, la instrucción `imprimir("¡Se acabó, vos erraste!");` y la instrucción `contador = contador + 1`.
¿Tiene sentido que él ejecute `imprimir("¡Se acabó, vos erraste!")`? Por supuesto, pero ¿recuerdas la razón por la que hacemos `contador = contador + 1`? Ten en cuenta que esta variable existe solo para un propósito: es la que romperá la sección de repetición `while`. Con cada repetición de la instrucción `imprimir`, necesitamos aumentarlo para saber que ya lo hemos ejecutado, una, dos, tres veces, etc.
Si no tenemos la instrucción `contador = contador + 1`, la condición pasada al `while` dará siempre `true` y caeremos en una repetición infinita, el famoso `loop infinito` y muy probable que nuestro navegador se cuelgue.

C

```
<meta charset="UTF-8">
<script>

    // funciones omitidas saltarLinea, imprimir, sortearNumero

    var numeroPensado = sortearNumero();

    var numeroLanzado = parseInt(prompt("Ingrese un número entre 0-10: "));

    if(numeroLanzado  == numeroPensado) {

        imprimir("Uau! Vos acertaste, pues yo pensé en el número " + numeroPensado);
    } else {
        var contador = 1;

        while(contador <= 10) {

            imprimir("¡Se acabó, vos erraste!");
        }
    }
</script>
```

Como No tiene la instrucción: **contador** = contador **+ 1**
La condición pasada al while dará siempre true y caeremos en una repetición infinita, el famoso loop infinito y muy probable que nuestro navegador se cuelgue.

Tabla de Multiplicar

```
<body>
    <h3>Programa - Tabla de Multiplicar</h3>
        <script>
            function saltarLinea(){
                document.write("<br><br>");
            }
            function imprimir(frase){
                document.write(frase);
                saltarLinea();
            }

            var numeroIngresado =parseInt(prompt("Ingrese la Tabla a multiplicar que quiere sabe
            var multiplicador = 1;

            while (multiplicador <= 10){
                imprimir(numeroIngresado * multiplicador);
                multiplicador = multiplicador +1;
            }
        </script>
</body>
```

For

La instrucción **for** permite repetir una instrucción o una instrucción compuesta "un número especificado de veces". El cuerpo de una instrucción for se ejecuta cero o más veces hasta que una condición opcional sea **false**.

```
for (expression 1; expression 2; expression 3) {
        // code block to be executed
}
```

La expresión 1 se ejecuta (una vez) antes de la ejecución del bloque de código.
La expresión 2 define la condición para ejecutar el bloque de código.
La expresión 3 se ejecuta (todas las veces) después de que se haya ejecutado el bloque de código.

```
<body>
    <h3>Programa - Tabla de Multiplicar</h3>
        <script>
            function saltarLinea(){
                document.write("<br><br>");
            }
            function imprimir(frase){
                document.write(frase);
                saltarLinea();
            }
            function sortearNumero(){
                return Math.round(Math.random()* 10);
            }

            var numeroIgresado =parseInt(prompt("Ingrese la Tabla a multiplicar que quiere saber
            var multiplicador = 1;

            for (multiplicador; multiplicador <= 10;  multiplicador = multiplicador +1){
                imprimir(numeroIgresado * multiplicador);
            }
        </script>
</body>
```

De while para for

Claudia y Armando están casados y ambos son médicos. Sin embargo, siempre estuvieron interesados en la tecnología y se inscribieron en Alura para aprender lógica de programación.

Recibieron el siguiente código como tarea:

```
var contador = 1;
while(contador <= 10) {
    alert("Contador actual: " + contador);
    contador = contador + 1;
}
```

Necesitan convertirlo para usar la instrucción `for`.

¿Cuál de las siguientes opciones realiza esta conversión correctamente?

A

```
<meta charset="UTF-8">
<script>
for( var contador = 1; contador <= 10; contador = contador + 1 ) {
    alert("Contador actual: " + contador);
}
</script>
```

¡Correcto! La declaración `for` se divide en tres partes que están separadas por un punto y coma. La primera se usa para declarar la variable que usaremos como contador, la segunda para la condición de repetición y la última para el incremento de la variable contador. De hecho, cuando queremos incrementar uno por uno, podemos usar el incremento posterior:

```
for( var contador = 1; contador <= 10; contador++ ) {
    alert("Contador actual: " + contador);
}
```

Ten en cuenta que usamos `contador++` en lugar de `contador=contador + 1`. La primera forma es mucho más ágil.

B

```
<meta charset="UTF-8">
<script>
for( var contador = 1; contador = contador + 1; contador <= 10 ) {
    alert("Contador actual: " + contador);
}
</script>
```

C

```
<meta charset="UTF-8">
<script>
for( var contador = 1; <= 10; contador = contador + 1 ) {
    alert("Contador actual: " + contador);
}
</script>
```

Calcular la media de las Edades de una Familia

```
<body>
    <h3>Programa - Calcular la media de las Edades de una Familia </h3>
        <script>
            function saltarLinea(){
                document.write("<br><br>");
            }
            function imprimir(frase){
                document.write(frase);
                saltarLinea();
            }
            var edadPadre = 65;
            var edadMadre = 55;
            var edadHijo1 = 37;
            var edadHijo2 = 30;

            var totalEdad = (edadPadre + edadMadre + edadHijo1 + edadHijo2);
            var mediaEdad = (totalEdad / 5);

            document.write("La media de las edades es: " + mediaEdad);

        </script>
</body>
```

En la iteración **totalEdades** estoy acumulando la edad, inicializo variable **totalEdades** fuera del **while**

```
<body>
    <h3>Programa - Calcular la media de las Edades de una Familia </h3>
        <script>
            function saltarLinea(){
                document.write("<br><br>");
            }
            function imprimir(frase){
                document.write(frase);
                saltarLinea();
            }

            var numeroMiembros = parseInt(prompt("Ingrese la cantidad de miembros de su familia"));
            var contador =1;
            var totaledades = 0;

            while (contador <= numeroMiembros){
                var edad = parseInt(prompt("Ingrese la edad del familiar: "));
                totaledades = totaledades + edad;
                contador++;
            }

            mediaEdades = totaledades / numeroMiembros;

            imprimir("La media de las edades de la familia es: " + mediaEdades);
            imprimir("*** FIN ***");
        </script>
</body>
```

Todos los números pares del 1 al 100

¿Listo para un ejercicio que te hará romper un poco la cabeza? Usando la instrucción `while`, imprime todos los números pares del 1 al 100 y al final imprime la palabra "FIN".

```
<body>
    <h3>Programa - Calcula los Números Pares</h3>
    <script>
        function saltarLinea(){
            document.write("<br><br>");
        }
        function imprimir(frase){
            document.write(frase);
            saltarLinea();
        }

        var contador = 2;
        imprimir("Números pares hasta el 100");

        while(contador <= 100 ){

            imprimir(contador);
            contador = contador + 2;//incrementar de dos en dos
        }
        imprimir("*** FIN ***");

    </script>
</body>
```

El contador comienza en 2 y con cada repetición se incrementa en dos más. Entonces, tendremos 2, 4, 6, 8 ... ¡hasta llegar a 100! Cuando el contador alcanza 102, el `while` se negará a repetir, por lo que se ejecutará la siguiente instrucción.

Recordar: Cuando trabajamos con un **while** siempre necesitamos de un **contador**.

Mejorando el código de (Juego_adivinacion1) ahora el jugador tendrá la opción de adivinar el numero hasta 3 intentos.

De igual manera nuestro contador **++**; debe ir al final de las instrucciones **if** { } y **else** { }

Break. - corta la instrucción while inmediatamente "va a salir de la instrucción" y sigue ls siguiente intrucción imprimir(" **** FIN ***");

```
<body>
    <h3>Programa Juego Adivinar un Número</h3>
        <script>
            function saltoLinea(){
                document.write("<br><br>");
            }
            function imrpimir(frase){
                document.write(frase);
                saltoLinea();
            }

            var numeroPensado = Math.round(Math.random() * 10);
            var intentos = 3;
            var contador = 1;

            while(contador <= intentos) {

                var numeroLanzado= parseInt(prompt("Ingrese un número entre 0 - 10 "));

                if(numeroPensado == numeroLanzado){
                    alert("Usted Acerto, en el intento: " + contador + " el número pensado era: " + numeroPensado);
                    break;
                }
                else {
                    alert("Usted Erro");
                }
                contador ++;
            }

            if(numeroPensado == numeroLanzado){
                    imrpimir("Usted Acerto, en el intento: " + contador + " el número pensado era: " + numeroPensado);
            }
            else {
                    imrpimir("Usted Erro, el número pensado era: "  + numeroPensado);
            }

            imrpimir("***** FIN *****");

        </script>
</body>
```

El programa que no para de preguntar

Observa el siguiente código:

```
<meta charset="UTF-8">

<script>

    function saltarLinea() {
        document.write("<br>");
    }
    function imprimir(frase) {
        document.write(frase);
        saltarLinea();
    }

    var respuesta = ""; // todavía no hay respuesta

    while(true) {
        respuesta = prompt("¿Cuál es tu nombre?");
    }
    imprimir("FIN");

</script>
```

Ten en cuenta que `while()` recibe el valor fijo `true`. Mientras el valor entre paréntesis sea `true`, repetirá la declaración dentro del bloque `{}`.

¿Cuál de las siguientes opciones tiene la versión correcta del código `while()`, donde en caso de que la respuesta del usuario sea "SALIR", el programa va a terminar, y por lo tanto, dejar de preguntar el nombre del usuario?

A

```
<meta charset="UTF-8">

<script>

    // función saltarLinea e imprimir omitidos

    var respuesta = ""; // todavía no hay respuesta

    while(true) {
        respuesta = prompt("¿Cuál es tu nombre?");
        break;
    }

    imprimir("FIN");

</script>
```

B

```
<meta charset="UTF-8">

<script>

    // función saltarLinea e imprimir omitidos

    var respuesta = ""; // todavía no hay respuesta

    while(true) {
        respuesta = prompt("¿Cuál es tu nombre?");
        if( respuesta = "SALIR") {
            break;
        }
    }

    imprimir("FIN");

</script>
```

C

```
<body>

    <meta charset="UTF-8">

    <script>

        function saltarLinea() {
            document.write("<br>");
        }
        function imprimir(frase) {
            document.write(frase);
            saltarLinea();
        }

        var respuesta = ""; // todavía no hay respuesta

        while(true) {
            respuesta = prompt("¿Cuál es tu nombre?");
        }
        imprimir("FIN");

    </script>
```

¡Correcto! Mientras la respuesta no sea SALIR, seguirá preguntando su nombre.

Repeticiones Anidadas

(**for** dentro otro **for**)

```
<body>
    <h3>Programa - Estrellas</h3>
    <script>
        function saltarLinea(){
            document.write("<br>");
        }
        function imrpimir(frase){
            document.write(frase);
            saltarLinea();
        }

        for (var lineas =1; lineas <= 3; lineas++){

            for( var columnas =1; columnas <=10; columnas++){
                document.write("*");
            }
            saltarLinea();

        }
        saltarLinea();
        saltarLinea();
        imrpimir("  FIN  ");

    </script>
</body>
```

Programa - Estrellas

****
****
****

FIN

Simulando una pantalla de inicio de sesión

Es común que los programas tengan una pantalla de inicio de sesión, es decir, la pantalla que identifica al usuario en el sistema. Para permitir el acceso, el nombre de usuario y la contraseña deben coincidir con los valores almacenados por la aplicación, independientemente de dónde se guardó la información. Tampoco es raro permitir un máximo de tres intentos y después del tercero, el sistema se niega a identificar al usuario y notifica al administrador del sistema o incluso al propietario de la cuenta.

Aquí está el código que inicia sesión, pero ATENCIÓN, solo le da una oportunidad al usuario:

```
<meta charset="UTF-8">
<script>

    var inicioDeSesionRegistrado = "alura";
    var contrasenhaRegistrada = "alura321";

    var inicioDeSesionIngresado = prompt("Ingrese su usuario");
    var contrasenhaIngresada = prompt("Ingrese su contraseña");

    if( inicioDeSesionRegistrado == inicioDeSesionIngresado && contrasenhaRegistrada == contrasenhaIngresada ) {
        alert("Bienvenido al sistema " + inicioDeSesionIngresado);
    } else {
        alert("inicio de sesión inválido. Favor intente de nuevo");
    }
</script>
```

Cambia el código anterior para que el usuario tenga 3 intentos de inicio de sesión. Atención: si lo hace bien en el primer intento, no tiene sentido seguir preguntando por su nombre de usuario y contraseña.

```
<script>

    var inicioDeSesionRegistrado = "alura";
    var contrasenhaRegistrada = "alura321";

    var intentosMax = 3;
    var intentoAct = 1;

    while(intentoAct <= intentosMax){

        var inicioDeSesionIngresado = prompt("Ingrese su usuario");
        var contrasenhaIngresada = prompt("Ingrese su contraseña");

            if( inicioDeSesionRegistrado == inicioDeSesionIngresado && contrasenhaRegistrada == contrasenhaIngresada ) {
                alert("Bienvenido al sistema " + inicioDeSesionIngresado);
                intentoAct = intentosMax;//paso todos los intentos que falten y salgo del loop. Allá abajo aumentará +1!
            }
            else {
                if (intentoAct == 3) {
                    alert("Agotaste el número permitido de intentos!");
                }
                else {
                    alert("Inicio de sesión inválido. Favor intente de nuevo");
                }
            }
            // vaya al próximo intento
            intentoAct = intentoActual +1
    }
    document.write("<br>");
    document.write(" * * * * * * * * * * * * * * * ");

</script>
```

Esto ni siquiera se acerca a la complejidad de crear un sistema de autenticación (inicio de sesión del usuario), pero experimentar con el tema, es sin duda un incentivo para continuar en el universo de la programación. ¡Sigue estudiando!

Lo que aprendimos

- Repetir tareas y actividades con iteraciones `while`.
- Repetir tareas y actividades con iteraciones `for`.
- Interrumpir iteraciones con `break`.
- Realizar iteraciones anidadas, un `loop` dentro de otro.

Input

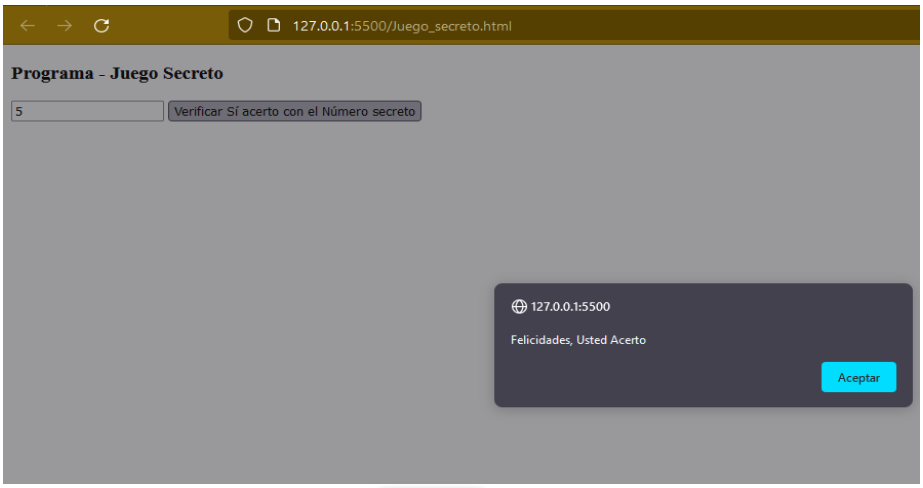
Es un elemento en html donde el usuario puede ingresar cualquier valor, se usa para crear controles interactivos para formularios basados en la web. `<input>`

Button

Un botón en el que se puede hacer clic. `<button> </button>`

```
<body>
    <h3>Programa - Juego Secreto</h3>
    <input type="text">
    <button>Verificar Sí acerto con el Número secreto</button>

    <script>
        var secreto =5;
        var input = document.querySelector("input");

        function verficar(){

            if(parseInt(input.value) == secreto){
                alert("Felicidades, Usted Acerto");
            }
            else {
                alert("Lo siento, Usted Erro");
            }
        }

        var button = document.querySelector("button");
        button.onclick = verficar;

    </script>
</body>
```

- QuerySelector.(“input”) - Traerá todos los datos que se ingresen en el input en Html. (establece comunicación entre JS. Y Html).
- En la condición **if** no puedo comparar **input** directamente porque en este caso input es un objeto “está capturando toda la etiqueta”. Tengo que pasarle **input.value** y transformarlo a número con “Parseint”
- De igual manera para comunicar mi botón de Html con JS utilizo: QuerySelector.(“button”) button.**onclick** - al dar clic me permitirá llamar a la función verificar y ejecutar la condición que se programó.



Más interacción del mundo JavaScript con mundo HTML

A lo largo de la capacitación, aprendimos a capturar la entrada del usuario a través de la función `prompt()`. No hay nada de malo en eso para aquellos que están aprendiendo lógica de programación, pero no vamos a dejar de aprender algo un poco más sofisticado y mejor para el usuario, ¿verdad?

El lenguaje HTML tiene etiquetas para capturar la entrada del usuario, incluidas las que representan botones.

Veamos un ejemplo:

```
<meta charset="UTF-8">

<input />
<button>Haz click aquí</button>

<script>

</script>
```

Cuando se carga el código, se muestra un campo de entrada rectangular para que el usuario ingrese datos y un botón (`button`). Sin embargo, `HTML` es un lenguaje estático, se muestran tanto el `input` como el `button` y no sucede nada. La buena noticia es que en el mundo `JavaScript` puedes acceder a las etiquetas del mundo `HTML`. Esto significa que podemos capturar lo que el usuario escribió en el `input`, incluyendo la programación de una respuesta para cuando hace clic en el `button`.

¿Cuál de las siguientes opciones captura la entrada de la caja de texto y el botón de las etiquetas de `HTML` en el mundo `JavaScript`?

```
<meta charset="UTF-8">

<input />
<button>Haz click aquí</button>

<script>
    var entrada = document.querySelector("input");
    var boton = document.querySelector("button");
</script>
```

¡Correcto! Lo aprendiste a través del `document.write` que escribimos en la pantalla, pero el `document` tiene también otras funcionalidades. Es a través de `document.querySelector` que podemos ir hasta el mundo `HTML` y llevar el elemento al mundo `JavaScript` para que podamos manipularlo. Pero ten cuidado, el correcto es `querySelector` con una `S` mayúscula. Si escribes con una `s` minúscula, cometerás un error de sintaxis. Continuando... `document.querySelector` recibe un parámetro del nombre de las etiquetas que queremos buscar del mundo HTML. Es más poderoso de lo que piensas y acepta recibir otros tipos de parámetros, pero para nuestra capacitación, entender que si pasamos el nombre de una etiqueta `HTML` nos devolverá la etiqueta en el mundo `JavaScript` es suficiente. Hay otras cuestiones involucradas en este proceso, pero no te preocupes. Las conocerás si deseas profundizar tus estudios en el lenguaje `JavaScript` .

B

```
<meta charset="UTF-8">

<input />
<button>Haz click aquí</button>

<script>
    var input = document.querySelector("entrada");
    var button = document.querySelector("boton");
</script>
```

C

```
<meta charset="UTF-8">

<input />
<button>Haz click aquí</button>

<script>
    var entrada = document.queryselector("input");
    var boton = document.queryselector("button");
</script>
```

Mejorando nuestro código

- input.value = ""; => Me permite limpiar el contenido que ingrese en input.
- input.focus(); => Le indica al cursor que vaya directamente a la casilla de input "se focalice ahí"
- Para indicarle al cursor que vaya directamente a la casilla de input una vez cargue la página.

  (input.focus();) Pongo la función al inicio antes de llamar a la función.

```
var secreto =5;
var input = document.querySelector("input");
input.focus();

function verficar(){
```

- De igual manera utilizamos la función conocida => Math.round(Math.random()*10); Nos permitirá generar un numero aleatorio y comprobar si acertó el número cuando ingresa en el casillero (input).

```
var secreto = Math.round(Math.random()*10);
```

```
<body>
    <h3>Programa - Juego Secreto</h3>
    <input type="text">
    <button>Verificar Sí acerto con el Número secreto</button>

    <script>
        var secreto = Math.round(Math.random()*10);
        var input = document.querySelector("input");
        input.focus();

        function verficar(){

            if(parseInt(input.value) == secreto){
                alert("Felicidades, Usted Acerto");
            }
            else {
                alert("Lo siento, Usted Erro");
            }
            input.value = "";
            input.focus();
        }

        var button = document.querySelector("button");
        button.onclick = verficar;

    </script>
</body>
```

Exhibiendo en un alert el nombre digitado

Carla estaba ayudando a sus colegas a prepararse para la próxima prueba de lógica de la programación. Le pidieron que resolviera la siguiente pregunta que tenía probabilidad de aparecer en la prueba: "lee del mundo `HTML` un texto escrito por el usuario, y con el clic de un botón muestra lo que se escribió".

```
<meta charset="UTF-8">

<input/>
<button>Mostrar texto escrito</button>

<script>
    var input = document.querySelector("input");

    function mostrarTexto() {

    }

    var button = document.querySelector("button");
    button.onclick = mostrarTexto;

</script>
```

ella asoció la función con `button.onclick`. Con esta asociación, cada vez que se hace clic en el botón, se ejecutará `mostrarTexto`, es como si, por detrás el navegador hiciera `mostrarTexto()` cada vez que se hace clic en el botón.
Pero cuando Carla comenzó a explicar el código de la función `mostrarTexto`, recibió una llamada y tuvo que salir con urgencia y dejó a medias la explicación.

¿Y ahora? ¿Pudieron sus amigos terminar el código?

¿Cuál de las siguientes opciones tiene el código completo de la función `mostrarTexto`? Recuerda que el código debe funcionar como se esperaba, que es mostrar una alerta al usuario cuando hace clic en el botón.

A

```
<meta charset="UTF-8">

<input/>
<button>Mostrar texto escrito</button>

<script>
    var input = document.querySelector("input");

    function mostrarTexto() {

        alert(value);
    }

    var button = document.querySelector("button");
    button.onclick = mostrarTexto;
</script>
```

B

```
<meta charset="UTF-8">

<input/>
<button>Mostrar texto escrito</button>

<script>
    var input = document.querySelector("input");

    function mostrarTexto() {

        alert(input);
    }

    var button = document.querySelector("button");
    button.onclick = mostrarTexto;
</script>
```

C

```
<meta charset="UTF-8">

<input/>
<button>Mostrar texto escrito</button>

<script>
    var input = document.querySelector("input");

    function mostrarTexto() {

        alert(input.value);
    }

    var button = document.querySelector("button");
    button.onclick = mostrarTexto;
</script>
```

¡Correcto! Cuando se llama a la función `mostrarTexto`, pasará como parámetro de la función `alert` el valor `input.value`. No podemos simplemente pasar `input`, porque el `input` corresponde a la etiqueta y queremos que su valor provenga de esa etiqueta. Por eso, pasamos `input.value`.

Lo que aprendimos

- A mejorar la interacción del usuario con uso de botones.
- A mejorar la interacción del usuario con uso de cajas de texto.
- A mejorar la usabilidad de nuestro programa con `focus`.

Array (arreglos)

El objeto Array es un objeto global que es usado en la construcción de arrays, son objetos tipo lista de alto nivel.

Declarando un array

A lo largo del curso, aprendimos a tratar con diferentes tipos de datos, como texto, número y booleano (por ejemplo, `true` o `false`).

Ejemplos de estos tipos son:

```
<script>
    var nombre = "Regina"; // texto, o técnicamente hablando, un string
    var edad = 22; // número
    var tieneLicencia = true; // booleano
</script>
```

Finalmente, aprendimos sobre el tipo de datos llamado `array`. Permite que una variable tenga múltiples valores.

¿Cuál de las siguientes opciones declara correctamente un `array`?

A

```
var edades = 10,12,20,30;
```

```
var cosas = ["Gisele", 12, true];
```

¡Correcto! Cada array se declara entre corchetes, el famoso `[ ]`. Sin embargo, si tenemos una declaración como esta `var cosas = [ ]`, tenemos un `array`, es decir, una lista vacía sin ningún elemento. Podemos, al declarar un `array`, agregar elementos. Estos elementos pueden ser de cualquier tipo de dato conocido: `var cosas = ["Gisele", 12, true];`
El primer elemento de nuestro `array` es un `string`, el segundo un `número` y el último un `booleano`.

C

```
var nombres = {"Helena", "Sulivan", "Saladino"};
```

- Dentro del **for** se ejecutará lo siguiente:
  1ra parte: va a llamar a la posición del arreglo que hemos creado (**var** posicion **= 0**;)
  2da parte: se ejecutará la condición (posicion **< 4**)
  3ra parte: el suma rizador de la variable (posicion **++**)

- Dentro de **function verificar** creo la variable (**var** encontrado = **false**;) la cual al momento de acertar el número indicado finalizará y me mostrará el mensaje (“Felicidades, usted Acerto”).

- Luego la variable **var** encontrado la ubico dentro de la condición **if** después de la función **alert**(“Felicidades, Usted acertó”); . => (encontrado = **true**;).

- **Break**. - una vez que adivina el numero ya no necesita seguir comprobando las demás iteraciones entonces corta el **for** (finaliza el juego) y sale inmediatamente.

☞ El programa va a entrar y llamar a la función verificar, realizara una iteración de 4 veces preguntara si acertó el valor y así hasta cumplir las 4 iteraciones de darse el caso.

```
<body>
    <h3>Programa - Juego Secreto</h3>
    <input type="text">
    <button>Verificar Sí acerto con el Número secreto</button>

    <script>
        // var secreto = Math.round(Math.random()*10);
        var secretos =[3,5,7,9];

        var input = document.querySelector("input");
        input.focus();

        function verficar(){

            var encontrado = false;

            for(var posicion = 0; posicion < 4; posicion++){

                if(parseInt(input.value) == secretos[posicion]){
                    alert("Felicidades, Usted Acerto");
                    encontrado = true;
                    break;
                }
            }
            if(encontrado == false){

                alert("Lo siento, Usted Erro");
            }

            input.value = "";
            input.focus();
        }

        var button = document.querySelector("button");
        button.onclick = verficar;

    </script>
</body>
```

Length

Se usa para devolver la longitud de un objeto o cadena.

```
var secretos = [3,5,7,9];

secretos.length;
```

Utilizo para que lea toda la información que contiene mi Array (esta manera no necesito especificar si mi array contiene tal número de datos, como en el **for** anterior implementado.)

```
<body>
    <h3>Programa - Juego Secreto</h3>
    <input type="text">
    <button>Verificar Sí acerto con el Número secreto</button>

    <script>
        // var secreto = Math.round(Math.random()*10);
        var secretos =[3,5,7,9];

        var input = document.querySelector("input");
        input.focus();

        function verficar(){

            var encontrado = false;

            for(var posicion = 0; posicion < secretos.length; posicion++){

                if(parseInt(input.value) == secretos[posicion]){
                    alert("Felicidades, Usted Acerto");
                    encontrado = true;
                    break;
                }
            }
            if(encontrado == false){

                alert("Lo siento, Usted Erro");
            }

            input.value = "";
            input.focus();
        }

        var button = document.querySelector("button");
        button.onclick = verficar;

    </script>
</body>
```

El tamaño de un array

Tenemos el siguiente array:

```
var codigos  = ["IDNOCLIP" , "IDDQD", "IDKFA"];
```

Aprendimos que podemos conocer el tamaño del array a través de:

A `codigos.lenhgt`

`codigos.length`

¡Correcto! Ten mucho cuidado de no escribir `codigos.lenhgt`, de lo contrario tendrás problemas, ya que el resultado será `undefined`.

C `codigos.size`

Superman quedó fuera por un pelo

En la prueba de estructura de datos, Felipe tenía la siguiente lista:

```
var heroes = ["Superman", "Thor", "Batman", "Mujer Maravilla"];
```

El ejercicio pidió que mostrara un alert para cada ítem de la lista y lo implementó de la siguiente manera:

```
for( var i = 1; i < heroes.length; i++) {
    alert(heroes[i]);
}
```

A pesar de creer que su código es correcto, obtuvo un 7/10 porque cometió un error. Su maestro dijo que mostraba todos los nombres excepto "Superman".

¿Cuál es el problema con el código de Felipe? ¿Cómo puede cambiarlo para que muestre todos los nombres en la lista sin omitir ninguno?

la variable de incremento `i` comienza con el valor de `1`, pero debería ser `0`. Ya que nuestra variable de incremento se utiliza como índice para acceder a cada elemento de la lista, el primer elemento de todo `array` tiene un índice `0`. ¡Por eso "Superman" se quedó fuera!

Array => heroes ["Superman", "Thor", "Batman", "Mujer Maravilla"] inicia desde 0
0 , 1, 2, 3

La forma correcta del código sería:

```
<meta charset="UTF-8">

<script>
    var heroes = ["Superman", "Thor", "Batman", "Mujer Maravilla"];
    for( var i = 0; i < heroes.length; i++) {
        alert(heroes[i]);
    }
</script>
```

Lo que aprendimos

- Concepto y dinámica de los arrays.
- Usar iteraciones para cargar arrays.
- Descubrir el tamaño de los arrays.

```
<body>
    <h3>Programa - Juego Secreto</h3>
    <input type="text">
    <button>Verificar Sí acerto con el Número secreto</button>

    <script>
        function aleatorio(){
            return Math.round(Math.random()*10);
        }
        function sortearNumeros(cantidad){
            var secretos = [];
            var contador = 1;

            while(contador <= cantidad){

                numeroAleatorio = aleatorio();
                secretos.push(numeroAleatorio);
                contador++;
            }
            return secretos;
        }

        var secretos = sortearNumeros(4);
        console.log(secretos);

        var input = document.querySelector("input");
        input.focus();

        function verficar(){

            var encontrado = false;
            for(var posicion = 0; posicion < secretos.length; posicion++){

                if(parseInt(input.value) == secretos[posicion]){
                    alert("Felicidades, Usted Acerto");
                    encontrado = true;
                    break;
                }
            }
            if(encontrado == false){

                alert("Lo siento, Usted Erro");
            }

            input.value = "";
            input.focus();
        }

        var button = document.querySelector("button");
        button.onclick = verficar;

    </script>
</body>
```

- Llamo a la variable => **sortearNumeros(4)**) y le estoy pasando cuatro.
- En **function sortearNumeros**

  Va a crear una variable (**var** secretos **=** []) de arreglos vacía.

  Defino mi contador (**var** contador **= 1**) va a realizar la función de contador de 1 a 4 veces dentro de **while**. N

- En **while**

  Definimos nuestro número aleatorio (numeroAleatorio **=** **aleatorio**()).
  Luego ingresamos y adicionamos ese número aleatorio, con la función push (secretos.**push**(numeroAleatorio)) Y hacemos nuestro contador**++**;
  Y de esta manera nos retornara secretos.(**return** secretos)

Push. - Añade uno o más elementos al Array.

Ensalada de fruta

Tenemos la siguiente lista de frutas:

```
var frutas = ["piña", "banana", "melón"];
```

Si queremos agregar un elemento más al `array` de frutas, ¿qué debemos hacer?

A

```
frutas.push["aguacate"];
```

B

```
frutas = "aguacate";
```

```
frutas.push("aguacate");
```

¡Correcto! Todo `array` tiene una función `push` que le permite "insertar" elementos en la lista.

Mejorando nuestro código

Crearemos un mecanismo en nuestro código para que los números aleatorios No generen números repetidos.

Creamos una iteración con **for** en la cual estamos diciendo que iteremos cuantas veces hayamos cargado en nuestro arreglo secretos (**secretos.length**).

Entonces si esta en la 1ra iteración dirá posición es 0, (secretos.length) no tiene ningún tamaño "ni va a entra al **for**" y va a volver de nuevo.

Ahora (secretos.length) va a tener una posición entonces entra al **If** y va a comparar si lo que cargamos en la nueva rodada (secretos.length) va a ser igual al nuevo (**numeroAleatorio**) generado. Entonces (encontrado **= True**) si "True" **=** "True" (lo niega).

Luego va a **if** (encontrado **== false**) no va a entrar "No va a añadir ese número aleatorio".
Y va a generar otro numero Aleatorio.

Entonces yo puedo generar diez números aleatorios pero Voy a generar la cantidad de números que le ordene (**var** secretos **= sortearNumeros**(**4**))

```
<body>
    <h3>Programa - Juego Secreto</h3>
    <input type="text">
    <button>Verificar Sí acerto con el Número secreto</button>

    <script>

        function aleatorio(){
            return Math.round(Math.random()*10);
        }
        function sortearNumeros(cantidad){
            var secretos = [];
            var contador = 1;

            while(contador <= cantidad){

               var numeroAleatorio = aleatorio();
               var encontrado = false;

               if (numeroAleatorio != 0){

                    for(var posicion =0; posicion < secretos.length; posicion++){

                        if(numeroAleatorio == secretos[posicion]){
                            encontrado = true;
                            break;
                        }
                    }
                    if (encontrado == false){
                        secretos.push(numeroAleatorio);
                        contador++;
                    }

               }

            }

            return secretos;
        }
```

```
        var secretos = sortearNumeros(4);
        console.log(secretos);//ver en consola navegador que hace mi programa

        var input = document.querySelector("input");
        input.focus();

        function verficar(){

            var encontrado = false;

            for(var posicion = 0; posicion < secretos.length; posicion++){

                if(parseInt(input.value) == secretos[posicion]){
                    alert("Felicidades, Usted Acerto");
                    encontrado = true;
                    break;
                }
            }
            if(encontrado == false){

                alert("Lo siento, Usted Erro");
            }

            input.value = "";
            input.focus();
        }

        var button = document.querySelector("button");
        button.onclick = verficar;

    </script>
</body>
```

¡No hay ingredientes repetidos aquí!

Armando tuvo la idea de crear un programa para crear recetas. En él, hay una lista en la que el usuario puede agregar ingredientes. Observemos:

```
<meta charset="UTF-8">

<h1>Recetas de Armando</h1>

<script>

    var ingredientes = [];
    var cantidad = parseInt(prompt("¿Cuántos ingredientes vas a añadir?"));
    var contador = 1;

    while( contador <= cantidad) {
        var ingrediente = prompt("Informe el ingrediente " + contador);
        ingredientes.push(ingrediente);
        contador++;
    }

    console.log(ingredientes);

</script>
```

Al probar el código, descubrimos que funciona. El problema es que podemos agregar ingredientes repetidos a una lista de recetas. Cambia el código para que no se puedan agregar los ingredientes repetidos.

NOTA: No necesitas responder aquí, intenta resolver el problema en tu computadora y compara tu solución con el código que mostramos a continuación.

```
<body>
    <meta charset="UTF-8">

    <h1>Recetas de Armando</h1>

    <script>

        var ingredientes = [];
        var cantidad = parseInt(prompt("¿Cuántos ingredientes vas a añadir?"));
        var contador = 1;

        while( contador <= cantidad) {
            var ingrediente = prompt("Informe el ingrediente " + contador);
            var existe = false;

            for(var posicion = 0; posicion < ingredientes.length; posicion++){
                if(ingredientes[posicion] == ingrediente){
                    existe = true;
                    alert("repetido " + ingrediente);
                    break;
                }
            }
            if(existe == false) {
                ingredientes.push(ingrediente);
                contador++;
            }

        }

        document.write(ingredientes);
        console.log(ingredientes);

    </script>
</body>
```

Antes de añadir un ingrediente al `array`, debemos verificar si se ha agregado antes. Por lo tanto, verificamos, para cada ingrediente agregado, si existe en la lista y si no, podemos agregarlo.

Ten en cuenta que creamos una variable llamada existe, comienza con false y solo es verdadera si el elemento existe. Si esto sucede, no tiene sentido continuar escaneando el `array` y salimos del `loop` por medio del `break`, aprovechamos para informar al usuario con otro `alert` sobre el ingrediente repetido. El break `rompe` el `for`, al final, validamos la variable `existe` para ver si incluimos el ingrediente o no. Mucha atención, porque solo podemos aumentar la variable del contador cuando el ingrediente no existe. Esto es importante, ya que, si existe, debemos repetir la pregunta de nuevo ingrediente para el usuario.

Lo que aprendimos

- Ingresar valores manualmente en los `arrays` con la función `push`.
- Resolver problemas más complejos usando varios conceptos de programación juntos en un mismo programa (Loops, arrays, condiciones, funciones y fórmulas matemáticas).

Este curso te dio el requisito previo para aprender otros lenguajes de programación o profundizar en el lenguaje de `JavaScript`. Si quieres ser aún mejor en lógica de programación, hay un tercer curso que puede desafiarte aún más: el curso de Lógica de programación práctica.

Para ustedes, mis mejores deseos y mucho éxito en sus estudios. 💪

😊 🎉 🎊

Felicidades, si llegaste a este ejercicio es porque te esforzaste por ejercitar todo lo que aprendiste. La lógica de programación es un pequeño paso, pero al mismo tiempo es un avance gigantesco, porque abre nuevos caminos y carreras para ser conquistados en el mundo del desarrollo de sistemas.