



PROGRAMACIÓN WEB

ALCANTARA GINERA FATIMA

08/05/2024

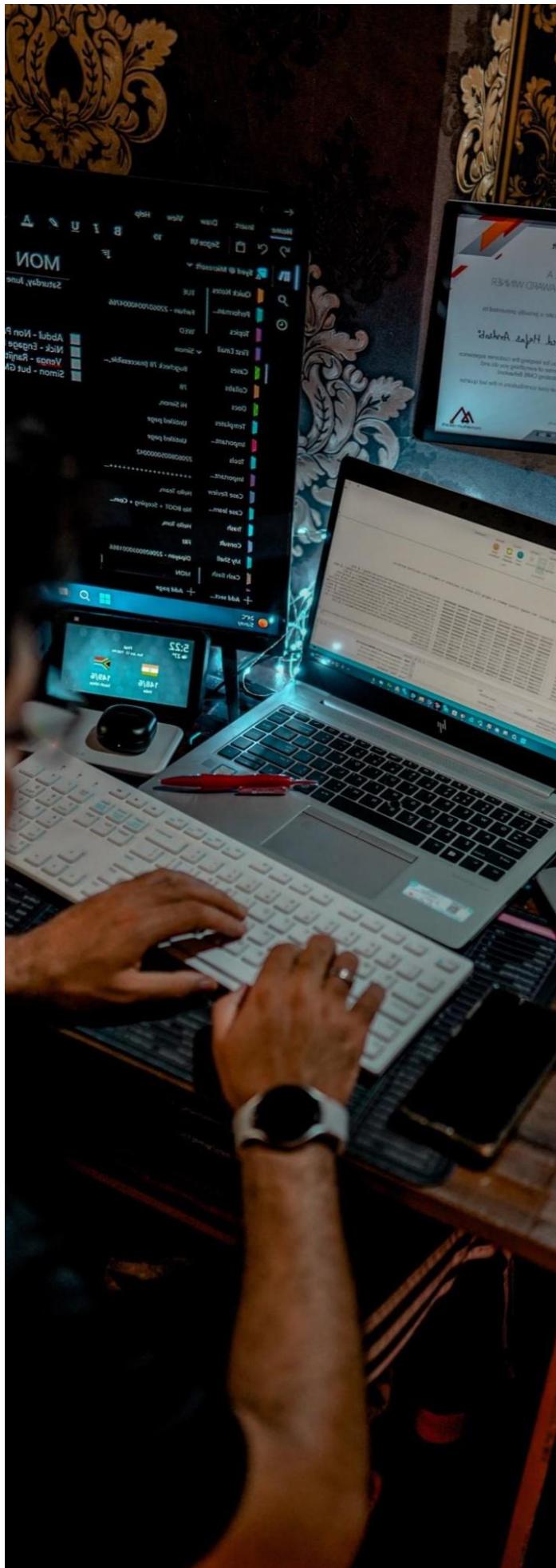


TABLE OF CONTENTS

INTRODUCCIÓN.....	3
RESUMEN.....	9
1. DESARROLLO	10
2. RESULTADOS.....	80
3. ConclusiÓN	111
4. Bibliography (Apa format)	112

INTRODUCCIÓN

El presente reporte tiene como objetivo principal la descripción detallada de lo realizado durante las prácticas de la materia programación web. En el análisis de este reporte técnico se encontrarán los pasos necesarios para recabar información y realizar cada una de las prácticas que realizamos en el Lenguaje de HyperText Markup Language (HTML), conectada a Cascading Style Sheets (css), en el cual también ocupamos JavaScript.

Organizado la presentación del reporte en quince actividades, de los cuales los tres primeros se dedican a las prácticas más simples del lenguaje html . css y JavaScript y los restantes constituyen la presentación de un nivel intermedio en el leguaje de html ,css y javaScript. Pasamos a exponer de forma sintética el contenido de las actividades que conforman este trabajo para que el lector pueda realizarse una primera representación.

RESUMEN

El objetivo principal de este proyecto es diseñar y desarrollar páginas web sencillas empezando desde lo más básico hasta un nivel intermedio del lenguaje HTML, creando así Fundamentos del Desarrollo Web: HTML, CSS y javaScrip son los lenguajes básicos utilizados para crear y diseñar páginas web. Al dominar estos lenguajes, se establece una base sólida.

El proyecto consiste en crear distintas actividades creando páginas web que presenten los conceptos básicos de HTML,css y javaScrip que se vean atractiva y fácil de crear. Se utilizará HTML para la estructura del sitio, incluyendo la creación de páginas. Además, se implementará CSS para el diseño y la estilización de las páginas web utilizando el Visual Estudio Code Editor.

1. DESARROLLO

PRÁCTICA 1: PIZARRA

Practica 1: consiste en la aplicación de una pizarra, que su funcionamiento el cual es que al dar clic al mouse empezara a dibujar en la pizarra, para esto llevaremos a cabo empezando con el lenguaje estructurado de HTML en el cual son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, el elemento de <Canvas> Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <!--Abre la etiqueta raíz del documento HTML y especifica el idioma principal del contenido (en este caso, inglés)-->
4   <head>
5     <!--Especifica el conjunto de caracteres (UTF-8) utilizado para codificar el contenido del documento-->
6     <meta charset="UTF-8">
7     <!--Define cómo el navegador debe controlar las dimensiones y la escala de la página en diferentes dispositivos-->
8     <meta name="viewport" content="width=device-width, initial-scale=1.0">
9     <!--Establece el título del documento, que se muestra en la barra de título del navegador o en la pestaña del navegador-->
10    <title>Document</title>
11    <!--Enlaza la hoja de estilos externa "style.css" con el documento HTML para aplicar estilos al contenido-->
12    <link rel="stylesheet" href="style.css">
13    <!--Cierra la sección de encabezado del documento-->
14  </head>
15  <!--Abre la sección del cuerpo del documento, donde se coloca el contenido visible del sitio web.-->
16  <body>
17    <!--Inserta un elemento <canvas> en el cuerpo del documento con el ID "pizarra". Este elemento se utilizará para dibujar líneas-->
18    <canvas id="pizarra"></canvas>
19  </body>
20  <!--Cierra la sección del cuerpo del documento-->
21 </html>
22 <!--Cierra la etiqueta raíz del documento HTML-->

```

Ilustración 1 Estructura básica de HTML

Posteriormente agregaremos comenzamos con el código de JavaScript comenzaremos con las variables. La variable es una característica, cualidad o propiedad observada que puede adquirir diferentes valores y es susceptible de ser cuantificada o medida. Para ser nominada como tal, debe tener la posibilidad de variar entre dos valores, como mínimo. Para generar una variable en JavaScript, se usa la palabra clave `let`. La siguiente declaración genera (en otras palabras: *declara* o *define*) una variable con el nombre “message”:

Variables:

1. **miCanvas**: Almacena una referencia al elemento `<canvas>` con el ID "pizarra".
2. **lineas**: Un array vacío para almacenar las líneas dibujadas.
3. **correccionX** y **correccionY**: Inicializadas en 0, se utilizan para ajustar la posición del mouse dentro del lienzo.
4. **pintarLinea**: Una variable booleana que indica si se está dibujando una línea actualmente.
5. **nuevaPosicionX** y **nuevaPosicionY**: Inicializadas en 0, almacenan la posición actual del mouse.

```

<script>
//=====
// VARIABLES
//=====
/*Esta línea selecciona un elemento HTML con el ID 'pizarra'*/
let miCanvas = document.querySelector('#pizarra');
/*Se declara una variable llamada 'lineas' como un arreglo vacío. Esta variable se utilizará para almacenar las líneas dibujadas*/
let lineas = [];
/* Se declara una variable `correccionX` e inicializa su valor en 0*/
let correccionX = 0;
/*Se declara una variable `correccionY` e inicializa su valor en 0*/
let correccionY = 0;
/*Se declara una variable `pintarLinea` e inicializa su valor en `false` */
let pintarLinea = false;
// Marca el nuevo punto
let nuevaPosicionX = 0;
let nuevaPosicionY = 0;
/*Esta línea utiliza el método `getBoundingClientRect()` en el elemento `miCanvas` para obtener un objeto `Dimension` que contiene la posición y el tamaño del lienzo*/
let posicion = miCanvas.getBoundingClientRect();
correccionX = posicion.x;
correccionY = posicion.y;

```

Ilustración 2 Creación de variables de JavaScript

Posteriormente comenzamos con las funciones estas funciones nos ayudaran a un procedimiento un conjunto de instrucciones que realiza una tarea o calcula un valor, pero para que un procedimiento califique como función, debe tomar alguna entrada y devolver una salida donde hay alguna relación obvia entre la entrada y la salida, en este procedimiento comenzaremos a definir las siguientes funciones utilizadas en el leguaje de JavaScript en las cuales utilizamos, ("empezarDibujo, dibujarLinea, dibujarLinea ,guardarLinea, pararDibujar:

1. Funciones:

1. EMPEZARDIBUJO:

- Establece la variable pintarLinea en true para indicar que se está comenzando a dibujar una línea.
- Agrega un nuevo array vacío a líneas para almacenar la nueva línea.

2. GUARDARLINEA:

- Agrega un nuevo objeto al último array en lineas. Este objeto contiene las coordenadas x e y de la posición actual del mouse.

```
//=====
// FUNCIONES
//=====

/**
 * Funcion que empieza a dibujar la linea
 */
function empezarDibujo () {
    pintarLinea = true;
    lineas.push([]);
};

/**
 * Funcion que guarda la posicion de la nueva linea
 */
function guardarLinea() {
    lineas[lineas.length - 1].push({
        x: nuevaPosicionX,
        y: nuevaPosicionY
    });
}
```

Ilustración 3 Creación de funciones

3. **dibujarLinea:**

- Previene el comportamiento predeterminado del evento (que podría ser seleccionar texto o arrastrar elementos).
- Verifica si la variable pintarLinea es true.
- Si se está dibujando una línea:
 - Obtiene el contexto de dibujo 2D del elemento <canvas> (ctx).
 - Establece los estilos de línea (grosor, estilo de unión y estilo de tapa).
 - Establece el color de la línea a blanco (#fff).
 - Obtiene la posición actual del mouse (nuevaPosicionX y nuevaPosicionY) considerando si es un evento de mouse o un evento táctil.
 - Llama a guardarLinea para almacenar la nueva posición del mouse.
 - Comienza un nuevo camino (ctx.beginPath).
 - Recorre el array lineas y dibuja cada línea:
 - Mueve el cursor a la primera posición de la línea (ctx.moveTo).
 - Dibuja líneas a cada punto subsecuente de la línea (ctx.lineTo).
 - Aplica el trazo a la línea (ctx.stroke).

4. **PARARDIBUJAR:**

- Establece la variable pintarLinea en false para indicar que se ha dejado de dibujar una línea.
- Llama a guardarLinea para almacenar la última posición del mouse.

```
function dibujarLinea (event) {  
    event.preventDefault();  
    if (pintarLinea) {  
        let ctx = miCanvas.getContext('2d')  
        // Estilos de linea  
        ctx.lineJoin = ctx.lineCap = 'round';  
        ctx.lineWidth = 10;  
        // Color de la linea  
        ctx.strokeStyle = '#fff';  
        // Marca el nuevo punto  
        if (event.changedTouches == undefined) {  
            // Versión ratón  
            nuevaPosicionX = event.layerX;  
            nuevaPosicionY = event.layerY;  
        } else {  
            // Versión touch, pantalla tactil  
            nuevaPosicionX = event.changedTouches[0].pageX - correccionX;  
            nuevaPosicionY = event.changedTouches[0].pageY - correccionY;  
        }  
        // Guarda la linea  
        guardarLinea();  
        // Redibuja todas las lineas guardadas  
        ctx.beginPath();  
        lineas.forEach(function (segmento) {  
            ctx.moveTo(segmento[0].x, segmento[0].y);  
        })  
        ctx.stroke();  
    }  
}
```

Ilustración 4 Creación de funciones para dibujar en la pizarra

Continuaremos con los eventos estos eventos son encargados de producir una señal de cierto tipo cuando un evento ocurre, y proporciona un mecanismo para que una acción se lleve a cabo (ejecutar código) de forma automática cuando el evento ocurra. Los eventos se lanzan dentro de la ventana del navegador y usualmente están asociados a un elemento en específico dentro de dicha ventana. Esto puede ser un solo elemento, un grupo de elementos, el documento HTML cargado la pestaña actual, o la ventana del navegador en su totalidad. Existen distintos tipos de eventos que pueden ocurrir.

Eventos:

- **Eventos de mouse:**

- ✓ mousedown: Se activa cuando se presiona el botón del mouse sobre el lienzo, llamando a la función empezarDibujo.
- ✓ mousemove: Se activa cuando se mueve el mouse sobre el lienzo,

```
5 //=====
6 // EVENTOS
7 //=====
8
9
10 // Eventos raton
11 miCanvas.addEventListener('mousedown', empezarDibujo, false);
12 miCanvas.addEventListener('mousemove', dibujarLinea, false);
13 miCanvas.addEventListener('mouseup', pararDibujar, false);
14
15 // Eventos pantallas táctiles
16 miCanvas.addEventListener('touchstart', empezarDibujo, false);
17 miCanvas.addEventListener('touchmove', dibujarLinea, false);
18
19 </script>
20
```

Ilustración 5 Creación de eventos del código de JAVASCRIPT

Después de definir el código de JavaScript continuaremos con el código de css este para darle diseño a la página que este será el diseño al elemento canvas, en el cual se le otorgara un ancho de deslizo que será una cantidad de 500 px, continuando también se le agregará una altura este de deslizo que también será por igual con una medición de 500px, por último se le agregara un color de fondo que este será un color negro

```
1 /*Selecciona todos los elementos <canvas> en el documento HTML.*/
2 canvas {
3     /*Establece el ancho del lienzo (canvas) en 500 pixeles.*/
4     width: 500px;
5     /*Establece la altura del lienzo (canvas) en 500 pixeles*/
6     height: 500px;
7     /*Establece el color de fondo del lienzo (canvas) en un tono oscuro de negro (#000909).*/
8     background-color: #000909;
9 }
10
```

Ilustración 6 ESTILOS DE CSS PARA LA PIZARA

PRÁCTICA 2: PIZARRA_2 DE PUNTOS

En la práctica 2 representa la creación de una pizarra, con el funcionamiento de con solo pasar el mouse se pintara en la pizarra, para esta práctica llevaremos a cabo empezando con el lenguaje estructurado de HTML en el cual son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, el elemento de <Canvas> Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.

HTML:

1. Declaración del tipo de documento: El código comienza con la declaración <!DOCTYPE HTML>, que indica al navegador que el documento es un archivo HTML5.
2. Etiqueta <html>: Abre la raíz del documento HTML y especifica el idioma principal del contenido como inglés (lang="en").
3. Sección <head>: Contiene metadatos e información sobre el documento:
4. Meta charset: Define la codificación de caracteres del documento como UTF-8 (<meta charset="UTF-8">).
5. Meta viewport: Controla cómo el navegador renderiza la página en diferentes dispositivos (<meta name="viewport" content="width=device-width, initial-scale=1.0">).
6. Título del documento: Establece el título de la página como "Pizarra Interactiva con Canvas" (<title>Pizarra Interactiva con Canvas</title>).
7. Estilos CSS: Define estilos CSS en línea para el elemento <canvas> (<style>). En este caso, se establece un borde negro sólido de 1 píxel.
8. Sección <body>: Contiene el contenido visible de la página:
9. Elemento <canvas>: Crea un elemento <canvas> con el ID "pizarra" para dibujar en él (<canvas id="pizarra"></canvas>).
10. Script JavaScript: Incluye un script JavaScript en la página (<script>).

```
index.html > html > head > style
1  <!--Declara la versión de HTML que el documento está utilizando.-->
2  <!DOCTYPE html>
3  <!--Abre la etiqueta raíz del documento HTML y especifica el idioma principal del contenido-->
4  <html lang="en">
5  <!--Abre la sección de encabezado del documento, donde se colocan metadatos y enlaces a recursos-->
6  <head>
7      <!--Especifica el conjunto de caracteres (UTF-8) utilizado para codificar el contenido-->
8      <meta charset="UTF-8">
9      <!--Define cómo el navegador debe controlar las dimensiones y la escala de la página en diferentes dispositivos-->
10     <meta name="viewport" content="width=device-width, initial-scale=1.0">
11     <!--Establece el título del documento-->
12     <title>Pizarra Interactiva con Canvas</title>
13     <!--Define estilos CSS dentro del propio documento HTML. En este caso, establece un borde negro sólido de 1 píxel-->
14     <style>
15         canvas {
16             border: 1px solid black;
17         }
18     </style>
19     <!--Cierra la sección de encabezado del documento-->
20 </head>
21 <!--Abre la sección del cuerpo del documento, donde se coloca el contenido visible del sitio-->
22 <body>
23     <!--Inserta un elemento <canvas> en el cuerpo del documento con el ID "pizarra"-->
24     <canvas id="pizarra"></canvas>
25     <!--Abre la etiqueta para agregar scripts JavaScript en el documento-->
26     <script>
```

Ilustración 7 ESTRUCTURA BASICA DE HTML AÑADIDO EL CANVAS

Posteriormente agregaremos comenzamos con el código de JavaScript comenzaremos con las variables. La variable es una característica, cualidad o propiedad observada que puede adquirir diferentes valores y es susceptible de ser cuantificada o medida. Para ser nominada como tal, debe tener la posibilidad de variar entre dos valores, como mínimo. Para generar una variable en JavaScript, se usa la palabra clave let. La siguiente declaración genera (en otras palabras: *declara* o *define*) una variable con el nombre "message"

JavaScript:

1. Variables:

- **miCanvas**: Almacena una referencia al elemento <canvas> con el ID "pizarra".
- **lineas**: Un array vacío para almacenar las líneas dibujadas.
- **correccionX y correccionY**: Inicializadas en 0, se utilizan para ajustar la posición del mouse dentro del lienzo.
- **nuevaPosicionX y nuevaPosicionY**: Inicializadas en 0, almacenan la posición actual del mouse.

```
//=====
// VARIABLES
//=====
/*Declara una variable miCanvas y la asigna al elemento <canvas> con el ID "pizarra"
let miCanvas = document.querySelector('#pizarra');
/*Declara una variable lineas como un array vacío que almacenará las líneas dibujadas
let lineas = [];
/*Declara una variable lineas como un array vacío que almacenará las líneas dibujadas
let correccionX = 0;
/*Declara variables correccionX y correccionY e inicialízalas en 0.*/
let correccionY = 0;
// Marca el nuevo punto
let nuevaPosicionX = 0;
/*Declara variables nuevaPosicionX y nuevaPosicionY e inicialízalas en 0.*/
let nuevaPosicionY = 0;
/*Obtiene la posición del elemento <canvas> en relación con la ventana gráfica y act
let posicion = miCanvas.getBoundingClientRect();
correccionX = posicion.x;
correccionY = posicion.y;
/*Establece el ancho y la altura del elemento <canvas> en 500 píxeles.*/
miCanvas.width = 500;
miCanvas.height = 500;
```

Ilustración 8 CREACION DE VARIABLES DE CODIGO DE JAVASCRIPT

Posteriormente comenzamos con las funciones estas funciones nos ayudaran a un procedimiento un conjunto de instrucciones que realiza una tarea o calcula un valor, pero para que un procedimiento califique como función, debe tomar alguna entrada y devolver una salida donde hay alguna relación obvia entre la entrada y la salida, en este procedimiento comenzaremos a definir las siguientes funciones utilizadas en el leguaje de JavaScript en las cuales utilizamos, (“empezarDibujo, dibujarLinea, guardarLinea, pararDibujar”).

2. Función empezarDibujo:

- Se activa cuando el usuario comienza a dibujar arrastrando el mouse.
- Agrega un nuevo array vacío a lineas para almacenar la nueva línea.
- Llama a la función dibujarLinea para comenzar a dibujar la línea.

3. Función guardarLinea:

- Agrega un nuevo objeto a la última línea del array lineas. Este objeto contiene las coordenadas x e y de la posición actual del mouse.

4. Función dibujarLinea:

- Obtiene las coordenadas del mouse (nuevaPosicionX y nuevaPosicionY).
- Obtiene el contexto de dibujo 2D del elemento <canvas> (ctx).
- Establece los estilos de línea (grosor, estilo de unión y estilo de tapa).
- Establece el color de la línea a negro.
- Llama a guardarLinea para almacenar la nueva posición del mouse.
- Borra el lienzo (ctx.clearRect).
- Recorre el array lineas y dibuja cada línea:
- Inicia un nuevo camino (ctx.beginPath).
- Mueve el cursor a la primera posición de la línea (ctx.moveTo).
- Dibuja líneas a cada punto subsecuente de la línea (ctx.lineTo).
- Aplica el trazo a la línea (ctx.stroke).

```
50  /**
51   * Funcion que empieza a dibujar la linea
52   */
53   function empezarDibujo(event) {
54     lineas.push([ ]);
55     dibujarLinea(event);
56   }
57
58 /**
59  * Funcion que guarda la posicion de la nueva linea
60  */
61   function guardarLinea() {
62     lineas[lineas.length - 1].push({
63       x: nuevaPosicionX,
64       y: nuevaPosicionY
65     });
66
67 /**
68  * Funcion dibuja la linea
69  */
70   function dibujarLinea(event) {
71     nuevaPosicionX = event.clientX - correccionX;
72     nuevaPosicionY = event.clientY - correccionY;
```

Ilustración 9 FUNCIONES DE EL CODIGO DE JAVASCRIPT

```

70      */
71      function dibujarLinea(event) {
72          nuevaPosicionX = event.clientX - correccionX;
73          nuevaPosicionY = event.clientY - correccionY;
74
75          let ctx = miCanvas.getContext('2d');
76          // Estilos de linea
77          ctx.lineJoin = ctx.lineCap = 'round';
78          ctx.lineWidth = 10;
79          // Color de la linea
80          ctx.strokeStyle = 'black';
81
82          // Guarda la linea
83          guardarLinea();
84          // Redibuja todas las lineas guardadas
85          ctx.clearRect(0, 0, miCanvas.width, miCanvas.height);
86          lineas.forEach(function (segmento) {
87              ctx.beginPath();
88              ctx.moveTo(segmento[0].x, segmento[0].y);
89              segmento.forEach(function (punto, index) {
90                  ctx.lineTo(punto.x, punto.y);
91              });
92              ctx.stroke();
93          });

```

Ilustración 10 FUNCIONES DE EL CODIGO DE JAVASCRIPT

Continuaremos con los eventos estos eventos son encargados de producir una señal de cierto tipo cuando un evento ocurre, y proporciona un mecanismo para que una acción se lleve a cabo (ejecutar código) de forma automática cuando el evento ocurra. Los eventos se lanzan dentro de la ventana del navegador y usualmente están asociados a un elemento en específico dentro de dicha ventana. Esto puede ser un solo elemento, un grupo de elementos, el documento HTML cargado la pestaña actual, o la ventana del navegador en su totalidad. Existen distintos tipos de eventos que pueden ocurrir.

5. Eventos:

- Se agrega un escuchador de eventos mousemove al elemento <canvas> para activar las funciones empezarDibujo cuando el usuario mueva el mouse dentro del lienzo.

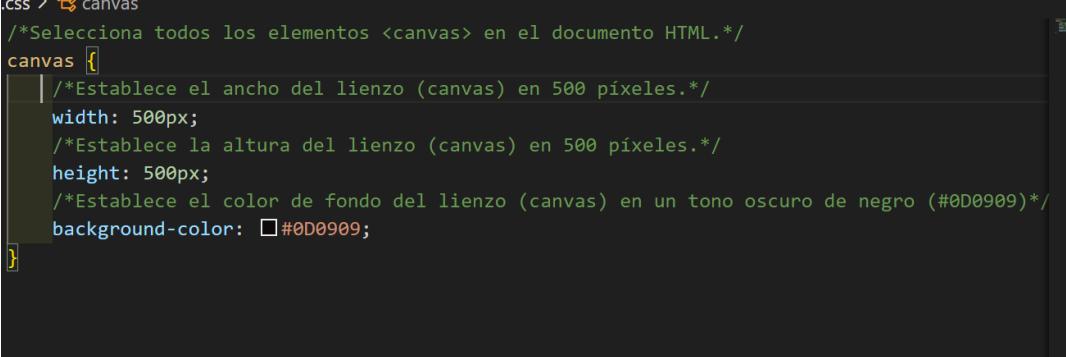
```

96      //=====
97      // EVENTOS
98      //=====
99
100     // Eventos raton
101     miCanvas.addEventListener('mousemove', empezarDibujo, false);
102
103 </script>
104 </body>
105 </html>

```

Ilustración 11 LOS EVENTOS DEL CODIGO DE JAVASCRIPT

Después de definir el código de JavaScript continuaremos con el código de css este para darle diseño a la página que este será el diseño al elemento canvas, en el cual se le otorgara un ancho de deslizo que será una cantidad de 500 px, continuando también se le agregará una altura este de deslizo que también será por igual con una medición de 500px, por último se le agregara un color de fondo que este será un color negro



```
.css > canvas
/*Selecciona todos los elementos <canvas> en el documento HTML.*/
canvas {
    /*Establece el ancho del lienzo (canvas) en 500 píxeles.*/
    width: 500px;
    /*Establece la altura del lienzo (canvas) en 500 píxeles.*/
    height: 500px;
    /*Establece el color de fondo del lienzo (canvas) en un tono oscuro de negro (#0D0909)*/
    background-color: #0D0909;
}
```

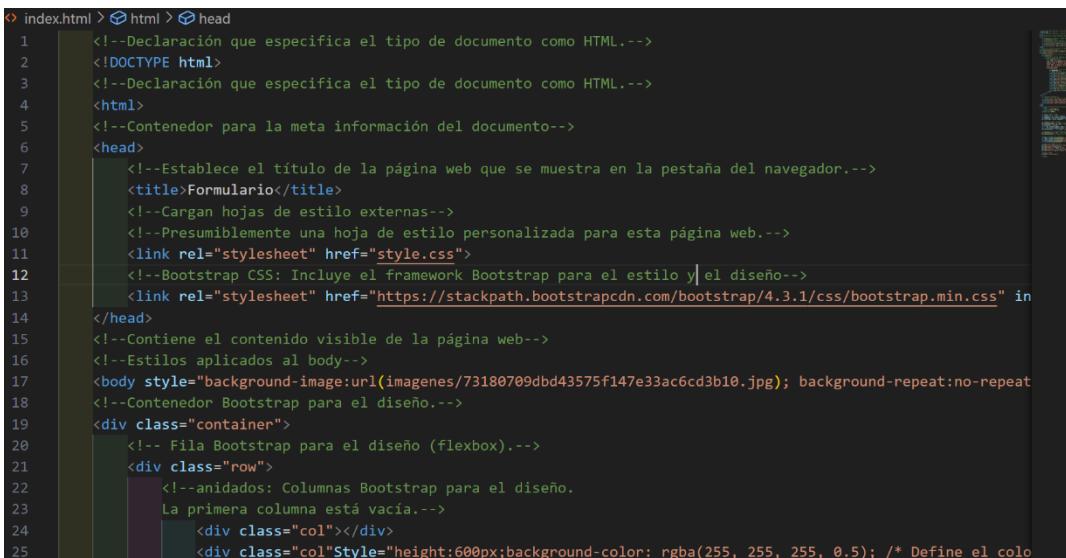
Ilustración 12 ESTILOS DE CSS

PRÁCTICA 3: FORMULARIO EN PDF

Para la práctica siguiente se requiere establecer un código HTML y JavaScript crea un formulario web simple que permite a los usuarios ingresar su nombre, apellido, correo electrónico y RFC. El formulario incluye un botón que, al hacer clic, genera un documento PDF con los datos ingresados. El código está estructurado y utiliza comentarios para explicar claramente su funcionalidad. Comenzaremos con la estructura básica de HTML en la cual es la estructura de nuestra página web.

HTML:

- ✓ <!DOCTYPE html>, que indica al navegador que el documento es un archivo HTML.
- ✓ **Etiqueta <html>**: Abre la raíz del documento HTML y especifica el idioma principal del contenido como español.
- ✓ **Sección <head>**: Contiene metadatos e información sobre el documento:
- ✓ **Título de la página**: Establece el título de la página web como "Formulario".
- ✓ Enlaces a hojas de estilo:
- ✓ **Estilo personalizado**: Incluye una hoja de estilo externa llamada "style.css" para aplicar estilos personalizados a la página.
- ✓ **Bootstrap CSS**: Incluye el framework Bootstrap para el estilo y el diseño.
- ✓ **Sección <body>**: Contiene el contenido visible de la página web:
- ✓ Estilos aplicados al body: Define estilos como imagen de fondo, color de fondo, tipo de letra y tamaño de letra para el cuerpo del documento.
- ✓ **Contenedor Bootstrap**: Envuelve todo el contenido de la página para facilitar el diseño con Bootstrap.
- ✓ **Fila Bootstrap**: Crea una fila para organizar el contenido en dos columnas.



```
index.html > html > head
1  <!--Declaración que especifica el tipo de documento como HTML.-->
2  <!DOCTYPE html>
3  <!--Declaración que especifica el tipo de documento como HTML.-->
4  <html>
5  <!--Contenedor para la meta información del documento-->
6  <head>
7  <!--Establece el título de la página web que se muestra en la pestaña del navegador.-->
8  <title>Formulario</title>
9  <!--Cargan hojas de estilo externas-->
10 <!--Presumiblemente una hoja de estilo personalizada para esta página web.-->
11 <link rel="stylesheet" href="style.css">
12 <!--Bootstrap CSS: Incluye el framework Bootstrap para el estilo y el diseño-->
13 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" in
14 </head>
15 <!--Contiene el contenido visible de la página web-->
16 <!--Estilos aplicados al body-->
17 <body style="background-image:url(imagenes/73180709dbd43575f147e33ac6cd3b10.jpg); background-repeat:no-repeat
18 <!--Contenedor Bootstrap para el diseño.-->
19 <div class="container">
20 <!-- Fila Bootstrap para el diseño (flexbox).-->
21 <div class="row">
22 <!---anidados: Columnas Bootstrap para el diseño.
23 La primera columna está vacía.-->
24 <div class="col"></div>
25 <div class="col" style="height:600px;background-color: rgba(255, 255, 255, 0.5); /* Define el colo
```

Ilustración 13 ESTRUCTURA BASICA DE HTML

Columna vacía: La primera columna está vacía para crear espacio entre el borde de la página y el contenido.

Columna del formulario: La segunda columna contiene el formulario.

Estilos aplicados a la columna: Define un color de fondo, un borde, un relleno y un ancho para la columna del formulario.

Formulario: El formulario se incluye dentro de un elemento <form> con el ID "myfrom".

Encabezado del formulario: Un elemento <h1> con el texto "FORMULARIO" como título del formulario.

Campos de entrada:

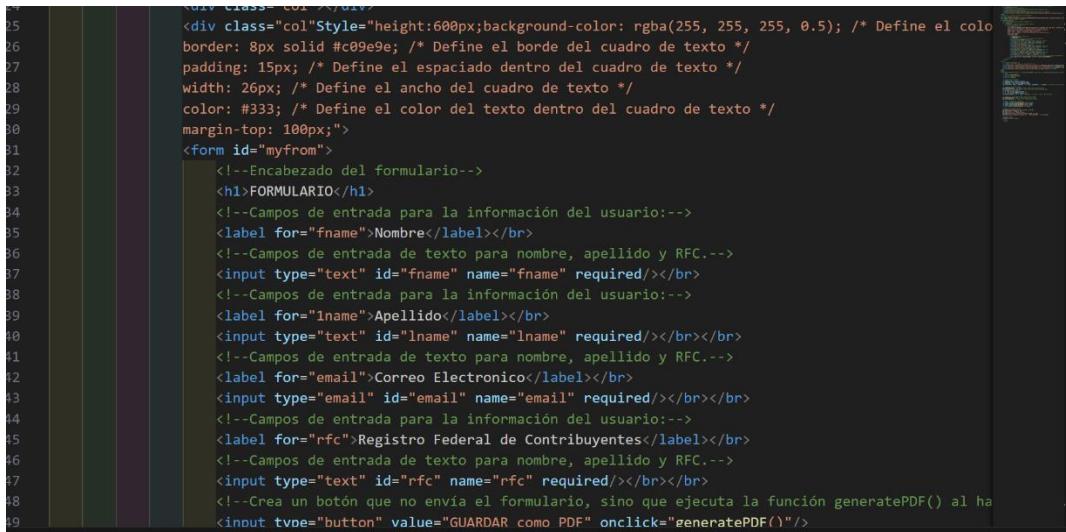
Nombre: Un campo de entrada de texto con el ID "fname" y el atributo required para que sea obligatorio.

Apellido: Un campo de entrada de texto con el ID "lname" y el atributo required para que sea obligatorio.

Correo electrónico: Un campo de entrada de correo electrónico con el ID "email" y el atributo required para que sea obligatorio.

RFC: Un campo de entrada de texto con el ID "rfc" y el atributo required para que sea obligatorio.

Botón de envío: Un botón con el texto "GUARDAR como PDF" que no envía el formulario, sino que ejecuta la función generatePDF() al hacer clic.



```
25      <div class="col"></div>
26
27      <div class="col" style="height:600px;background-color: rgba(255, 255, 255, 0.5); /* Define el color de fondo de la columna */
28          border: 8px solid #c09e9e; /* Define el borde del cuadro de texto */
29          padding: 15px; /* Define el espaciado dentro del cuadro de texto */
30          width: 26px; /* Define el ancho del cuadro de texto */
31          color: #333; /* Define el color del texto dentro del cuadro de texto */
32          margin-top: 100px;">
33
34      <form id="myfrom">
35          <!-- Encabezado del formulario--&gt;
36          &lt;h1&gt;FORMULARIO&lt;/h1&gt;
37          <!-- Campos de entrada para la información del usuario:--&gt;
38          &lt;label for="fname"&gt;Nombre&lt;/label&gt;&lt;br&gt;
39          &lt;!-- Campos de entrada de texto para nombre, apellido y RFC.--&gt;
40          &lt;input type="text" id="fname" name="fname" required/&gt;&lt;br&gt;
41          &lt;!-- Campos de entrada para la información del usuario:--&gt;
42          &lt;label for="lname"&gt;Apellido&lt;/label&gt;&lt;br&gt;
43          &lt;input type="text" id="lname" name="lname" required/&gt;&lt;br&gt;&lt;br&gt;
44          &lt;!-- Campos de entrada de texto para nombre, apellido y RFC.--&gt;
45          &lt;label for="email"&gt;Correo Electronico&lt;/label&gt;&lt;br&gt;
46          &lt;input type="email" id="email" name="email" required/&gt;&lt;br&gt;&lt;br&gt;
47          &lt;!-- Campos de entrada para la información del usuario:--&gt;
48          &lt;label for="rfc"&gt;Registro Federal de Contribuyentes&lt;/label&gt;&lt;br&gt;
49          &lt;!-- Campos de entrada de texto para nombre, apellido y RFC.--&gt;
50          &lt;input type="text" id="rfc" name="rfc" required/&gt;&lt;br&gt;&lt;br&gt;
51          &lt;!-- Crea un botón que no envía el formulario, sino que ejecuta la función generatePDF() al hacer clic--&gt;
52          &lt;input type="button" value="GUARDAR como PDF" onclick="generatePDF()"/&gt;</pre>
```

Ilustración 14 CAMPOS GENERADOS PARA CREAR EL FORMULARIO

Posteriormente vamos a agregar el código de JavaScript esto se lleva a los siguientes elementos; en los cuales vamos a incluir lo siguiente ofrecer a los usuarios, la posibilidad de descargar un archivo PDF, con información específica de su interés, puede ser de enorme utilidad. Resumiendo, técnicamente, una funcionalidad así, implica **generar el archivo de forma dinámica**, bajo petición. Y durante mucho tiempo, hacer esto, se ha considerado competencia exclusiva de entornos backend.

JavaScript:

Librería jsPDF: Se incluye la librería jsPDF para generar documentos PDF.

Función generatePDF ():

1. Crea una instancia de la clase jsPDF.
2. Configura el margen del documento.
3. Agrega una marca de agua al documento con el texto "ALCANTARA" en el centro de la página.
4. Restablece los estilos de texto predeterminados.
5. Obtiene los valores de los campos de entrada del formulario: nombre, apellido, correo electrónico y RFC.
6. Agrega los datos del formulario al documento PDF:
7. Título: "Datos Registrados Correctamente:".
8. Nombre: "Nombre: [nombre ingresado]".
9. Apellido: "Apellido: [apellido ingresado]".
10. Correo electrónico: "Dirección de Correo Electrónico: [correo electrónico ingresado]".
11. RFC: "Número de Registro Federal de Contribuyentes: [RFC ingresado]".
12. Guarda el documento PDF con el nombre "Formulario.pdf".

```
<!--El código JavaScript define la función generatePDF() para crear un documento PDF utilizando la librería j
<script>
    function generatePDF(){
        //crea una instancia de jsPDF
        var doc = new jsPDF();

        // Configuración de margen
        var margin = 10; // Margen en pixeles
        var pageWidth = doc.internal.pageSize.width;
        var pageHeight = doc.internal.pageSize.height;
        doc.rect(margin, margin, pageWidth - 2 * margin, pageHeight - 2 * margin); // Dibuja el rectángulo del ma

        // Configurar la marca de agua
        doc.setFontSize(40); // Tamaño de fuente grande para la marca de agua
        doc.setTextColor(200, 200, 200); // Color gris claro para la marca de agua
        var watermark = 'ALCANTARA';
        var x = doc.internal.pageSize.width / 2;
        var y = doc.internal.pageSize.height / 2;
        doc.text(watermark, x, y, { align: 'center' }); // Centrar y rotar la marca de agua

        // Restablecer las configuraciones para el texto normal
        doc.setTextColor(0, 0, 0); // Color negro para el texto
        doc.setFontSize(16); // Tamaño de la fuente para el texto
```

Ilustración 15 GERANDO LAS FUNCIONES DEL PDF EN EL CODIGO DE JAVASCRIPT

```

//obtener los datos del formulario
var fname = document.getElementById("fname").value;
var lname =document.getElementById("lname").value;
var email= document.getElementById("email").value;
var rfc =document.getElementById("rfc").value;

//Agregar los elementos al pdf
doc.text("Datos Registrados Correctamente: ", 55,20);
doc.text("Nombre:"+fname,15,35);
doc.text("Apellido:"+ lname, 15, 45);
doc.text("Direccion de Correo Electronico"+ email, 15, 55);
doc.text("Numero de Registro Federal de" + "Contribuyentes:" + rfc, 15, 65);

//Guardar el PDF
doc.save("Formulario.pdf");
}
</script>

```

Ilustración 16 OBTECIONDE DATOS DEL FORMULARIO Y AGRAFGANDO LOS ELEMNTOS AL PDF

Reporte detallado de los estilos CSS

Este código CSS define estilos para los elementos <h1>, input [type="text"], input [type="email"] e input [type="button"] en un documento HTML. Los estilos se aplican utilizando selectores CSS específicos para cada tipo de elemento, comenzamos con el primer elemento que es con <h1>, en seguida seguimos con el impar para establecer el color de fondo, Proporciona estilos básicos para mejorar la apariencia del cuerpo de la página.

Estilo para h1: text-align: center; Centra el texto de todos los elementos <h1> en la página.

Estilo para input[type="text"] e input[type="email"]:

Fondo: background-color: aliceblue; Establece un color de fondo azul claro para los campos de entrada de texto y correo electrónico.

Borde: border: 2px solid black; Elimina el borde predeterminado de los campos de entrada.

border-bottom: 2px solid #fff; Agrega un borde inferior blanco sólido para simular un efecto de profundidad.

Radio de bordes: border-radius: 20px; Aplica un borde ligeramente redondeado a los campos de entrada.

Relleno: padding: 10px; Agrega espacio interno (padding) a los campos de entrada para que el texto no esté pegado a los bordes.

Margen inferior: margin-bottom: 1rem; Agrega un margen inferior a cada campo de entrada para separarlos.

Ancho: width: calc(100% - 90px); Establece el ancho de los campos de entrada para que ocupen casi todo el ancho del formulario, restando 90 píxeles de padding y bordes.

Comentarios: Los comentarios explican la función de cada propiedad CSS.

Estilo para input[type="button"]:

Fondo: background: rgba(255, 255, 255, 0.85); Establece un fondo blanco semitransparente para los botones.

Borde: border: 1px solid rgba(255, 255, 255, 0.6); Agrega un borde blanco con transparencia a los botones.

Radio de bordes: border-radius: 20px; Aplica un borde ligeramente redondeado a los botones.

Relleno: padding: 10px 20px; Agrega espacio interno (padding) a los botones para que el texto no esté pegado a los bordes.

Cursor: cursor: pointer; Cambia el cursor a la forma de una mano al pasar sobre el botón.

Margen:

margin-right: 10px; Agrega un margen derecho a los botones para separarlos horizontalmente.

margin-left: 170px; Agrega un margen izquierdo a los botones para ubicarlos en la página.

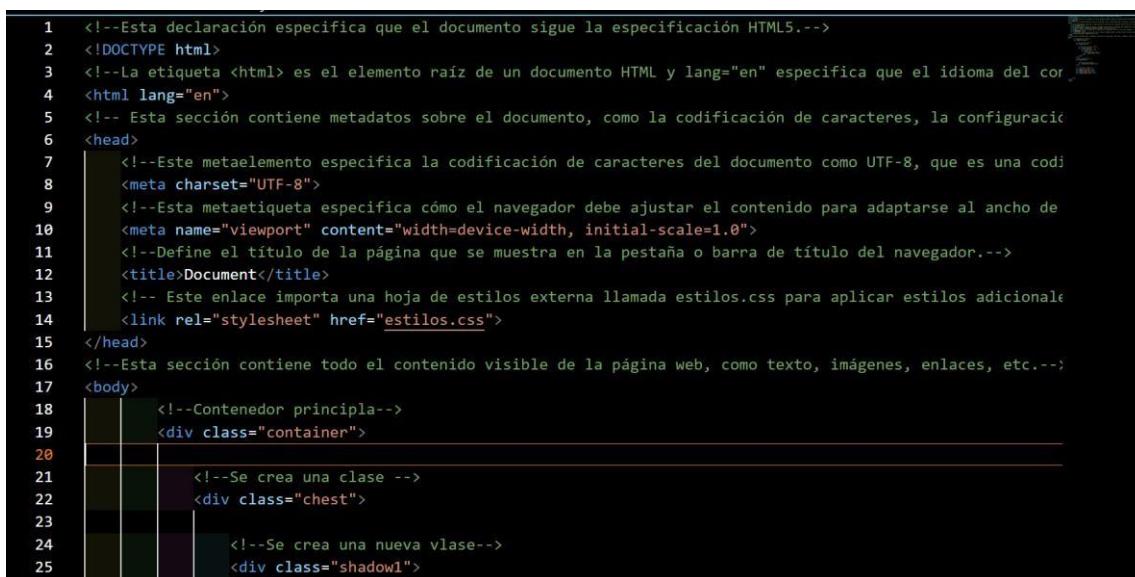
Comentarios: Los comentarios explican la función de cada propiedad CSS.

```
/*Selecciona todos los elementos <h1> (encabezado de nivel 1) en el documento HTML.  
Aplica los siguientes estilos dentro de las llaves */  
h1{  
    text-align: center;  
}  
/*Este es un selector compuesto que usa comas para apuntar a dos tipos de entrada diferentes:  
input[type="text"] : Selecciona todos los elementos <input> con el atributo type establecido en "text" (campos  
input[type="text"],  
input[type="email"] {  
    background-color: #aliceblue; /* Fondo de los inputs ligeramente más opaco que el formulario */  
    border: 2px solid black; /* Sin bordes */  
    border-bottom: 2px solid #fff; /* Borde inferior simulando un efecto de profundidad */  
    border-radius: 20px; /* Bordes ligeramente redondeados */  
    padding: 10px; /* Espacio interno para los textos */  
    margin-bottom: 1rem; /* Espacio entre los campos de texto */  
    width: calc(100% - 90px); /* Anchura completa menos el padding */  
    /* margin-left: 40px; */  
}  
  
/*Este selector CSS selecciona todos los elementos <input> en el documento HTML donde el atributo type está e  
input[type="button"] {  
    background: #rgba(255, 255, 255, 0.85); /* Fondo de los botones menos transparente */  
    border: 1px solid #rgba(255, 255, 255, 0.6); /* Borde con transparencia */  
    border-radius: 20px; /* Bordes redondeados */
```

Ilustración 17 CREANDO LOS DISEÑO DE LA PAGINA

PRÁCTICA 4: AMONGOS EN PDF

En esta actividad comenzamos con la práctica 4, el propósito de esta práctica es practicar las figuras con un personaje de un video juego llamado amagos, esta práctica se requiere estilar al personaje del video juego, al igual que convertirlo en pdf y personalizarlo a un gusto personal, inicializaremos con la estructura básica que tenemos en html5,y creando el link del estilo css, esto para agregarle estilos al personaje del video juego, posteriormente que ya había creado la estructura básica y la hoja de estilos de css agregue un contenedor principal llamado "conteiner".



```
1 <!--Esta declaración especifica que el documento sigue la especificación HTML5.-->
2 <!DOCTYPE html>
3 <!--La etiqueta <html> es el elemento raíz de un documento HTML y lang="en" especifica que el idioma del contenido es inglés-->
4 <html lang="en">
5 <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la configuración del navegador, etc. -->
6 <head>
7   <!--Este metaelemento especifica la codificación de caracteres del documento como UTF-8, que es una codificación ampliamente utilizada-->
8   <meta charset="UTF-8">
9   <!--Esta metaetiqueta especifica cómo el navegador debe ajustar el contenido para adaptarse al ancho de la pantalla-->
10  <meta name="viewport" content="width=device-width, initial-scale=1.0">
11  <!--Define el título de la página que se muestra en la pestaña o barra de título del navegador.-->
12  <title>Document</title>
13  <!-- Este enlace importa una hoja de estilos externa llamada estilos.css para aplicar estilos adicionales-->
14  <link rel="stylesheet" href="estilos.css">
15 </head>
16 <!--Esta sección contiene todo el contenido visible de la página web, como texto, imágenes, enlaces, etc.-->
17 <body>
18   <!--Contenedor principal-->
19   <div class="container">
20     <!--Se crea una clase -->
21     <div class="chest">
22       <!--Se crea una nueva clase-->
23       <div class="shadow1">
24         <!--Se crea una clase-->
25       </div>
26     </div>
27   </div>
28 </body>
29 </html>
```

Ilustración 63 creación de conteiner en HTML5

Este contenedores se utilizan comúnmente para agrupar contenido y aplicar estilos a ese contenido de manera uniforme, posteriormente agregamos un `<div class="chest">`: Dentro del contenedor principal, hay otro `<div>` con una clase llamada "chest". Este elemento representa algo que tiene un cofre.

`<div class="shadow1">`: Dentro del contenedor "chest", hay otro `<div>` con una clase llamada "shadow1". Esto representa la sombra del objeto del cofre.

`<div class="eye">`: Dentro del contenedor "shadow1", hay otro `<div>` con una clase llamada "eye". Esto representar un ojo.

`<div class="white"></div>`: Dentro del contenedor "eye", hay otro `<div>` con una clase llamada "white". Este `<div>` representa el color blanco, que podría ser el blanco del ojo.

```

20
21      <!--Se crea una clase -->
22      <div class="chest">
23
24          <!--Se crea una nueva clase-->
25          <div class="shadow1">
26              <!--Se crea otra clase -->
27              <div class="eye">
28                  <!--Se crea otra clase -->
29                  <div class="white"></div>
30
31          </div>
32      </div>

```

Ilustración 64 elementos que se aplicaran en css

Este código HTML está estructurado utilizando elementos `<div>` y aplicando clases CSS a cada elemento para aplicar estilos específicos. Las clases CSS se utilizan para asignar estilos a los elementos y pueden definirse en una hoja de estilos CSS separada para mantener el código HTML más limpio y modular. En seguida se creó otra clase `<div class="back">`. Este `<div>` representa la parte posterior del objeto que estás creando y tiene una clase llamada "back". Posteriormente Dentro del contenedor "back", hay otro `<div>` con una clase llamada "back-shadow" que representa la sombra de la parte posterior del objeto.

En seguida vamos a representar una pierna izquierda con la clase `<div class="left-leg"></div>` al igual vamos a representar la otra pierna derecha y tiene una clase llamada "right-leg". En seguida vamos a crear dos clases, la primera de ellas se llamará `<div class="middle-leg"></div>`: Este `<div>` representa una pierna media y tiene una clase llamada "middle-leg" y la segunda clase se llamará similar `<div class="middle-leg2"></div>`: Este `<div>` también representa una pierna media, pero con una clase diferente llamada "middle-leg2". Estas dos clases tengan estilos ligeramente diferentes o se utilicen para representar partes diferentes del objeto.

```

31      </div>
32  </div>
33  <!--Se crea una clase -->
34  <div class="back">
35
36      <!--Se crea otra clase -->
37      <div class="back-shadow"></div>
38  </div>
39
40  <!--Se crean 4 clases --->
41  <div class="left-leg"></div>
42  <div class="right-leg"></div>
43  <div class="middle-leg"></div>
44  <div class="middle-leg2"></div>
45
46
47

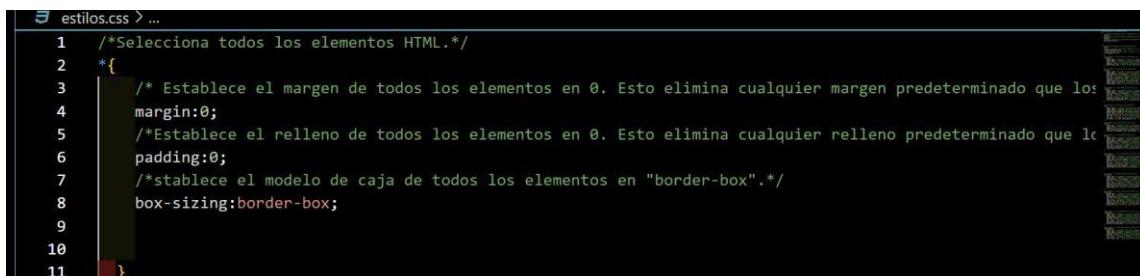
```

Ilustración 65clases para creación del cuerpo del dibujo

CODIGO DE CSS

Comenzando la estilacion de la práctica para el personaje vamos a comenzar utilizando el selector universal este elemento se representa con un *, seguido de un conjunto de reglas CSS, como primer elemento agregue el selector universal se aplica a todos los elementos HTML en el documento, posteriormente dentro del sector universal se establece el margen de todos los elementos en 0 píxeles. El margen es el espacio alrededor del borde de un elemento, representado con un “margin”, en seguida se agregó un relleno de todos los elementos en 0 píxeles. El relleno es el espacio dentro del borde de un elemento representado con un “padding”.

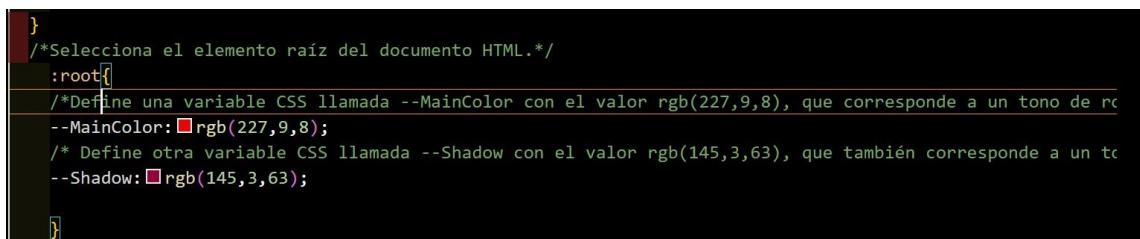
Por ultimo se estableció un box-sizing: border-box;: Establece el modelo de caja de todos los elementos como "border-box". En este modelo, el ancho y la altura de un elemento incluyen el relleno y el borde, pero no el margen. Esto facilita el control del tamaño total de un elemento y evita que el tamaño cambie cuando se agrega relleno o borde. Comúnmente al comienzo de un archivo de estilos para establecer márgenes, rellenos y el modelo de caja de todos los elementos en un documento, lo que proporciona un punto de partida consistente para estilos más específicos.



```
estilos.css > ...
1  /*Selecciona todos los elementos HTML.*/
2  *{
3      /* Establece el margen de todos los elementos en 0. Esto elimina cualquier margen predeterminado que los
4      margin:0;
5      /*Establece el relleno de todos los elementos en 0. Esto elimina cualquier relleno predeterminado que los
6      padding:0;
7      /*stabliece el modelo de caja de todos los elementos en "border-box".*/
8      box-sizing:border-box;
9
10
11 }
```

Ilustración 66 se utilizó un selector universal para establecer imagen que viene por defecto

Comenzamos con los estilos de css en este siguiente paso se establecerá un root que su función es selecciona el elemento raíz del documento, que es el elemento <html>. Al definir variables en: root, estas variables estarán disponibles globalmente en todo el documento CSS. Dentro de este selector se le agregara MainColor: rgb(227, 9, 8); esto con la finalidad de asignarle un color. Esta variable podría utilizarse para definir el color principal utilizado en la página.



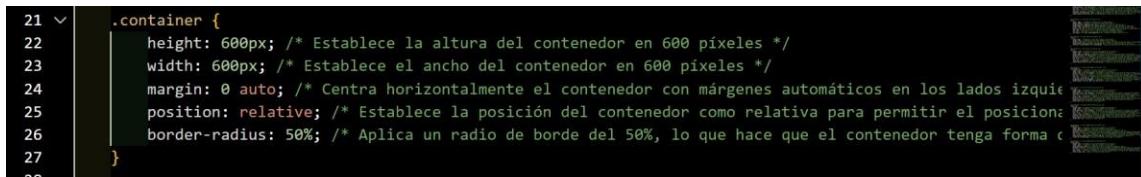
```
}
```

```
/*Selecciona el elemento raíz del documento HTML.*/
:root{
    /*Define una variable CSS llamada --MainColor con el valor rgb(227,9,8), que corresponde a un tono de ro
    --MainColor:rgb(227,9,8);
    /* Define otra variable CSS llamada --Shadow con el valor rgb(145,3,63), que también corresponde a un te
    --Shadow:rgb(145,3,63);
}
```

Ilustración 67 se estable el elemento de raíz de HTML5

Continuamos con definir estilos con la clase llamada conteiner, esta clase tendrá como elementos un height: 600px;; Establece la altura del contenedor en 600 píxeles, width: 600px;; Establece el ancho del contenedor en 600 píxeles. margin: 0 auto;; Esto centra horizontalmente el contenedor en el medio de su contenedor padre. El 0 representa que no hay margen en la parte superior e inferior, y auto establece un margen izquierdo y derecho automático, lo que distribuye igualmente el espacio a cada lado del contenedor. position: relative; Establece la posición del contenedor como relativa.

Y por último se agregara en la misma clase un elemento que será border-radius: 50%; Aplica un radio de borde de 50% al contenedor, lo que lo convierte en un círculo perfecto. Esta propiedad crea esquinas redondeadas al contenedor, con un radio igual a la mitad del ancho o altura del contenedor, lo que resulta en un círculo cuando el valor es del 50%. Estos estilos combinados crean un contenedor cuadrado con esquinas redondeadas, centrado horizontalmente en su contenedor padre y con una altura y ancho de 600 píxeles.



```
21 .container {  
22     height: 600px; /* Establece la altura del contenedor en 600 píxeles */  
23     width: 600px; /* Establece el ancho del contenedor en 600 píxeles */  
24     margin: 0 auto; /* Centra horizontalmente el contenedor con márgenes automáticos en los lados izquierdos */  
25     position: relative; /* Establece la posición del contenedor como relativa para permitir el posicionamiento absoluto de sus hijos */  
26     border-radius: 50%; /* Aplica un radio de borde del 50%, lo que hace que el contenedor tenga forma circular */  
27 }  
28
```

Ilustración 68 Estalación de contenedor

En seguida vamos a agregar la clase llamada `chest` esta clase tendrá sus elementos como primer elemento es `height: 268px;` y `width: 200px;` Establecen la altura y el ancho del elemento `.chest` en 268 píxeles y 200 píxeles respectivamente, en seguida agregamos un `background-color: var(--Shadow);` Establece el color de fondo del elemento `.chest` usando una variable CSS personalizada llamada `--Shadow`. La cual también vamos a agregar `top: 50%;` y `left: 50%;` Posiciona el elemento `.chest` en el 50% del tamaño del contenedor padre, tanto vertical como horizontalmente.

Dentro de la misma clase se agregara un `transform: translate(-50%, -50%);` Desplaza el elemento `.chest` hacia arriba y hacia la izquierda en un 50% de su propio ancho y altura, respectivamente. Esto se usa en conjunto con `top: 50%;` y `left: 50%;` para centrar el elemento exactamente en el centro del contenedor padre. Este elemento que sigue es importante que es `position: absolute;` Establece el posicionamiento del elemento `.chest` en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta. Y en la cual también le agregaremos `border-radius: 37% 55% 1% 0% / 44% 45% 36% 35%;` Establece radios de borde elípticos para crear una forma compleja. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo.

Y por últimos elementos serán agregados `border: 10px solid black;` Establece un borde sólido de 10 píxeles de ancho y color negro alrededor del elemento `.chest`.

`border-bottom: none;` Elimina el borde en la parte inferior del elemento `.chest`. En resumen, estos estilos combinados crean un elemento con la apariencia de un cofre con una forma y un diseño específicos.

```

/*Se manipula a la clase chest con estilos css*/
.chest {
    height: 268px; /* Establece la altura del elemento en 268 píxeles */
    width: 200px; /* Establece el ancho del elemento en 200 píxeles */
    background-color: var(--Shadow); /* Establece el color de fondo utilizando la variable CSS --Shadow */
    top: 50%; /* Posiciona el elemento en el 50% del desplazamiento vertical del contenedor primario */
    left: 50%; /* Posiciona el elemento en el 50% del desplazamiento horizontal del contenedor primario */
    transform: translate(-50%, -50%); /* Aplica una traslación del -50% tanto en el eje X como en el eje Y */
    position: absolute; /* Establece la posición del elemento como absoluta para que pueda ser posicionado */
    border-radius: 37% 55% 1% 0% / 44% 45% 36% 35%; /* Aplica un radio de borde no simétrico */
    border: 10px solid black; /* Establece un borde sólido de 10 píxeles de ancho y color negro */
    border-bottom: none; /* Elimina el borde inferior */
}

```

Ilustración 69 Estalación de clase chest en css

Este bloque de código CSS define estilos para una clase llamada .shadow1. Aquí está la descripción de cada propiedad:

height: 210px; y width: 170px: Establecen la altura y el ancho del elemento .shadow1 en 210 píxeles y 170 píxeles respectivamente.

background-color: var(--MainColor): Establece el color de fondo del elemento .shadow1 utilizando una variable CSS personalizada llamada --MainColor. top: 52%; y left: 54%: Posiciona el elemento .shadow1 en el 52% y 54% del tamaño del contenedor padre, respectivamente.

Position: absolute: Establece el posicionamiento del elemento .shadow1 en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta.

transform: translate(-50%, -63%): Desplaza el elemento .shadow1 hacia arriba y hacia la izquierda en un 50% y 63% de su propio ancho y altura, respectivamente. Esto se usa en conjunto con top: 52%; y left: 54%; para posicionar el elemento de manera precisa.

border-radius: 45% 88% 38% 42% / 73% 64% 82% 70%: Establece radios de borde elípticos para crear una forma compleja. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo.

En resumen, estos estilos combinados crean un elemento con la apariencia de una sombra con una forma y un diseño específicos.

```

43  /*Se manipula a la clase shadow1 con estilos css*/
44  .shadow1 {
45      height: 210px; /* Establece la altura del elemento en 210 píxeles */
46      width: 170px; /* Establece el ancho del elemento en 170 píxeles */
47      background-color: var(--MainColor); /* Establece el color de fondo utilizando la variable CSS --MainColor */
48      top: 52%; /* Posiciona el elemento en el 52% del desplazamiento vertical del contenedor primario */
49      left: 54%; /* Posiciona el elemento en el 54% del desplazamiento horizontal del contenedor primario */
50      position: absolute; /* Establece la posición del elemento como absoluta para que pueda ser posicionado */
51      transform: translate(-50%, -63%); /* Aplica una traslación del -50% en el eje X y -63% en el eje Y */
52      border-radius: 45% 88% 38% 42% / 73% 64% 82% 70%; /* Aplica un radio de borde no simétrico */
53  }
54

```

Ilustración 70 estilos de clase shadow1 en css

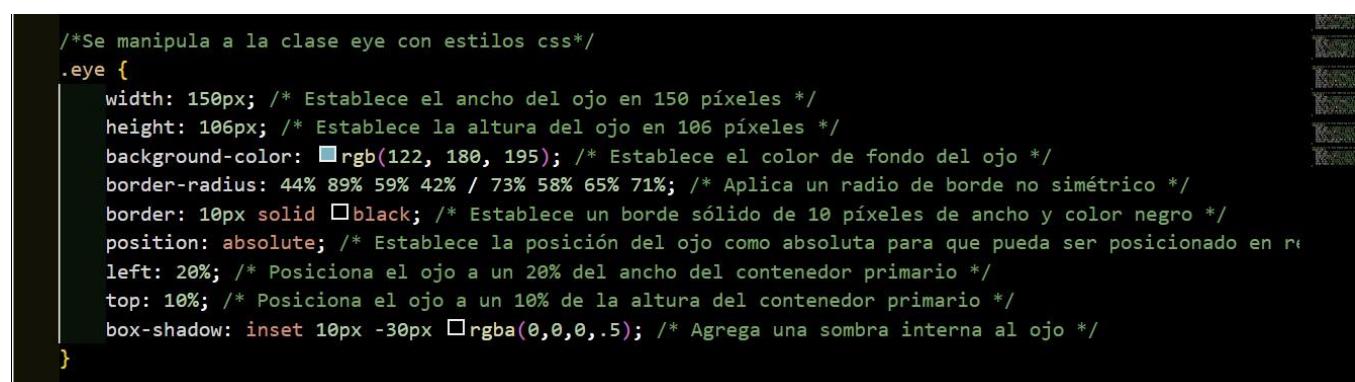
Este bloque de código CSS define estilos para una clase llamada .eye. Aquí está la descripción de cada propiedad:

width: 150px; y height: 106px;: Establecen el ancho y la altura del elemento .eye en 150 píxeles y 106 píxeles respectivamente.

background-color: rgb(122, 180, 195);: Establece el color de fondo del elemento .eye utilizando el valor RGB rgb(122, 180, 195).

border-radius: 44% 89% 59% 42% / 73% 58% 65% 71%;: Establece radios de borde elípticos para crear una forma compleja. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo.

border: 10px solid black;: Establece un borde sólido de 10 píxeles de ancho y color negro alrededor del elemento .eye



```
/*Se manipula a la clase eye con estilos css*/
.eye {
    width: 150px; /* Establece el ancho del ojo en 150 píxeles */
    height: 106px; /* Establece la altura del ojo en 106 píxeles */
    background-color: #rgb(122, 180, 195); /* Establece el color de fondo del ojo */
    border-radius: 44% 89% 59% 42% / 73% 58% 65% 71%; /* Aplica un radio de borde no simétrico */
    border: 10px solid #black; /* Establece un borde sólido de 10 píxeles de ancho y color negro */
    position: absolute; /* Establece la posición del ojo como absoluta para que pueda ser posicionado en relación a su contenedor */
    left: 20%; /* Posiciona el ojo a un 20% del ancho del contenedor primario */
    top: 10%; /* Posiciona el ojo a un 10% de la altura del contenedor primario */
    box-shadow: inset 10px -30px #rgba(0,0,0,.5); /* Agrega una sombra interna al ojo */
}
```

Ilustración 71 estilos de la clase eye en css

El bloque de código CSS que proporcionaste define estilos para una clase llamada .back. Aquí está la descripción de cada propiedad:

Height: 173px; y width: 62px: Establecen la altura y el ancho del elemento .back en 173 píxeles y 62 píxeles respectivamente. Border: 10px solid black: Establece un borde sólido de 10 píxeles de ancho y color negro alrededor del elemento .back.

background-color: var(--MainColor): Establece el color de fondo del elemento .back utilizando una variable CSS personalizada llamada --MainColor.

position: absolute;: Establece el posicionamiento del elemento .back en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta. top: 62.5%; y left: 21%: Posiciona el elemento .back en el 62.5% y 21% del tamaño del contenedor padre, respectivamente.

transform: translate(23%, -61%): Desplaza el elemento .back horizontalmente en un 23% de su propio ancho y verticalmente en un -61% de su propio alto. Esto se utiliza para ajustar la posición del elemento.

border-radius: 48% 0% 1% 45% / 44% 45% 36% 58%;: Establece radios de borde elípticos para crear una forma compleja. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente

orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo. En resumen, estos estilos combinados crean un elemento de un fondo con una forma y un diseño específicos.

El bloque de código CSS que proporcionaste define estilos para una clase llamada .back-shadow.

height: 133px; y width: 43px; Establecen la altura y el ancho del elemento .back-shadow en 133 píxeles y 43 píxeles respectivamente.

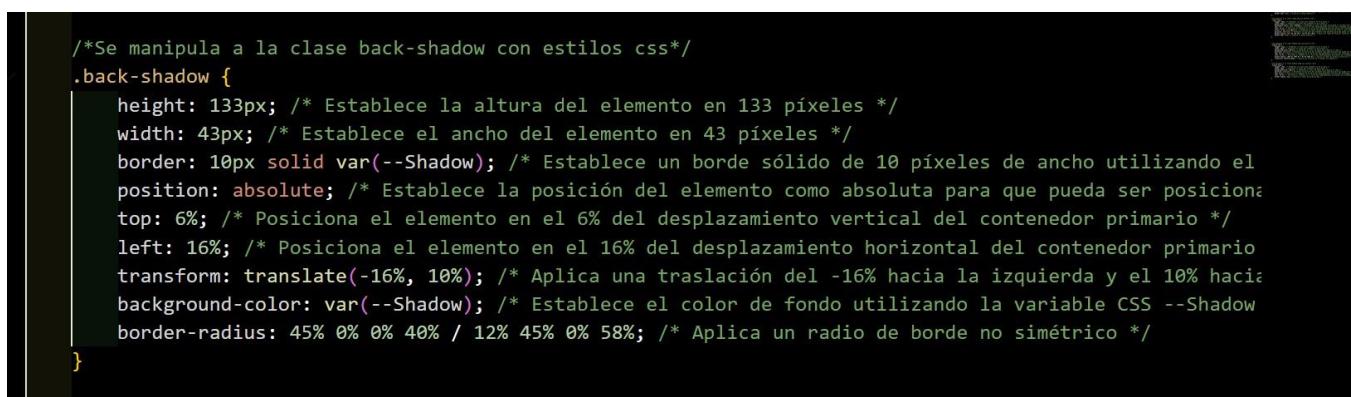
border: 10px solid var(--Shadow); Establece un borde sólido de 10 píxeles de ancho y el color definido por la variable CSS personalizada --Shadow alrededor del elemento .back-shadow.

position: absolute; Establece el posicionamiento del elemento .back-shadow en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta. top: 6%; y left: 16%; Posiciona el elemento .back-shadow en el 6% y 16% del tamaño del contenedor padre, respectivamente.

transform: translate(-16%, 10%); Desplaza el elemento .back-shadow hacia la izquierda en un 16% de su propio ancho y hacia abajo en un 10% de su propio alto. Esto se utiliza para ajustar la posición del elemento.

background-color: var(--Shadow); Establece el color de fondo del elemento .back-shadow utilizando una variable CSS personalizada llamada --Shadow.

border-radius: 45% 0% 0% 40% / 12% 45% 0% 58%; Establece radios de borde elípticos para crear una forma compleja. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo.



```
/*Se manipula a la clase back-shadow con estilos css*/
.back-shadow {
    height: 133px; /* Establece la altura del elemento en 133 píxeles */
    width: 43px; /* Establece el ancho del elemento en 43 píxeles */
    border: 10px solid var(--Shadow); /* Establece un borde sólido de 10 píxeles de ancho utilizando el
    position: absolute; /* Establece la posición del elemento como absoluta para que pueda ser posicionado */
    top: 6%; /* Posiciona el elemento en el 6% del desplazamiento vertical del contenedor primario */
    left: 16%; /* Posiciona el elemento en el 16% del desplazamiento horizontal del contenedor primario */
    transform: translate(-16%, 10%); /* Aplica una traslación del -16% hacia la izquierda y el 10% hacia abajo */
    background-color: var(--Shadow); /* Establece el color de fondo utilizando la variable CSS --Shadow */
    border-radius: 45% 0% 0% 40% / 12% 45% 0% 58%; /* Aplica un radio de borde no simétrico */
}
```

Ilustración 72 estilos de la clase bacck –shadow en css

El bloque de código CSS que proporcionaste define estilos para una clase llamada .left-leg.

height: 70px; y width: 84px; Establecen la altura y el ancho del elemento .left-leg en 70 píxeles y 84 píxeles respectivamente.

background-color: var(--Shadow); Establece el color de fondo del elemento .left-leg utilizando una variable CSS personalizada llamada --Shadow.

top: 71%; y left: 32.8%; Posiciona el elemento .left-leg en el 71% y 32.8% del tamaño del contenedor padre, respectivamente. border: 10px solid black; Establece un borde sólido de 10 píxeles de ancho y color negro alrededor del elemento .left-leg.

position: absolute;: Establece el posicionamiento del elemento .left-leg en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta.

border-radius: 10% 10% 38% 29% / 11% 0% 61% 100%;: Establece radios de borde elípticos para crear una forma compleja. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo. border-top: none;: Elimina el borde superior del elemento .left-leg.

En resumen, estos estilos combinados crean un elemento con la apariencia de una pierna izquierda con una forma y un diseño específicos.

```
/*Se manipula a la clase right-leg con estilos css*/
.right-leg {
    height: 70px; /* Establece la altura del elemento en 70 píxeles */
    width: 84px; /* Establece el ancho del elemento en 84 píxeles */
    background-color: var(--Shadow); /* Establece el color de fondo utilizando la variable CSS --Shadow */
    top: 71%; /* Posiciona el elemento en el 71% del desplazamiento vertical del contenedor primario */
    left: 52.6%; /* Posiciona el elemento en el 52.6% del desplazamiento horizontal del contenedor primario */
    border: 9.4px solid black; /* Establece un borde sólido de 9.4 píxeles de ancho y color negro */
    position: absolute; /* Establece la posición del elemento como absoluta para que pueda ser posicionado */
    border-radius: 10% 10% 38% 29% / 11% 0% 61% 100%; /* Aplica un radio de borde no simétrico */
    border-top: none; /* Elimina el borde superior */
```

Ilustración 73 estilos de right-leg de css

El bloque de código CSS que proporcionaste define estilos para una clase llamada .middle-leg. Aquí está la descripción de cada propiedad:

height: 13px; y width: 40px;: Establecen la altura y el ancho del elemento .middle-leg en 13 píxeles y 40 píxeles respectivamente. background-color: black; Establece el color de fondo del elemento .middle-leg en negro.

position: absolute;: Establece el posicionamiento del elemento .middle-leg en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta.

top: 70.4%; y left: 51.5%: Posiciona el elemento .middle-leg en el 70.4% y 51.5% del tamaño del contenedor padre, respectivamente.

border-radius: 100% 0% 100% 0% / 0% 0% 100% 0%: Establece radios de borde elípticos para crear una forma específica. Los números representan los radios de los bordes en porcentajes, y cada uno se aplica en el siguiente orden: superior izquierdo, superior derecho, inferior derecho, inferior izquierdo.

transform: rotate(-5deg): Aplica una rotación de -5 grados al elemento .middle-leg. Esto gira el elemento en sentido antihorario.

En resumen, estos estilos combinados crean un elemento con la apariencia de una pierna media con una forma y un diseño específicos, incluida una rotación.

```
/*Se manipula a la clase middle-leg con estilos css*/
.middle-leg {
    height: 13px; /* Establece la altura del elemento en 13 píxeles */
    width: 40px; /* Establece el ancho del elemento en 40 píxeles */
    background-color: black; /* Establece el color de fondo del elemento en negro */
    position: absolute; /* Establece la posición del elemento como absoluta para que pueda ser posicionado */
    top: 70.4%; /* Posiciona el elemento en el 70.4% del desplazamiento vertical del contenedor primario */
    left: 51.5%; /* Posiciona el elemento en el 51.5% del desplazamiento horizontal del contenedor primario */
    border-radius: 100% 0% 100% 0% / 0% 0% 100% 0%; /* Aplica un radio de borde no simétrico */
    transform: rotate(-5deg); /* Aplica una rotación de -5 grados al elemento */
}
```

Ilustración 74 Estilos de middle-leg en css

El bloque de código CSS que proporcionaste define estilos para una clase llamada .middle-leg2. Aquí está la descripción de cada propiedad:

height: 14px; y width: 40px;: Establecen la altura y el ancho del elemento .middle-leg2 en 14 píxeles y 40 píxeles respectivamente. background-color: black;: Establece el color de fondo del elemento .middle-leg2 en negro.

position: absolute;: Establece el posicionamiento del elemento .middle-leg2 en relación con su contenedor padre más cercano que tiene una posición relativa o absoluta.

top: 70.6%; y left: 45.5%;: Posiciona el elemento .middle-leg2 en el 70.6% y 45.5% del tamaño del contenedor padre, respectivamente.

border-radius: 5px;: Establece un radio de borde de 5 píxeles al elemento .middle-leg2. Esto crea esquinas redondeadas en el elemento.

En resumen, estos estilos combinados crean un elemento con la apariencia de una segunda pierna media con una forma y un diseño específicos.

```
/*Se manipula a la clase middle-leg2 con estilos css*/
.middle-leg2 {
    height: 14px; /* Establece la altura del elemento en 14 píxeles */
    width: 40px; /* Establece el ancho del elemento en 40 píxeles */
    background-color: black; /* Establece el color de fondo del elemento en negro */
    position: absolute; /* Establece la posición del elemento como absoluta para que pueda ser posicionado */
    top: 70.6%; /* Posiciona el elemento en el 70.6% del desplazamiento vertical del contenedor primario */
    left: 45.5%; /* Posiciona el elemento en el 45.5% del desplazamiento horizontal del contenedor primario */
    border-radius: 5px; /* Aplica un radio de borde de 5px */
}
```

Ilustración 75 estilos de middle-leg2 de css

CODIGO DE JAVASCRIPT

El script consta de dos partes principales: la asignación de un evento de clic a un botón y la generación del archivo PDF. Cuando se hace clic en el elemento con el ID 'generate-pdf', se selecciona el contenido del elemento con la clase 'container' y se convierte en un archivo PDF utilizando la biblioteca html2pdf. Este archivo PDF se guarda en el dispositivo del usuario.

Asignación del Evento de Clic:

`document.getElementById('generate-pdf').addEventListener('click', function() { ... });`: Este código asigna un evento de clic al elemento HTML con el ID 'generate-pdf'. Cuando este elemento se hace clic, se ejecutará la función anónima proporcionada.

Generación del PDF:

`var element = document.querySelector('.container');`: Este código selecciona el elemento HTML con la clase 'container' y lo almacena en la variable element.

`html2pdf().from(element).save();`: Utiliza la biblioteca html2pdf para convertir el contenido del elemento element en un archivo PDF y luego lo guarda. html2pdf() es un método proporcionado por la biblioteca html2pdf que permite iniciar el proceso de conversión. from(element) indica que se convertirá el contenido del elemento especificado. save() guarda el PDF generado.

```
<!-- Script para convertir a PDF -->
<!--Script JavaScript que se ejecuta cuando se hace clic en el botón "Generar PDF". Busca el contenido dentro
<script>
    document.getElementById('generate-pdf').addEventListener('click', function() {
        var element = document.querySelector('.container');
        html2pdf().from(element).save();
    });
</script>
```

Ilustración 18 CODIGO DE JAVASCRIPT PARA GENERALO EN PDF

PRÁCTICA 5: TABLAS DE MULTIPLICAR

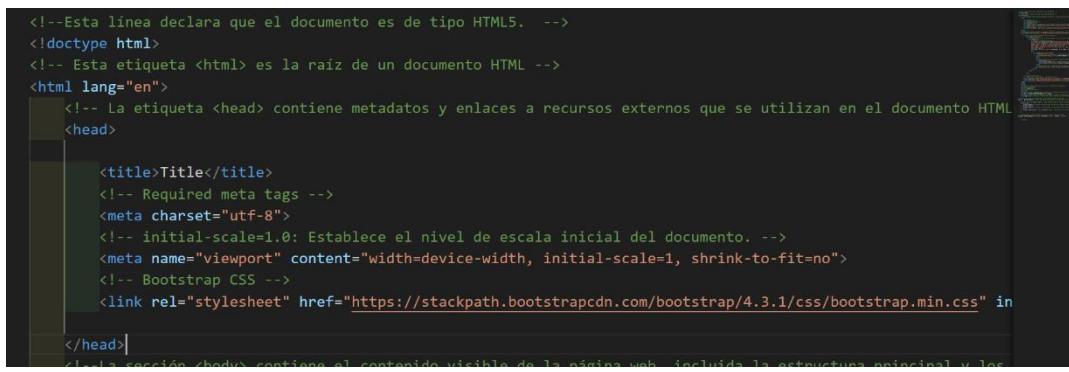
Práctica 5 consiste en una página web que muestra las tablas de multiplicar. Comenzaremos con HTML que son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, vídeos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc }

Empezamos con la declaración de código JAVASCRIPT

Script de JavaScript: Un enlace a un archivo JavaScript externo llamado "ciclos.js" (<script src="ciclos.js"></script>) que presumiblemente contiene la lógica para generar las tablas de multiplicar.

Librerías opcionales de JavaScript: Incluye las librerías jQuery, Popper.js y Bootstrap JS, aunque no se usen explícitamente en este código HTML.



```
<!--Esta línea declara que el documento es de tipo HTML. -->
<!doctype html>
<!-- Esta etiqueta <html> es la raíz de un documento HTML -->
<html lang="en">
    <!-- La etiqueta <head> contiene metadatos y enlaces a recursos externos que se utilizan en el documento HTML -->
    <head>
        <title>Title</title>
        <!-- Required meta tags -->
        <meta charset="utf-8">
        <!-- initial-scale=1.0: Establece el nivel de escala inicial del documento. -->
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1143x" crossorigin="anonymous">
    </head>
```

Ilustración 19 ESTRUCTURA BASICA DE HTML

Este código JavaScript solicita al usuario un número, genera la tabla de multiplicar de ese número y la muestra en la página web. Comenzaremos con solicitar el numero al usuario este para esta función utilizaremos un prompt esta función se muestra un cuadro de diálogo modal en el navegador que contiene un campo de entrada de texto y dos botones, generalmente "Aceptar" y "Cancelar". Y el bucle for nos ayuda a ejecutar un bloque de código varias veces. Su utilidad principal radica en la automatización de tareas repetitivas y en la iteración sobre estructuras de datos, como arreglos o listas

- La función `prompt()` se utiliza para mostrar un cuadro de diálogo que solicita al usuario que ingrese un número. El texto del mensaje es "Ingrese el número para generar la tabla de multiplicar:".
- El resultado de la función `prompt()` se almacena en la variable `tabla`.

BUCLE FOR PARA GENERAR LA TABLA:

El bucle `for` se ejecuta 9 veces, desde `i = 1` hasta `i = 9`.

Dentro del bucle, se realizan las siguientes acciones:

- Se multiplica el valor de la variable `tabla` por el valor actual del contador `i`.
- El resultado de la multiplicación se almacena en la variable `resultado`.
- Se utiliza la función `document.write()` para imprimir la siguiente línea en la página web:
- `tabla + "x" + i + "=" + resultado + "
"`
- Esta línea muestra la operación de multiplicación (por ejemplo, "2 x 3 = 6") y agrega un salto de línea (`
`) al final.

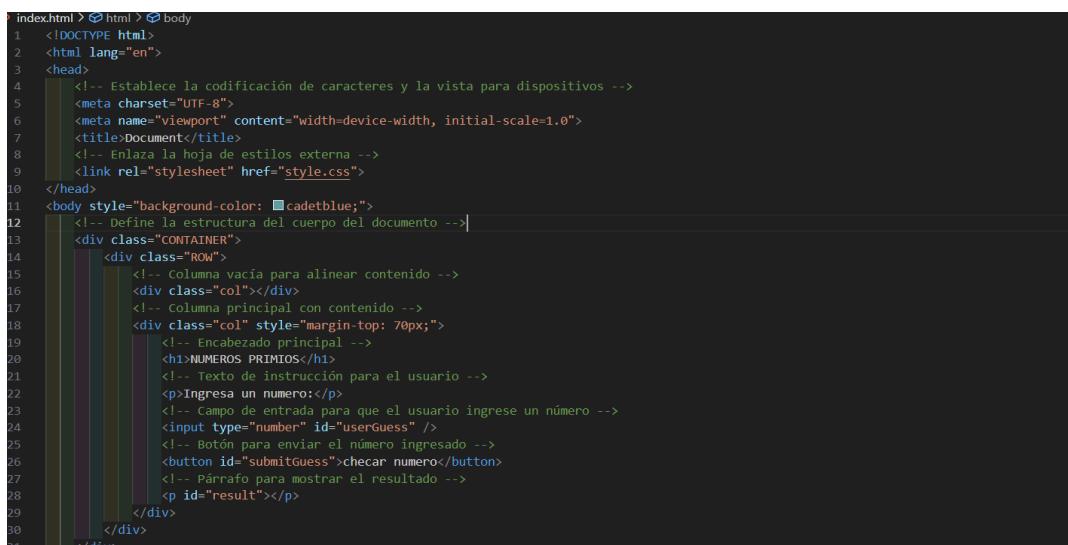
```
1 // Solicita al usuario que ingrese el número de la tabla de multiplicar
2 var tabla = prompt("Ingrese el número para generar la tabla de multiplicar:");
3
4 // Bucle for para generar la tabla de multiplicar del número ingresado
5 for (var i = 1; i < 10; i++) {
6     // Multiplica el número de la tabla por el número del ciclo actual e imprime el resultado
7     document.write(tabla + "x" + i + "=" + (tabla * i) + "<br>");
8 }
9
```

Ilustración 20 Código de JavaScript funciones que le añadimos

PRACTICA 6: NUMEROS PRIMOS

Practica 6 consiste en la creación de una página web que permite al usuario ingresar un número y verifica si es primo o no. La página utiliza Bootstrap para el diseño y JavaScript para la lógica de verificación. Comenzaremos con el lenguaje de HTML ya que los elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, videos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.



```
index.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <!-- Establece la codificación de caracteres y la vista para dispositivos -->
5      <meta charset="UTF-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Document</title>
8      <!-- Enlaza la hoja de estilos externa -->
9      <link rel="stylesheet" href="style.css">
10 </head>
11 <body style="background-color: #cadetblue;">
12     <!-- Define la estructura del cuerpo del documento -->
13     <div class="CONTAINER">
14         <div class="ROW">
15             <!-- Columna vacía para alinear contenido -->
16             <div class="col"></div>
17             <!-- Columna principal con contenido -->
18             <div class="col" style="margin-top: 70px;">
19                 <!-- Encabezado principal -->
20                 <h1>NUMEROS PRIMOS</h1>
21                 <!-- Texto de instrucción para el usuario -->
22                 <p>Ingresa un numero:</p>
23                 <!-- Campo de entrada para que el usuario ingrese un número -->
24                 <input type="number" id="userGuess" />
25                 <!-- Botón para enviar el número ingresado -->
26                 <button id="submitguess">checkar numero</button>
27                 <!-- Párrafo para mostrar el resultado -->
28                 <p id="result"></p>
29             </div>
30         </div>
31     </div>
```

Ilustración 21 estructura básica de HTML para la creación de la pagina

Continuando con la estructura del contenido vamos a empezar agregando un container este container nos ayudara a dividir la pantalla y hacer nuestras mediciones para ver qué espacio ocuparemos y en qué lugar queda mejor estructurado para esto vamos ocupar un row que este se encargara generar las columnas y las col se utilizaran para alinear el contenido este código nos ayuda bootstrap que ya que bootstrap trabaja con container, row y col.

Estructura del contenido:

Se utiliza un contenedor principal con la clase "CONTAINER" para organizar el contenido.

Dentro del contenedor, se crea una fila (<div class="ROW">) con dos columnas.

1. La primera columna está vacía (<div class="col"></div>) y se utiliza para alinear el contenido de la segunda columna.
2. La segunda columna (<div class="col">) contiene el contenido principal:
3. **Encabezado:** Un encabezado <h1> con el texto "NUMEROS PRIMIOS".
4. **Instrucción:** Un párrafo <p> que indica al usuario que ingrese un número.
5. **Campo de entrada:** Un campo de entrada de tipo "number" con el ID "userGuess" para que el usuario ingrese un número.
6. **Botón:** Un botón con el ID "submitGuess" y el texto "chechar numero" para enviar el número ingresado.
7. **Párrafo para el resultado:** Un párrafo <p> con el ID "result" para mostrar el resultado de la verificación (si el número es primo o no).

```
<div class="CONTAINER">
  <div class="ROW">
    <!-- Columna vacía para alinear contenido -->
    <div class="col"></div>
    <!-- Columna principal con contenido -->
    <div class="col" style="margin-top: 70px;">
      <!-- Encabezado principal -->
      <h1>NUMEROS PRIMIOS</h1>
      <!-- Texto de instrucción para el usuario -->
      <p>Ingresa un numero:</p>
      <!-- Campo de entrada para que el usuario ingrese un número -->
      <input type="number" id="userGuess" />
      <!-- Botón para enviar el número ingresado -->
      <button id="submitGuess">chechar numero</button>
      <!-- Párrafo para mostrar el resultado -->
      <p id="result"></p>
    </div>
  </div>
```

Ilustración 22 contenido de container de row y col

Eseguida vamos a ser el codigo de javaScrip esto lleva a cabo con utilizarionde las funciones al igual que los eventos y elementos comnezaremos con el primero de ellos que es la obtencionde los elementos del dom esto nos ayuadara métodos para obtener elementos del DOM (Document Object Model), la estructura que representa un documento HTML en el navegador. En seguida comenzaremos con el evento de el botón clic este su función es para interactuar con el usuario y capturar sus acciones y por ultimo contamos con las funciones este sirve para formar de implementar una función para verificar si un número es primo.

1. Obtener elementos del DOM:

Se utilizan las funciones getElementById() para obtener referencias a los elementos del DOM por su ID:

- ✓ userGuessInput: El campo de entrada de texto con el ID "userGuess".
- ✓ submitButton: El botón con el ID "submitGuess".
- ✓ resultParagraph: El párrafo con el ID "result".

2. Evento de clic en el botón:

- ✓ Se agrega un evento de clic al botón submitButton utilizando la función addEventListener().
- ✓ Cuando se hace clic en el botón, se ejecuta la función anónima que se pasa como argumento.

3. Función para verificar el número:

- ✓ La función anónima dentro del evento de clic realiza lo siguiente:
- ✓ Obtiene el valor ingresado por el usuario del campo de entrada userGuessInput y lo convierte a un número utilizando la función Number().
- ✓ Almacena el valor convertido en la variable userGuess.
- ✓ Utiliza la función isNaN() para verificar si el valor ingresado es un número válido.
- ✓ Si isNaN(userGuess) es true, significa que el valor no es un número válido y se muestra un mensaje de error en el párrafo resultParagraph.
- ✓ Si el valor es un número válido, se verifica si es primo:
- ✓ Si userGuess es divisible por 2 (userGuess % 2 === 0), significa que no es primo y se muestra un mensaje que indica que el número no es primo en el párrafo resultParagraph.
- ✓ Si userGuess no es divisible por 2, significa que es un número primo y se muestra un mensaje que indica que el número es primo en el párrafo resultParagraph.

```
<script>
    // Obtiene elementos del DOM por su ID
    const userGuessInput = document.getElementById("userGuess");
    const submitButton = document.getElementById("submitGuess");
    const resultParagraph = document.getElementById("result");

    // Agrega un evento de clic al botón de enviar
    submitButton.addEventListener("click", () => {
        // Obtiene el valor ingresado por el usuario y lo convierte a número
        const userGuess = Number(userGuessInput.value);
        // Verifica si el valor ingresado es un número válido
        if (isNaN(userGuess)) {
            // Si no es un número válido, muestra un mensaje de error
            resultParagraph.textContent = "numero invalido!";
        } else if (userGuess % 2 === 0) {
            // Si el número ingresado es divisible por 2, muestra que no es primo
            resultParagraph.textContent = `El NUMERO ${userGuess} NO ES PRIMO!`;
        } else {
            // Si el número ingresado no es divisible por 2, muestra que es primo
            resultParagraph.textContent = `EL NUMERO ${userGuess} ES PRIMO!`;
        }
    });
</script>
```

Ilustración 23 CODIGO DE JAVASCRIPT

Posteriormente vamos a comenzar con definir los estilos del cuerpo de la página, comenzaremos con la estilización de `body` que esto se refiere toda la página web que llevará estos estilos.

1. Estilos para el cuerpo de la página (`body`):

- `font-family: sans-serif;`: Establece la fuente para todo el texto del cuerpo de la página a una fuente sans-serif. Las fuentes sans-serif son aquellas que no tienen remates en los extremos de las letras, como Arial o Helvetica.
- `text-align: center;`: Centra el texto del cuerpo de la página horizontalmente.

2. Estilos para el párrafo de resultado (`#result`):

- `font-size: 24px;`: Establece el tamaño de la fuente del párrafo de resultado a 24 píxeles.
- `font-weight: bold;`: Hace que el texto del párrafo de resultado sea más grueso (negrita).
- `color: #00698f;`: Establece el color del texto del párrafo de resultado a un azul oscuro (#00698f).

3. Estilos para el encabezado principal (`h1`):

- `font-family: "Jersey 10 Charted", sans-serif;`: Establece la fuente para el encabezado principal a "Jersey 10 Charted". Si esta fuente no está disponible, se utilizará una fuente sans-serif predeterminada.

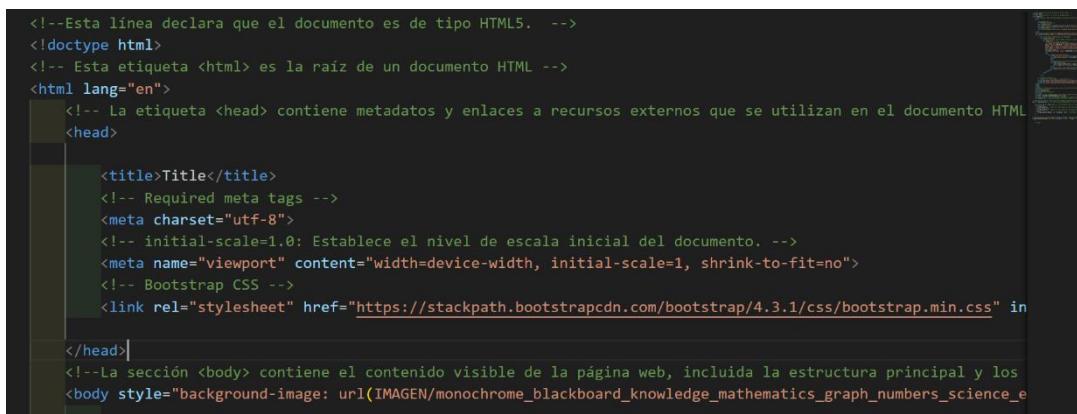
```
# style.css > ...
1
2 /* Estilos para el cuerpo de la página */
3 body {
4     font-family: sans-serif; /* Utiliza una fuente sans-serif para el texto */
5     text-align: center; /* Centra el contenido del cuerpo */
6 }
7
8 /* Estilos para el párrafo de resultado */
9 #result {
10    font-size: 24px; /* Tamaño de fuente grande */
11    font-weight: bold; /* Texto en negrita */
12    color: #00698f; /* Color del texto en azul oscuro */
13 }
14
15 /* Estilos para el encabezado principal */
16 h1 {
17     font-family: "Jersey 10 Charted", sans-serif; /* Utiliza una fuente personalizada con respaldo sans-serif */
18 }
19
20 |
```

Ilustración 24 ESTILOS DE CSS PARA LA PAGINA WEB

PRACTICA 7: SERIE FIBONACCI

Practica 7 consiste en la aplicación formar una serie Fibonacci para comenzar va a ocupar el lenguaje de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, vídeos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.



The screenshot shows a code editor displaying the following HTML code:

```
<!--Esta línea declara que el documento es de tipo HTML5. -->
<!doctype html>
<!-- Esta etiqueta <html> es la raíz de un documento HTML -->
<html lang="en">
    <!-- La etiqueta <head> contiene metadatos y enlaces a recursos externos que se utilizan en el documento HTML -->
    <head>
        <title>Title</title>
        <!-- Required meta tags -->
        <meta charset="utf-8">
        <!-- initial-scale=1.0: Establece el nivel de escala inicial del documento. -->
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1143x" crossorigin="anonymous">
    </head>
    <!--La sección <body> contiene el contenido visible de la página web, incluida la estructura principal y los enlaces internos-->
    <body style="background-image: url(IMG/monochrome_blackboard_knowledge_mathematics_graph_numbers_science_etc.jpg)">
        <h1>Serie Fibonacci</h1>
        <p>La Serie Fibonacci es una secuencia de números en la que cada número es la suma de los dos anteriores, comenzando con 0 y 1.</p>
        <ul>
            <li>0</li>
            <li>1</li>
            <li>1</li>
            <li>2</li>
            <li>3</li>
            <li>5</li>
            <li>8</li>
            <li>13</li>
            <li>21</li>
            <li>34</li>
            <li>55</li>
            <li>89</li>
            <li>144</li>
            <li>233</li>
            <li>377</li>
            <li>610</li>
            <li>987</li>
            <li>1597</li>
            <li>2584</li>
            <li>4181</li>
            <li>6771</li>
            <li>10946</li>
            <li>17923</li>
            <li>28854</li>
            <li>46777</li>
            <li>75551</li>
            <li>122323</li>
            <li>197674</li>
            <li>319997</li>
            <li>519997</li>
            <li>839994</li>
            <li>1359991</li>
            <li>2219985</li>
            <li>3539976</li>
            <li>5759951</li>
            <li>9299902</li>
            <li>15059853</li>
            <li>24119705</li>
            <li>39239458</li>
            <li>63359263</li>
            <li>102598521</li>
            <li>165957784</li>
            <li>268556265</li>
            <li>433512549</li>
            <li>701568854</li>
            <li>1133187703</li>
            <li>1834775557</li>
            <li>3168551260</li>
            <li>5003326817</li>
            <li>8166653677</li>
            <li>13166653677</li>
            <li>21333307354</li>
            <li>34499907354</li>
            <li>55933214708</li>
            <li>89433214708</li>
            <li>144366429416</li>
            <li>233799629416</li>
            <li>377599258832</li>
            <li>610368887648</li>
            <li>987737775280</li>
            <li>1594737775280</li>
            <li>2582475550560</li>
            <li>4172213101120</li>
            <li>6754688602240</li>
            <li>10926871704640</li>
            <li>17881553307200</li>
            <li>28808424611840</li>
            <li>46689977923680</li>
            <li>75378402545520</li>
            <li>122067375091040</li>
            <li>197445777632480</li>
            <li>31941315272480</li>
            <li>53882630544960</li>
            <li>85764255827920</li>
            <li>139646886655200</li>
            <li>225393772255200</li>
            <li>360830658910400</li>
            <li>586224431820800</li>
            <li>947054880731200</li>
            <li>1533279262552000</li>
            <li>2470333145073600</li>
            <li>4003606345073600</li>
            <li>6473942490147200</li>
            <li>10477548835147200</li>
            <li>17005497665147200</li>
            <li>27480995325147200</li>
            <li>44486492650294400</li>
            <li>71967088275294400</li>
            <li>116453580925294400</li>
            <li>188417169150588800</li>
            <li>304874338300588800</li>
            <li>513248676600588800</li>
            <li>820123015200588800</li>
            <li>1333371691200588800</li>
            <li>2253543382400588800</li>
            <li>3586915074400588800</li>
            <li>5840458456800588800</li>
            <li>9427373513600588800</li>
            <li>15267831970400588800</li>
            <li>24695663940800588800</li>
            <li>40001505915200588800</li>
            <li>64697169825600588800</li>
            <li>104698678681600588800</li>
            <li>170000000000000588800</li>
        </ul>
    </body>
</html>
```

Ilustración 25 ESTRUCTURA BASICA EN HTML

En seguida agregaremos un container Define un contenedor adaptable para el contenido, con su respectivo row que su funcionalidad es Define una fila para el diseño, posteriormente seguimos con la selección de columnas con estilos específicos, Definimos un encabezado de nivel 1 centrado, agreamos líneas de salto para crear espacio adicional , seguimos a definir un panel de bootstrap con estilo primario, también agreamos encabezado del panel con texto en negrita. Se agregaron las Definiciones el cuerpo del panel donde se encuentra la entrada del usuario, un cuadro de texto de entrada de número con un placeholder. un botón de éxito de Bootstrap que ejecuta la función JavaScript Fibonacci() cuando se hace clic.

```

<div class="container">
    <!--Define una fila para el diseño (aunque no se usa mucho en este caso).-->
    <div class="row">
        <!--Define un elemento de columna.-->
        <div class="col" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color de fondo con transparencia */; border: 5px solid #c0e9e8; /* Define el borde del cuadro de texto */; padding: 10px; /* Define el espaciado dentro del cuadro de texto */; width: 26px; /* Define el ancho del cuadro de texto */; color: #333; /* Define el color del texto dentro del cuadro de texto */; margin-top: 100px;">
            <h1 style="text-align: center;">BIENVENIDOS A LA SERIE FIBONACI </h1>
            <br>
            <br>
            <div class="panel panel-primary">
                <!--Aplica estilos a la sección del encabezado del panel.-->
                <div class="panel-heading"><strong>SERIE FIBONACI</strong></div>
                <br>
                <!--Aplica estilos a la sección del cuerpo del panel donde se encuentra la entrada del usuario.-->
                <div class="panel-body">
                    <input id="num" name="numero" class="form-control" type="number" placeholder="ingresa un numero">
                    <button class="btn btn-success" onclick="fibonacci()"> Generar serie fibonacci</button>
                </div>
                <!--Aplica estilos a la sección del pie de página del panel donde se muestra el resultado.-->
                <div class="panel-footer" id="resultado" style="word-wrap: break-word;"></div>
            </div>
        </div>
    </div>
</div>

```

Ilustración 26 CONTENIDO DE HTML

Posteriormente continuamos con el código JavaScript define una función fibonacci() que calcula la secuencia de Fibonacci hasta un número ingresado por el usuario. Este código JavaScript proporciona una función para calcular la secuencia de Fibonacci basada en la entrada del usuario y muestra el resultado dentro de un elemento HTML específico lo primero que agregamos fueron funciones

Función fibonacci ():

- Se declara una función llamada fibonacci () .

Declaración de variables:

- Dentro de la función, se declaran dos variables:
- x: Se inicializa con el valor 0.
- valor: Se utiliza para almacenar el número ingresado por el usuario. Se obtiene el valor del elemento HTML con ID "num" usando document.getElementById('num').value.

Creación del arreglo:

- Se declara un arreglo vacío llamado arreglo utilizando var arreglo = []; . Este arreglo almacenará la secuencia de Fibonacci calculada.

Comentario innecesario:

- La línea valor => parseInt(valor); está comentada. Esta línea convierte un valor de cadena a un entero, pero no se utiliza en la lógica actual del código. Se recomienda eliminarla para mejorar la claridad.

Ciclo for:

- Se utiliza un ciclo for para iterar desde 0 hasta valor-1 (exclusivo). En cada iteración:
 - Si x es igual a 0, se agrega 0 al arreglo (arreglo.push(0)) como el primer elemento de la secuencia.
 - Si x es igual a 1, se agrega 1 al arreglo (arreglo.push(1)) como el segundo elemento de la secuencia.
 - En caso contrario, se calcula el siguiente número de Fibonacci sumando los dos elementos anteriores del arreglo (arreglo [x - 1] y arreglo [x - 2]) y se agrega el resultado al arreglo (arreglo.push (arreglo[x - 1] + arreglo[x - 2])).

Mostrando el resultado:

- Se asume que existe un elemento HTML con ID "resultado" (verifique su código HTML).
- Se utiliza document.getElementById("resultado").innerHTML para modificar el contenido interno (innerHTML) del elemento con ID "resultado".
- Se construye una cadena con la etiqueta <h3> y el contenido del arreglo (arreglo) convertido a cadena.
- Finalmente, la cadena se muestra como un encabezado <h3> dentro del elemento con ID "resultado".

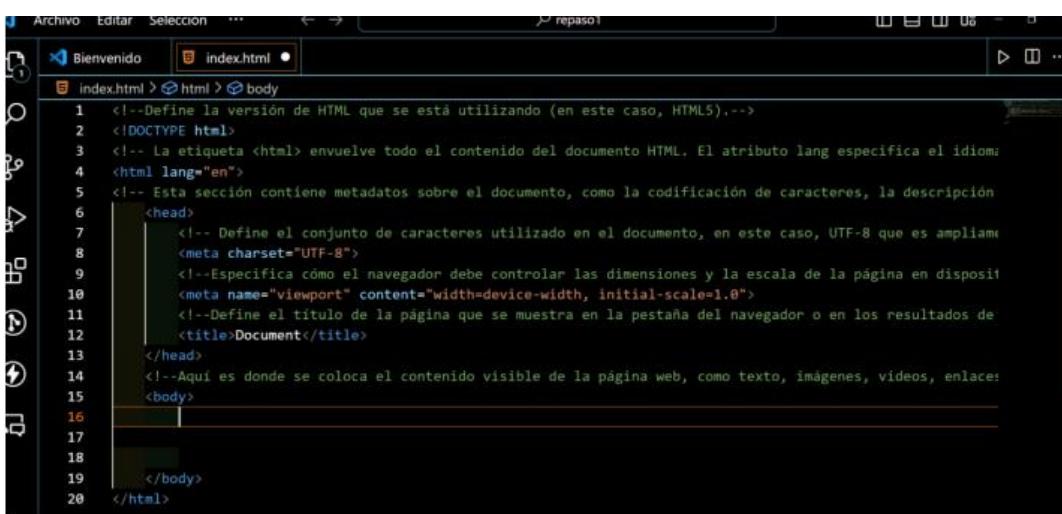
```
59 <script>
60   /*Contiene el código JavaScript que calcula la secuencia de Fibonacci*/
61   /*Se define la función*/
62   function fibonacci(){
63     /*Se declaran dos variables */
64     var x = 0;
65     /*recupera el valor ingresado por el usuario en el campo de entrada usando*/
66     var valor = document.getElementById('num').value;
67     /*Se crea una matriz vacía arreglo para almacenar los números de Fibonacci.*/
68     var arreglo = []; // Declares an empty array named "arreglo" to store Fibonacci sequence values.
69
70   // Function to convert a string value to an integer (unused in this code)
71   valor => parseInt(valor); // This line could be removed as it's not used in the current logic.
72
73   for (let x = 0; x < valor; x++) { // Loop iterates from 0 to "valor" (exclusive)
74     if (x === 0) {
75       arreglo.push(0); // Pushes 0 to the arreglo for the first element (base case).
76     } else if (x === 1) {
77       arreglo.push(1); // Pushes 1 to the arreglo for the second element (base case).
78     } else {
79       arreglo.push(arreglo[x - 1] + arreglo[x - 2]); // Calculates the next Fibonacci number using previous two elements.
80     }
81   }
82
83   // Assuming there's an HTML element with ID "resultado" (check your HTML)
84   document.getElementById("resultado").innerHTML = "<h3>" + arreglo + "</h3>"; // Displays the arreglo as an h3 heading within the ele
85   }
86
87 </script>
```

Ilustración 27 CODIGO DE JAVASCRIPT

PRÁCTICA 8: LEYES DE OHM, CULOM Y NEWTON

Práctica 8 consiste en la aplicación del tener páginas enlazadas y cada una tengan las leyes de ohm, culón y newton para comenzar va a ocupar el lenguaje de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, vídeos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.



The screenshot shows a code editor window with the following content:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
    <meta charset="UTF-8">
    <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.-->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.-->
    <title>Document</title>
  </head>
  <!--Aqui es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.-->
  <body>
  </body>
</html>
```

Ilustración 28 ESTURUCTURA BASICA DE HTML

Posteriormente continuamos con el código HTML este documento será el principal para manejar a los otros index, continuando con la estructura de nuestra pagina principal vamos a representar la estructura básica de una página web que muestra un título centrado "LEYES DE FISICA" y un botón desplegable con la etiqueta "calcula". El código utiliza Bootstrap 4 para el diseño y la interactividad del botón desplegable. Continuando con el contenedor principal vamos a definir cada paso que hice en el código para llegar a los resultados esperados en esta práctica.

Contenedor Principal:

- La sección principal está definida por la etiqueta <div> con la clase container. Este contenedor define el ancho máximo de la página y centra el contenido horizontalmente en pantallas más grandes.

Fila Principal:

- Dentro del contenedor principal, se define una fila con la etiqueta <div> y la clase row. Esta fila permite agrupar elementos horizontalmente.

Columnas Vacías:

- La fila principal contiene dos columnas vacías con las clases col-sm-3 y col-sm-2. Estas columnas no tienen contenido y se utilizan para espaciar el contenido principal hacia el centro de la página en dispositivos de tamaño mediano (pantallas de tabletas).

Columna Principal:

- La fila principal también contiene una columna principal con la clase col-sm-10. Esta columna contiene el contenido principal de la página, que en este caso es un subcontenedor.

Subfila:

- Dentro de la columna principal, se define otra fila con la etiqueta <div> y la clase row. Esta subfila permite agrupar elementos horizontalmente dentro de la columna principal.

Columna Vacía (Subfila):

- La subfila contiene otra columna vacía con la clase col-sm-3. Esta columna se utiliza para espaciar horizontalmente el contenido dentro de la subfila.

Columna Principal (Leyes de Física):

- La subfila contiene una columna principal con la clase col-sm-6. Esta columna contiene el título "LEYES DE FISICA" y el botón desplegable.

Estilos en Línea:

- La columna principal tiene estilos en línea aplicados mediante el atributo Style. Estos estilos definen el alto, el color de fondo, el borde, el relleno, el ancho, el color del texto y el margen superior del contenedor.

Saltos de Línea:

- Varios saltos de línea (
) se utilizan dentro de la columna principal para crear espacio vertical antes del título.

Título:

- Un encabezado <h1> con estilos en línea define el título "LEYES DE FISICA". El estilo en línea define el color del texto y la alineación central.

Contenedor del Botón Desplegable:

- Un contenedor con la clase drop (posiblemente definido en una hoja de estilos externa) se utiliza para agrupar el botón desplegable.

Salto de Línea:

- Otro salto de línea se agrega dentro del contenedor del botón desplegable.

Grupo de Botones:

- Un grupo de botones se define con la etiqueta <div> y la clase btn-group. Los estilos en línea definen el ancho, alto, y la justificación del contenido dentro del grupo (centrado en este caso).

Salto de Línea:

- Se agrega otro salto de línea dentro del grupo de botones.

Botón Principal:

- Un botón principal se define con la etiqueta <button>. El botón tiene las clases btn, btn-secondary, y dropdown-toggle. Los estilos en línea definen el color de fondo, los márgenes y la alineación del botón. El atributo data-toggle="dropdown" habilita la funcionalidad del menú desplegable.

Etiqueta del Botón:

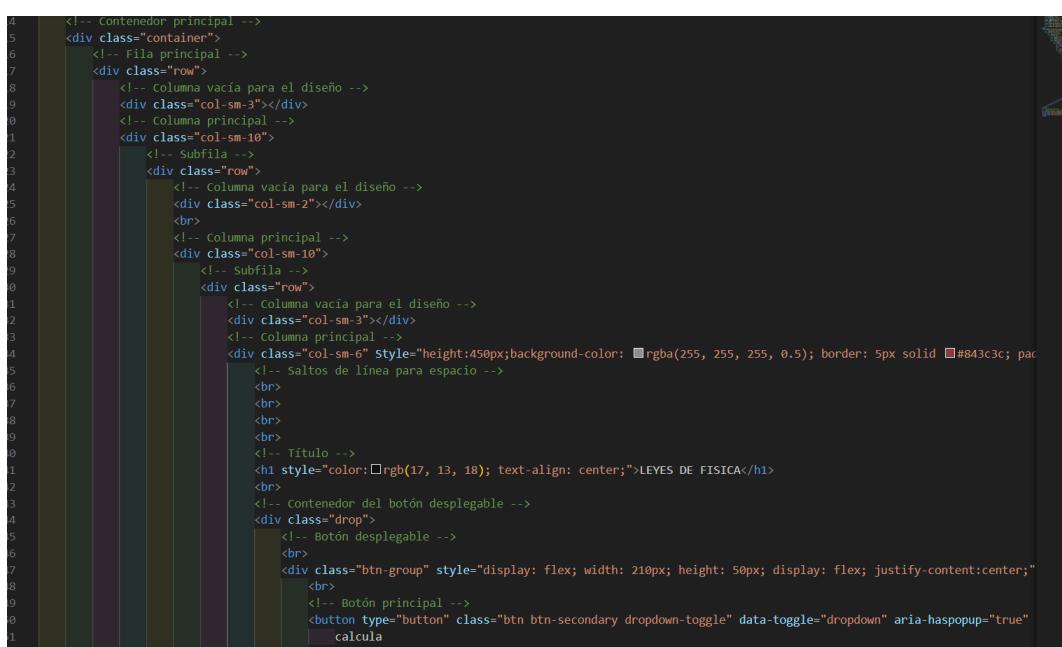
- El contenido del botón es el texto "calcula".

Menú Desplegable:

- Un menú desplegable se define con la etiqueta <div> y las clases dropdown-menu y dropdown-menu-right. La clase dropdown-menu-right indica que el menú se desplegará hacia la derecha del botón.

Enlaces del Menú Desplegable (Faltan):

- El código actual no muestra ningún enlace dentro del menú desplegable. Se espera que este menú contenga enlaces a diferentes páginas o secciones relacionadas con las leyes de la física.



```
1  <!-- Contenedor principal -->
2  <div class="container">
3      <!-- Fila principal -->
4      <div class="row">
5          <!-- Columna vacía para el diseño -->
6          <div class="col-sm-3"></div>
7          <!-- Columna principal -->
8          <div class="col-sm-10">
9              <!-- Subfila -->
10             <div class="row">
11                 <!-- Columna vacía para el diseño -->
12                 <div class="col-sm-2"></div>
13                 <br>
14                 <!-- Columna principal -->
15                 <div class="col-sm-10">
16                     <!-- Subfila -->
17                     <div class="row">
18                         <!-- Columna vacía para el diseño -->
19                         <div class="col-sm-3"></div>
20                         <!-- Columna principal -->
21                         <div class="col-sm-6" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); border: 5px solid #843c3c; padding: 10px; margin: 10px; font-size: 1.2em; font-weight: bold; text-align: center; border-radius: 10px; transition: all 0.5s ease-in-out; position: relative; z-index: 1; ">
22                             <br>
23                             <br>
24                             <br>
25                             <br>
26                             <!-- Título -->
27                             <h1 style="color:#rgb(17, 13, 18); text-align: center;">LEYES DE FISICA</h1>
28                             <br>
29                         <!-- Contenedor del botón desplegable -->
30                         <div class="drop">
31                             <!-- Botón desplegable -->
32                             <br>
33                             <div class="btn-group" style="display: flex; width: 210px; height: 50px; display: flex; justify-content:center; align-items:center; border-radius: 10px; background-color: #843c3c; color: white; font-size: 1.2em; font-weight: bold; text-align: center; border: none; padding: 0; margin: 0; ">
34                                 <br>
35                                 <!-- Botón principal -->
36                                 <button type="button" class="btn btn-secondary dropdown-toggle" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" style="width: 100%; height: 100%; border: none; background-color: inherit; color: inherit; font-size: inherit; font-weight: inherit; text-align: inherit; ">
37                                     calcula
38                                 </button>
39                             </div>
40                         </div>
41                     </div>
42                 </div>
43             </div>
44         </div>
45     </div>
46 
```

Ilustración 29 CONTEINIDO DE HTML COMENZANDO CON UNCONTENEDOR

Posteriormente vamos a crear nuevos índex con los siguientes nombres para identificarlos ohm, culón y newton, en seguida vamos a crear una carpeta de imágenes esta será útil para el fondo de las páginas, ya que las cree la primera que estructura fue la página de ohm que será la primera la vamos a enlazar con nuestra página principal, comenzaremos con la estructura básica de HTML que como hemos definido HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

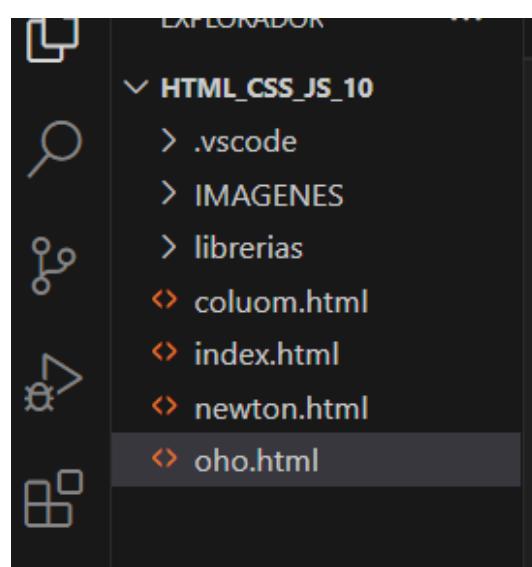


Ilustración 30 INDEX A CREAR DE COLOM, NEWTON Y OHON

```

1 <!--Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
2 <!DOCTYPE html>
3 <!-- La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma
4 <html lang="en">
5 <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción
6 <head>
7     <!-- Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamen
8     <meta charset="UTF-8">
9     <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositi
10    <meta name="viewport" content="width=device-width, initial-scale=1.0">
11    <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de
12    <title>Document</title>
13 </head>
14 <!--Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces
15 <body>
16
17
18
19 </body>
20 </html>

```

Ilustración 31 ESTRUCTURA BASICA DE HTML

El código HTML suministrado crea una página web que permite a los usuarios calcular el voltaje (V) utilizando la Ley de Ohm ($V = I * R$) ingresando valores de corriente (I) y resistencia (R). La página utiliza Bootstrap 4 para el estilo y JavaScript para la interactividad.

Enlace a CSS de Bootstrap:

```

<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"

```

- Este enlace importa el archivo CSS de Bootstrap desde un CDN (Content Delivery Network), proporcionando los estilos necesarios para el diseño, los componentes y la apariencia de la página web. Bootstrap es un framework CSS popular que simplifica el desarrollo web al ofrecer estilos y componentes prediseñados.

Sección Body:

- La sección `<body>` contiene el contenido visible de la página web, incluidos la estructura principal, los elementos y los componentes interactivos.
- El atributo `style` aplica estilos CSS en línea a todo el cuerpo de la página, definiendo la imagen de fondo, la familia de fuentes, el tamaño de fuente, el tamaño del fondo, la fijación del fondo y la posición del fondo.

Contenedor:

- El elemento `<div>` con la clase `container` proporciona un contenedor adaptable para el contenido de la página, asegurando que los elementos estén organizados y alineados adecuadamente en diferentes tamaños de pantalla. La clase `.container` de Bootstrap ayuda a crear un diseño fluido que se adapta a varios dispositivos.

`<div class="row">`

- El elemento `<div>` con la clase `row` representa una fila horizontal dentro del contenedor. Sirve como contenedor para las columnas que albergarán el contenido de la página.

Columna Vacía:

```
<div class="col-sm-4"></div>
```

- El elemento `<div>` con la clase `col-sm-4` representa una columna vacía que ocupa 4 de las 12 columnas de la fila. Esto crea un elemento espaciador en el lado izquierdo de la columna de contenido.
- Este elemento `<h1>` crea un encabezado grande con el texto "OHM" mostrado en el centro de la columna de contenido. El atributo `style` aplica estilos CSS en línea para establecer el color del texto a RGB(23, 20, 20).

Etiqueta de entrada de corriente:

- Este elemento `<label>` crea una etiqueta para el campo de entrada de corriente. El atributo `for` especifica el ID del campo de entrada al que está asociada la etiqueta (`id="numero1"`). El texto "Ingresa el valor de corriente:" indica al usuario que ingrese el valor de la corriente.

Campo de entrada de corriente:

- Este elemento `<input>` crea un campo de entrada de número para que el usuario ingrese el valor de la corriente. El atributo `type` especifica el tipo de entrada como "number", asegurando que solo se puedan ingresar valores numéricos. El atributo `class` aplica la clase Bootstrap `form-control` para el estilo. El atributo `name` asigna el nombre "numero1" al campo de entrada. El atributo `id` establece el ID del campo de entrada en "numero1", coincidiendo con el atributo `for` de la etiqueta. El atributo `placeholder` proporciona una sugerencia al usuario ("solo valores numéricos"). El atributo `required` hace que el campo de entrada sea obligatorio.

Salto de línea:

- Este elemento `
` inserta un salto de línea después del campo de entrada de corriente, creando un espacio vertical entre el campo de entrada y el siguiente elemento.

Etiqueta de entrada de resistencia:

- Este elemento `<label>` crea una etiqueta para el campo de entrada de resistencia. El atributo `for` especifica el ID del campo de entrada al que está asociada la etiqueta (`id="numero2"`). El texto "Ingresa el valor de resistencia:" indica al usuario que ingrese el valor de la resistencia.

Campo de entrada de resistencia:

- Este elemento `<input>` crea un campo de entrada de número para que el usuario ingrese el valor de la resistencia. El atributo `type` especifica el tipo de entrada como "number", asegurando que solo se puedan ingresar valores numéricos. El atributo `class` aplica la clase Bootstrap `form-control` para el estilo. El atributo `name` asigna el nombre "numero2" al campo de entrada. El atributo `id` establece el ID del campo de entrada en "numero2", coincidiendo con el atributo `for` de la etiqueta. El atributo `placeholder` proporciona una sugerencia al usuario ("solo valores numéricos"). El atributo `required` hace que el campo de entrada sea obligatorio.

Salto de línea:

- Este elemento `
` inserta un salto de línea después del campo de entrada de resistencia, creando un espacio vertical entre el campo de entrada y el siguiente elemento.

Etiqueta de resultado:

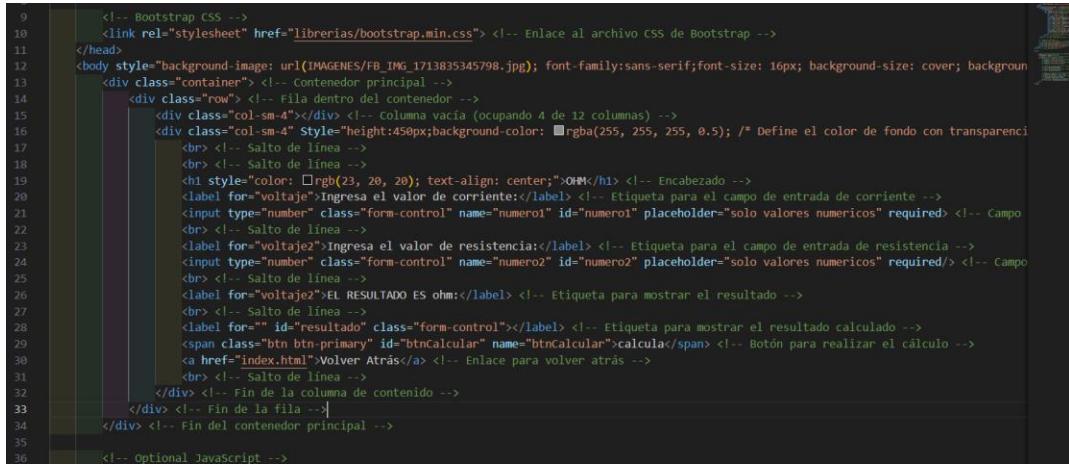
- Este elemento `<label>` crea una etiqueta para el área de visualización del resultado. El atributo `for` especifica el ID del elemento al que está asociada la etiqueta (`id="resultado"`). El texto "EL RESULTADO ES ohm:" informa al usuario que el voltaje calculado se mostrará en ohmios (Ω).

Salto de línea:

- Este elemento
 inserta un salto de línea después de la etiqueta de resultado, creando un espacio vertical entre la etiqueta y el siguiente elemento.

Área de visualización del resultado:

- Este elemento <label> crea una etiqueta vacía con el ID "resultado" para mostrar el valor del voltaje calculado.



```
9      <!-- Bootstrap CSS -->
10     <link rel="stylesheet" href="librerias/bootstrap.min.css"> <!-- Enlace al archivo CSS de Bootstrap -->
11   </head>
12   <body style="background-image: url(IMAGENES/FB_IMG_1713835345798.jpg); font-family:sans-serif;font-size: 16px; background-size: cover; background-color: #f2f2f2; margin: 0; padding: 0;">
13     <div class="container"> <!-- Contenedor principal -->
14       <div class="row"> <!-- Fila dentro del contenedor -->
15         <div class="col-sm-4"></div> <!-- columna vacía (ocupando 4 de 12 columnas) -->
16         <div class="col-sm-4" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color de fondo con transparencia -->
17           <br> <!-- Salto de línea -->
18           <br> <!-- Salto de línea -->
19           <h1 style="color: #rgb(23, 28, 28); text-align: center;">OHM</h1> <!-- Encabezado -->
20           <label for="voltaje">Ingresa el valor de corriente:</label> <!-- Etiqueta para el campo de entrada de corriente -->
21           <input type="number" class="form-control" name="numero1" id="numero1" placeholder="solo valores numéricos" required> <!-- Campo de entrada de corriente -->
22           <br> <!-- Salto de línea -->
23           <label for="voltaje2">Ingresa el valor de resistencia:</label> <!-- Etiqueta para el campo de entrada de resistencia -->
24           <input type="number" class="form-control" name="numero2" id="numero2" placeholder="solo valores numéricos" required> <!-- Campo de entrada de resistencia -->
25           <br> <!-- Salto de línea -->
26           <label for="voltaje2">EL RESULTADO ES ohm:</label> <!-- Etiqueta para mostrar el resultado -->
27           <br> <!-- Salto de línea -->
28           <label for="" id="resultado" class="form-control"></label> <!-- Etiqueta para mostrar el resultado calculado -->
29           <span class="btn btn-primary" id="btnCalcular" name="btnCalcular">calcular</span> <!-- Botón para realizar el cálculo -->
30           <a href="index.html">Volver Atrás</a> <!-- Enlace para volver atrás -->
31           <br> <!-- Salto de línea -->
32         </div> <!-- Fin de la columna de contenido -->
33       </div> <!-- Fin de la fila -->
34     </div> <!-- Fin del contenedor principal -->
35   <!-- Optional JavaScript -->
```

Ilustración 32 CONTENIDO DE HTML COMENZANDO CON UN CONTEDOR

Posteriormente vamos a agregar el código JavaScript mejora la funcionalidad de la página HTML al agregar un detector de eventos al botón "Calcular". Cuando se hace clic en el botón, recupera los valores ingresados en los campos de entrada de corriente y resistencia, realiza el cálculo de la Ley de Ohm (voltaje = corriente / resistencia) y muestra el valor de voltaje calculado en el área de resultados. En general, el código JavaScript agrega efectivamente interactividad y funcionalidad a la página HTML, permitiendo a los usuarios calcular el voltaje usando la Ley de Ohm y mostrando los resultados de una manera fácil de usar comenzaremos con la descripción del código.

Función Documento Listo:

- Este código utiliza la función \$(document).ready() de jQuery para garantizar que el código JavaScript se ejecute solo después de que se haya cargado y analizado todo el documento HTML. Esto evita posibles errores causados por el acceso a elementos DOM antes de que estén completamente disponibles.

Manejador de eventos de clic en el botón:

- Este código adjunta un detector de eventos al elemento de botón con el ID "btnCalcular". La función del manejador de eventos click se activa cada vez que el usuario hace clic en el botón.

Obteniendo valores de entrada:

- Dentro de la función del manejador de eventos, se utiliza el método val() para extraer los valores ingresados en los campos de entrada con ID "numero1" y "numero2". Estos valores se almacenan en las variables numero1 y numero2, respectivamente.

Analizando valores de entrada a enteros:

- La función parseInt() se utiliza para convertir los valores de cadena recuperados de los campos de entrada (numero1 y numero2) en números enteros. Esto asegura que los valores se traten como datos numéricos para el cálculo.

Realizando el cálculo de la Ley de Ohm:

- El cálculo de la Ley de Ohm se realiza dividiendo el valor de la corriente (almacenado en uno) por el valor de la resistencia (almacenado en dos). El resultado se almacena en la variable operacion_newtons.

Mostrando el voltaje calculado:

- El valor de voltaje calculado (almacenado en operacion_newtons) se inserta en el elemento con el ID "resultado" usando el método text(). Esto muestra efectivamente el resultado al usuario.

```
44 <script>
45
46 // Espera a que el documento HTML esté completamente cargado antes de ejecutar el código jQuery
47 $(document).ready(function(){
48     // Agrega un evento de clic al botón con el ID 'btnCalcular'
49     $('#btnCalcular').click(function(){
50         // Obtiene el valor del campo de entrada con el ID 'numero1'
51         let numero1 = $('#numero1').val();
52         // Obtiene el valor del campo de entrada con el ID 'numero2'
53         let numero2 = $('#numero2').val();
54
55         // Convierte el valor obtenido de 'numero1' a un número entero
56         let uno = parseInt(numero1);
57         // Convierte el valor obtenido de 'numero2' a un número entero
58         let dos = parseInt(numero2);
59
60         // Realiza la operación de división entre los dos números obtenidos
61         let operacion_newtons = (uno / dos);
62         // Define la variable resultado para almacenar el resultado de la operación
63         let resultado=operacion_newtons;
64         // Inserta el resultado de la operación en el elemento con el ID 'resultado'
65         $('#resultado').text(operacion_newtons);
66     });
67
68 </script>
```

Ilustración 33 CODIGO DE JAVASCRIPT

Ahora vamos a ir a estructurar el siguiente index que será el newton, vamos a abrir el index comenzaremos con la estructura básica de HTML que como hemos definido HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web esta página también será enlazada con la página principal.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.

- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

```

1  <!--Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
2  <!DOCTYPE html>
3  <!-- La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma
4  <html lang="en">
5  <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción
6  <head>
7      <!-- Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamen
8      <meta charset="UTF-8">
9      <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositi
10     <meta name="viewport" content="width=device-width, initial-scale=1.0">
11     <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de
12     <title>Document</title>
13 </head>
14 <!--Aqui es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces:
15 <body>
16
17
18
19
20 </body>
</html>

```

Ilustración 34 ESTRUCTURA BÁSICA DE HTML

Una vez que ya está la estructura vamos a comenzar con el código complementario de la página de Newton vamos a agregar nuevamente este enlace importa el archivo CSS de Bootstrap desde un CDN (Content Delivery Network), proporcionando los estilos necesarios para el diseño, los componentes y la apariencia de la página web. Bootstrap es un framework CSS popular que simplifica el desarrollo web al ofrecer estilos y componentes prediseñados.

Sección Body:

La sección <body> contiene el contenido visible de la página web, incluidos la estructura principal, los elementos y los componentes interactivos.

El atributo style aplica estilos CSS en línea a todo el cuerpo de la página, definiendo la imagen de fondo, la familia de fuentes, el tamaño de fuente, el tamaño del fondo, la fijación del fondo y la posición del fondo.

Contenedor:

El elemento <div> con la clase container proporciona un contenedor adaptable para el contenido de la página, asegurando que los elementos estén organizados y alineados adecuadamente en diferentes tamaños de pantalla. La clase .container de Bootstrap ayuda a crear un diseño fluido que se adapta a varios dispositivos.

Fila:

<div class="row">

El elemento <div> con la clase row representa una fila horizontal dentro del contenedor. Sirve como contenedor para las columnas que albergarán el contenido de la página.

Columna Vacía:

<div class="col-sm-4"></div>

El elemento <div> con la clase col-sm-4 representa una columna vacía que ocupa 4 de las 12 columnas de la fila. Esto crea un elemento espaciador en el lado izquierdo de la columna de contenido.

Este elemento `<h1>` crea un encabezado grande con el texto "newton" mostrado en el centro de la columna de contenido. El atributo `style` aplica estilos CSS en línea para establecer el color del texto a RGB(23, 20, 20).

11. Etiqueta de entrada de aceleración:

- Esta etiqueta `<label>` crea un texto que indica al usuario que ingrese el valor de la aceleración, incluyendo la unidad de medida (metros por segundo al cuadrado, m/s^2). El atributo `for` especifica el ID del campo de entrada al que está asociada la etiqueta (`id="numero1"`).

12. Campo de entrada de número (Aceleración):

- Este elemento `<input>` crea un campo de entrada de número para que el usuario ingrese el valor de la aceleración. El atributo `type` especifica el tipo de entrada como "number", asegurando que solo se puedan ingresar valores numéricos. La clase `form-control` de Bootstrap aplica estilos predefinidos al campo de entrada. El atributo `name` asigna el nombre "numero1" al campo de entrada. El atributo `id` establece el ID del campo de entrada en "numero1", lo que permite a la etiqueta `for` vincularlo. El atributo `placeholder` proporciona una sugerencia al usuario ("Solo valores numéricos"). El atributo `required` hace que el campo de entrada sea obligatorio.

13. Salto de línea:

- Este elemento `
` inserta un salto de línea después del campo de entrada de aceleración, creando un espacio vertical entre el campo de entrada y el siguiente elemento.

14. Etiqueta de entrada de masa:

- Esta etiqueta `<label>` crea un texto que indica al usuario que ingrese el valor de la masa, incluyendo la unidad de medida (kilogramos, kg). El atributo `for` especifica el ID del campo de entrada al que está asociada la etiqueta (`id="numero2"`).

15. Campo de entrada de número (Masa):

- Este elemento `<input>` crea un campo de entrada de número para que el usuario ingrese el valor de la masa. El atributo `type` especifica el tipo de entrada como "number", asegurando que solo se puedan ingresar valores numéricos. La clase `form-control` de Bootstrap aplica estilos predefinidos al campo de entrada. El atributo `name` asigna el nombre "numero2" al campo de entrada. El atributo `id` establece el ID del campo de entrada en "numero2", lo que permite a la etiqueta `for` vincularlo. El atributo `placeholder` proporciona una sugerencia al usuario ("Solo valores numéricos"). El atributo `required` hace que el campo de entrada sea obligatorio.

16. Salto de línea:

- Este elemento `
` inserta un salto de línea después del campo de entrada de masa, creando un espacio vertical entre el campo de entrada y el siguiente elemento.

17. Etiqueta de resultado:

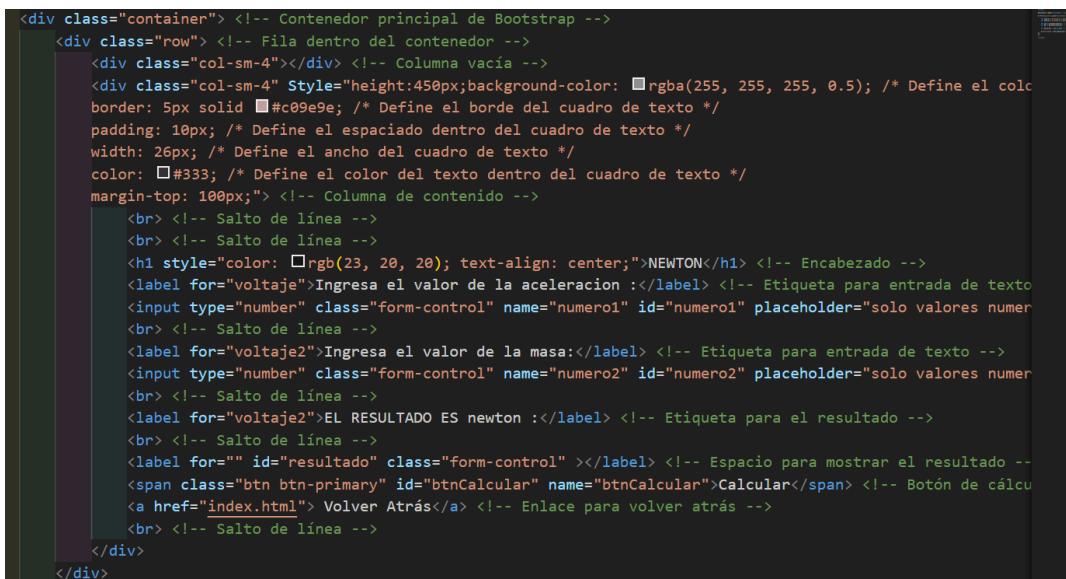
- Esta etiqueta `<label>` crea un texto que indica al usuario el significado del resultado que se mostrará a continuación (la fuerza en Newtons). El atributo `for` no se utiliza en este caso porque no está vinculado a un elemento de entrada específico.

18. Espacio para mostrar el resultado:

- Esta etiqueta <label> se utiliza como un espacio en blanco para mostrar el resultado calculado. El atributo `id` se establece en "resultado" para que el JavaScript pueda acceder y modificar su contenido. La clase `form-control` de Bootstrap aplica estilos predefinidos al campo de resultado.

19. Botón de cálculo:

- Este elemento crea un botón con el texto "Calcular". La clase `btn btn-primary` de Bootstrap aplica estilos para un botón primario. El atributo `id` se establece en "btnCalcular" para que el JavaScript pueda acceder y activar el cálculo.



```
<div class="container"> <!-- Contenedor principal de Bootstrap -->
<div class="row"> <!-- Fila dentro del contenedor -->
  <div class="col-sm-4"></div> <!-- Columna vacía -->
  <div class="col-sm-4" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color */ border: 5px solid #c9e9e9; /* Define el borde del cuadro de texto */ padding: 10px; /* Define el espaciado dentro del cuadro de texto */ width: 26px; /* Define el ancho del cuadro de texto */ color: #333; /* Define el color del texto dentro del cuadro de texto */ margin-top: 100px;"> <!-- Columna de contenido -->
    <br> <!-- Salto de línea -->
    <br> <!-- Salto de línea -->
    <h1 style="color: #rgb(23, 20, 20); text-align: center;">NEWTON</h1> <!-- Encabezado -->
    <label for="voltaje">Ingresa el valor de la aceleracion :</label> <!-- Etiqueta para entrada de texto -->
    <input type="number" class="form-control" name="numero1" id="numero1" placeholder="solo valores numericos" />
    <br> <!-- Salto de línea -->
    <label for="voltaje2">Ingresa el valor de la masa:</label> <!-- Etiqueta para entrada de texto -->
    <input type="number" class="form-control" name="numero2" id="numero2" placeholder="solo valores numericos" />
    <br> <!-- Salto de línea -->
    <label for="" id="resultado" class="form-control" style="height: 40px; width: 26px;">EL RESULTADO ES newton :</label> <!-- Etiqueta para el resultado -->
    <span class="btn btn-primary" id="btnCalcular" name="btnCalcular">Calcular</span> <!-- Botón de cálculo -->
    <a href="index.html"> Volver Atrás</a> <!-- Enlace para volver atrás -->
    <br> <!-- Salto de línea -->
  </div>
</div>
```

Ilustración 35 ESTRUCTURA DE LA PAGINA WUEB DE NEWTON

En seguida vamos a agregar el código JavaScript realza la funcionalidad de la página HTML añadiendo interactividad y cálculos. Utiliza jQuery para esperar a que el documento esté completamente cargado y luego adjunta un detector de eventos clic al botón con el ID "btnCalcular". Al hacer clic en el botón, recupera los valores ingresados en los campos de entrada de aceleración y masa, realiza la operación de multiplicación (fuerza = masa * aceleración) y muestra el resultado de la fuerza calculada en el área designada. el código JavaScript proporcionado agrega efectivamente interactividad y funcionalidad a la página HTML, permitiendo a los usuarios calcular la fuerza usando la Segunda Ley del Movimiento de Newton y mostrando los resultados de una manera fácil de usar.

1. Función Documento Listo:

- Este código utiliza la función `$(document).ready()` de jQuery para garantizar que el código JavaScript se ejecute solo después de que se haya cargado y analizado todo el documento HTML. Esto evita posibles errores causados por el acceso a elementos DOM antes de que estén completamente disponibles.

2. Manejador de eventos clic para el botón:

- Este código adjunta un detector de eventos clic al elemento con el ID "btnCalcular" usando `$('#btnCalcular').click()` de jQuery. La función `function()` dentro de las llaves define las acciones que se ejecutarán cuando se haga clic en el botón.

3. Obteniendo valores de entrada:

- Estas líneas de código utilizan los métodos `$('#numero1').val()` y `$('#numero2').val()` de jQuery para recuperar los valores ingresados en los campos de entrada con ID "numero1" (aceleración) y "numero2" (masa), respectivamente. Los valores recuperados se almacenan en las variables `numero1` y `numero2`.

4. Analizando valores de entrada a enteros:

- La función `parseInt()` se utiliza para convertir los valores de cadena recuperados de los campos de entrada (`numero1` y `numero2`) en números enteros. Esto asegura que los valores se traten como datos numéricos para el cálculo. Los valores convertidos se almacenan en las variables `uno` y `dos`.

5. Realizando el cálculo:

```
let operacion = (uno * dos);
```

- La operación de multiplicación se realiza multiplicando los valores almacenados en `uno` (aceleración) y `dos` (masa). El resultado del cálculo se almacena en la variable `operacion`.

6. Mostrando el resultado calculado:

- Esta línea de código utiliza el método `$('#resultado').text()` de jQuery para insertar el valor calculado almacenado en `operacion` en el elemento con el ID "resultado". Este elemento sirve como el área designada para mostrar el resultado de la fuerza calculada al usuario.

```
(document).ready(function(){ // Espera a que el documento HTML esté completamente cargado antes de ejecutar el código
    $('#btnCalcular').click(function(){
        let numero1 = $('#numero1').val(); // Obtiene el valor del campo de entrada con el ID 'numero1'
        let numero2 = $('#numero2').val(); // Obtiene el valor del campo de entrada con el ID 'numero2'

        let uno = parseInt(numero1); // Convierte el valor obtenido de 'numero1' a un número entero
        let dos = parseInt(numero2); // Convierte el valor obtenido de 'numero2' a un número entero

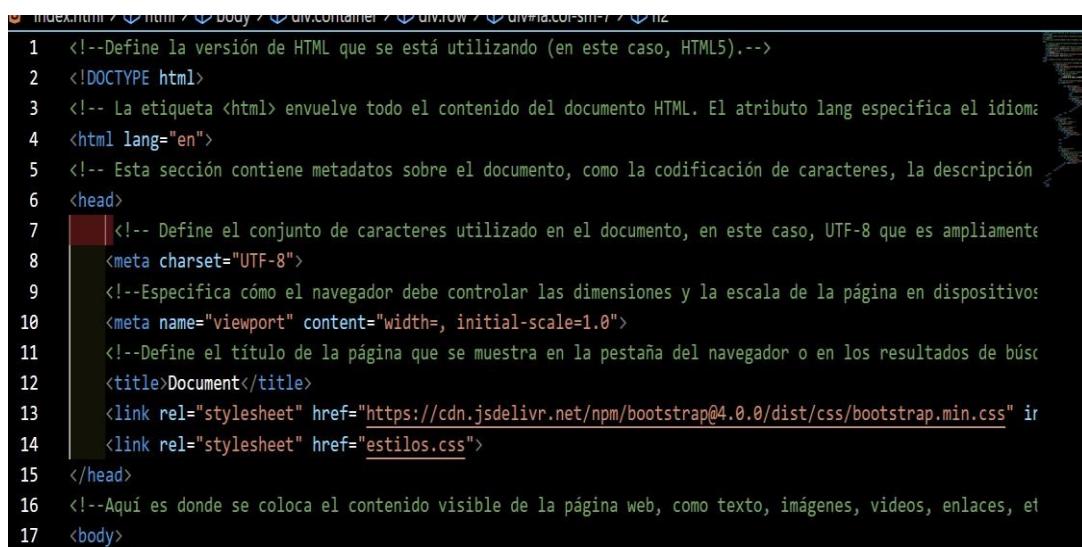
        let operacion = (uno * dos); // Realiza la operación de multiplicación entre los dos números obtenidos

        $('#resultado').text(operacion); // Inserta el resultado de la operación en el elemento con el ID 'resultado'
    });
});
```

Ilustración 36 CODIGO DE JAVASCRIPT

Posteriormente vamos a estructurar el último índice que es el de Colom para comenzar va a ocupar el lenguaje de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, videos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.



```

index.html
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css" integrity="sha384-Gn5384f9cEaQ4Z4+c5GwYBjEoO7M+qkxV1lqquF0mQZLJ4d8HrDzGZK2oqKXWg==">
8      <link rel="stylesheet" href="estilos.css">
9    </head>
10   <body>
11
12
13
14
15
16
17

```

Ilustración 37 ESTRUCTURA BASICA DE HTML

Una vez que ya está la estructura vamos a comenzar con el código complementario de la página de Newton vamos a agregar nuevamente este enlace importando el archivo CSS de Bootstrap desde un CDN (Content Delivery Network), proporcionando los estilos necesarios para el diseño, los componentes y la apariencia de la página web. Bootstrap es un framework CSS popular que simplifica el desarrollo web al ofrecer estilos y componentes prediseñados.

Sección Body:

- La sección <body> contiene el contenido visible de la página web, incluidos la estructura principal, los elementos y los componentes interactivos.
- El atributo style aplica estilos CSS en línea a todo el cuerpo de la página, definiendo la imagen de fondo, la familia de fuentes, el tamaño de fuente, el tamaño del fondo, la fijación del fondo y la posición del fondo.

Contenedor:

- El elemento <div> con la clase container proporciona un contenedor adaptable para el contenido de la página, asegurando que los elementos estén organizados y alineados adecuadamente en diferentes tamaños de pantalla. La clase .container de Bootstrap ayuda a crear un diseño fluido que se adapta a varios dispositivos.

```
<div class="row">
```

- El elemento <div> con la clase row representa una fila horizontal dentro del contenedor. Sirve como contenedor para las columnas que albergarán el contenido de la página.

Columna Vacía:

- El elemento <div> con la clase col-sm-4 representa una columna vacía que ocupa 4 de las 12 columnas de la fila. Esto crea un elemento espaciador en el lado izquierdo de la columna de contenido.
- Este elemento <h1> crea un encabezado grande con el texto "Newton" mostrado en el centro de la columna de contenido. El atributo style aplica estilos CSS en línea para establecer el color del texto a RGB(23, 20, 20).

11. Etiqueta de entrada de aceleración:

- Esta etiqueta <label> crea un texto que indica al usuario que ingrese el valor de la aceleración, incluyendo la unidad de medida (metros por segundo al cuadrado, m/s²). El atributo for especifica el ID del campo de entrada al que está asociada la etiqueta (id="numero1").

12. Campo de entrada de número (Aceleración):

- Este elemento <input> crea un campo de entrada de número para que el usuario ingrese el valor de la aceleración. El atributo type especifica el tipo de entrada como "number", asegurando que solo se puedan ingresar valores numéricos. La clase form-control de Bootstrap aplica estilos predefinidos al campo de entrada. El atributo name asigna el nombre "numero1" al campo de entrada. El atributo id establece el ID del campo de entrada en "numero1", lo que permite a la etiqueta for vincularlo. El atributo placeholder proporciona una sugerencia al usuario ("Solo valores numéricos"). El atributo required hace que el campo de entrada sea obligatorio.

13. Salto de línea:

- Este elemento
 inserta un salto de línea después del campo de entrada de aceleración, creando un espacio vertical entre el campo de entrada y el siguiente elemento.

14. Etiqueta de entrada de masa:

- Esta etiqueta <label> crea un texto que indica al usuario que ingrese el valor de la masa, incluyendo la unidad de medida (kilogramos, kg). El atributo for especifica el ID del campo de entrada al que está asociada la etiqueta (id="numero2").

15. Campo de entrada de número (Masa):

Este elemento <input> crea un campo de entrada de número para que el usuario ingrese el valor de la masa. El atributo type especifica el tipo de entrada como "number", asegurando que solo se puedan ingresar valores numéricos. La clase form-control de Bootstrap aplica estilos predefinidos al campo de entrada. El atributo name asigna el nombre "numero2" al campo de entrada. El atributo id establece el ID del campo de entrada en "numero2", lo que permite a la etiqueta for vincularlo. El atributo placeholder proporciona una sugerencia al usuario ("Solo valores numéricos"). El atributo required hace que el campo de entrada sea obligatorio.

16. Salto de línea:

- Este elemento
 inserta un salto de línea después del campo de entrada de masa, creando un espacio vertical entre el campo de entrada y el siguiente elemento.

17. Etiqueta de resultado:

- Esta etiqueta <label> crea un texto que indica al usuario el significado del resultado que se mostrará a continuación (la fuerza en Newtons). El atributo for no se utiliza en este caso porque no está vinculado a un elemento de entrada específico.

18. Espacio para mostrar el resultado:

- Esta etiqueta <label> se utiliza como un espacio en blanco para mostrar el resultado calculado. El atributo id se establece en "resultado" para que el JavaScript pueda acceder y modificar su contenido. La clase form-control de Bootstrap aplica estilos predefinidos al campo de resultado.

19. Botón de cálculo:

- Este elemento crea un botón con el texto "Calcular". La clase btn btn-primary de Bootstrap aplica estilos para un botón primario. El atributo id se establece en "btnCalcular" para que el JavaScript pueda acceder y activar el cálculo.

En seguida vamos a agregar el código JavaScript que realiza la funcionalidad de la página HTML añadiendo interactividad y cálculos. Utiliza jQuery para esperar a que el documento esté completamente cargado y luego adjunta un detector de eventos clic al botón con el ID "btnCalcular". Al hacer clic en el botón, recupera los valores ingresados en los campos de entrada de corriente y resistencia, realiza la operación de dividir (corriente /resistencia) y muestra el resultado en el área designada.

2. Función Documento Listo:

- Este código utiliza la función `$(document).ready()` de jQuery para garantizar que el código JavaScript se ejecute solo después de que se haya cargado y analizado todo el documento HTML. Esto evita posibles errores causados por el acceso a elementos DOM antes de que estén completamente disponibles.

3. Manejador de eventos clic para el botón:

- Este código adjunta un detector de eventos clic al elemento con el ID "btnCalcular" usando `$('#btnCalcular').click()` de jQuery. La función `function()` dentro de las llaves define las acciones que se ejecutarán cuando se haga clic en el botón.

4. Obteniendo valores de entrada:

- Estas líneas de código utilizan los métodos `$('#numero1').val()` y `$('#numero2').val()` de jQuery para recuperar los valores ingresados en los campos de entrada con ID "numero1" (resistencia) y "numero2" (corriente), respectivamente. Los valores recuperados se almacenan en las variables `numero1` y `numero2`.

5. Analizando valores de entrada a enteros:

- La función `parseInt()` se utiliza para convertir los valores de cadena recuperados de los campos de entrada (`numero1` y `numero2`) en números enteros. Esto asegura que los valores se traten como datos numéricos para el cálculo. Los valores convertidos se almacenan en las variables `uno` y `dos`.

6. Realizando el cálculo:

- La operación de multiplicación se realiza dividir los valores almacenados en `uno` (resistencia) y `dos` (corriente). El resultado del cálculo se almacena en la variable `operacion`.

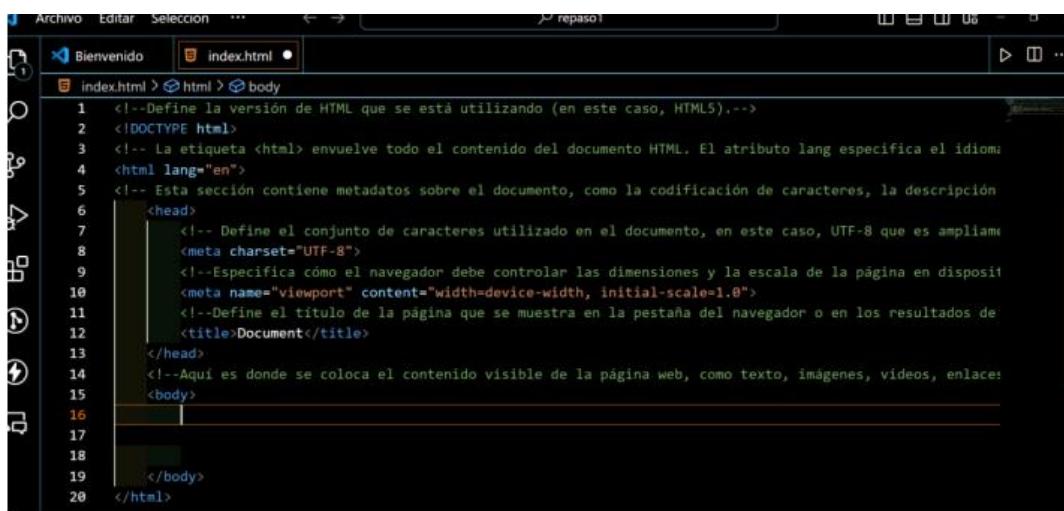
7. Mostrando el resultado calculado:

- Esta línea de código utiliza el método `$('#resultado').text()` de jQuery para insertar el valor calculado almacenado en `operacion` en el elemento con el ID "resultado". Este elemento sirve como el área designada para mostrar el resultado de la fuerza calculada al usuario.

PRÁCTICA 9: LEYES DE FÍSICA VELOCIDAD, ACCELERACIÓN Y VOLTAJE

Práctica 8 consiste en la aplicación del tener páginas enlazadas y cada una tengan las leyes de velocidad, culón y voltaje y corriente para comenzar va a ocupar el lenguaje de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, vídeos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <HTML> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.



The screenshot shows a code editor window with the title bar "Bienvenido" and "index.html". The editor displays the following HTML code:

```
1  <!--Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
2  <!DOCTYPE html>
3  <!-- La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma-->
4  <html lang="en">
5  <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción-->
6  <head>
7  <!-- Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamen-->
8  <meta charset="UTF-8">
9  <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositi-->
10 <meta name="viewport" content="width=device-width, initial-scale=1.0">
11 <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de-->
12 <title>Document</title>
13 </head>
14 <!--Aqui es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlace-->
15 <body>
16
17
18
19 </body>
20 </html>
```

Ilustración 38 ESTURUCTURA BASICA DE HTML

Continuando con el código vamos a estructurar el código HTML proporcionado que este index va a representar nuestra página principal que se estará estructurado de la siguiente manera:

- Contenedor principal: Un contenedor con la clase "container" envuelve todo el contenido de la página.
- Fila principal: Una fila dentro del contenedor principal contiene dos columnas.
- Columna lateral: Una columna con la clase "col-sm-3" ocupa 3 de 12 columnas en dispositivos pequeños y no contiene ningún elemento visible.
- Columna de contenido: Una columna con la clase "col-sm-10" ocupa 10 de 12 columnas en dispositivos pequeños y contiene todo el contenido principal de la página.
- Sección de título: Una fila dentro de la columna de contenido contiene dos columnas:
- Columna vacía: Una columna con la clase "col-sm-2" ocupa 2 de 12 columnas en dispositivos pequeños y no contiene ningún elemento visible.
- Columna de título: Una columna con la clase "col-sm-10" ocupa 10 de 12 columnas en dispositivos pequeños y contiene un título de nivel 1 (`<h1>`) con el texto "CALCULOS DE FISICA" centrado y con color rojo oscuro (#ac2424).
- Sección de cálculo: Un contenedor con la clase "drop" contiene el botón de cálculo.
- Botón de cálculo: Un botón de tipo "button" con la clase "btn btn-secondary dropdown-toggle" y el texto "calcula" tiene estilos en línea para el color de fondo (azul oscuro #42, 101, 227), la posición superior e izquierda, y la alineación de los ítems.

El código HTML utiliza varios estilos en línea para personalizar la apariencia de los elementos:

- **Imagen de fondo:** Se utiliza una imagen de fondo (`url(img/portada.jpg)`) para cubrir todo el viewport de la página. La imagen se fija al viewport (`background-attachment: fixed`) y se centra (`background-position: center center`).
- **Fuente:** Se establece una fuente sans-serif para todo el texto de la página (`font-family:sans-serif`).
- **Tamaño de fuente:** El tamaño de fuente predeterminado se establece en 16px (`font-size: 16px`).
- **Color de fondo:** Se utiliza un color de fondo blanco semitransparente (`rgba(255, 255, 255, 0.5)`) para el área de cálculo.
- **Borde:** Se añade un borde de 5 píxeles de color rojo oscuro (#843c3c) al área de cálculo.
- **Espaciado:** Se añade un espaciado interno de 10 píxeles al área de cálculo (`padding: 10px`).
- **Ancho:** Se establece un ancho de 26 píxeles para el área de cálculo (`width: 26px`).
- **Color del texto:** Se utiliza un color rojo oscuro (#ac2424) para el texto dentro del área de cálculo.
- **Margen superior:** Se añade un margen superior de 80 píxeles al área de cálculo (`margin-top: 80px`).

Consideraciones adicionales:

- El código parece ser un fragmento de una página web más grande. Se necesitaría el contexto completo de la página para comprender completamente su propósito y funcionalidad.
- El código utiliza algunos estilos en línea, lo que puede dificultar el mantenimiento y la reutilización del código. Se recomienda utilizar hojas de estilo externas para definir los estilos CSS de la página.
- El código no incluye ningún contenido real relacionado con los cálculos físicos. Se necesitaría código adicional para implementar la funcionalidad de cálculo.

```

<div class="container">
    <!-- Contenedor principal -->
    <div class="row">
        <div class="col-sm-4"></div>
        <!-- Columna vacía para espaciado -->
        <br>
        <!-- Salto de línea -->
        <div class="col-sm-4" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color de fondo con transparencia */ border: 5px solid #c0e9e9; /* Define el borde del cuadro de texto */ padding: 10px; /* Define el espacio dentro del cuadro de texto */ width: 26px; /* Define el ancho del cuadro de texto */ color: #333; /* Define el color del texto dentro del cuadro de texto */ margin-top: 100px; ">
            <!-- Columna principal para contenido -->
            <br>
            <br>
            <!-- Saltos de línea -->
            <h1 style="color: #rgb(23, 20, 20);text-align: center;">VOLTAJE</h1>
            <!-- Título principal -->
            <label for="voltaje" style="text-align: center;">Ingresa el valor de corriente:</label>
            <!-- Etiqueta para descripción del primer campo de entrada -->
            <input type="number" class="form-control" name="numero1" id="numero1" placeholder="solo valores numéricos" required>
            <!-- Campo de entrada para corriente -->
            <br>
            <!-- Saltos de línea -->
            <label for="voltaje2" style="text-align: center;">Ingresa el valor de resistencia:</label>
            <!-- Etiqueta para descripción del segundo campo de entrada -->
            <input type="number" class="form-control" name="numero2" id="numero2" placeholder="solo valores numéricos" required>
            <!-- Campo de entrada para resistencia -->
            <br>
            <br>
            <!-- Saltos de línea -->
            <span class="btn btn-primary" id="btnCalcular" name="btnCalcular">Calcular</span>
            <!-- Botón para calcular -->
            <a href="index.html"> Volver Atrás</a>
            <!-- Enlace para regresar -->
            <br>
            <!-- Salto de línea -->

```

Ilustración 39 estructura de la página principal de HTML

CORRIENTE

ESTRUCTURA GENERAL:

El código HTML y JavaScript proporcionado se utiliza para crear una página web que permite calcular la corriente eléctrica (I) utilizando la ley de Ohm ($I = V/R$), donde V es el voltaje y R es la resistencia. La página web tiene una interfaz simple con dos campos de entrada para ingresar los valores de voltaje y resistencia, un botón para realizar el cálculo y un enlace para volver a la página anterior.

Código HTML:

Estructura:

- El código HTML está estructurado utilizando HTML5 y Bootstrap 4.
- Se utiliza un contenedor principal (`<div class="container">`) para organizar el contenido de la página.
- Dentro del contenedor principal, se utiliza una fila (`<div class="row">`) para dividir el contenido en dos columnas.
- La columna de la izquierda está vacía.
- La columna de la derecha contiene el formulario para ingresar los valores de voltaje y resistencia, el botón para realizar el cálculo y el enlace para volver a la página anterior.

Estilos:

Se utilizan estilos en línea para personalizar la apariencia de la página, como el color de fondo, el color del texto, el tamaño de fuente y el margen.

- Se utiliza una imagen de fondo (url(img/q.jpg)) para cubrir todo el viewport de la página. La imagen se fija al viewport (background-attachment: fixed) y se centra (background-position: center center).
- Se utiliza una fuente sans-serif para todo el texto de la página (font-family:sans-serif).
- El tamaño de fuente predeterminado se establece en 16px (font-size: 16px).
- Se utiliza un color de fondo blanco semitransparente (rgba(255, 255, 255, 0.5)) para el área del formulario.
- Se añade un borde de 5 píxeles de color rojo oscuro (#c09e9e) al área del formulario.
- Se añade un espaciado interno de 10 píxeles al área del formulario (padding: 10px).
- Se establece un ancho de 26 píxeles para el área del formulario (width: 26px).
- Se utiliza un color rojo oscuro (#333) para el texto dentro del área del formulario.
- Se añade un margen superior de 100 píxeles al área del formulario (margin-top: 100px).

Elementos principales:

- Título: Se utiliza un encabezado de nivel 1 (

) con el texto "CORRIENTE" para mostrar el título de la página.
- Etiqueta de entrada para voltaje: Se utiliza una etiqueta para describir el campo de entrada de voltaje, y un campo de entrada de tipo number con el id "numero1" para ingresar el valor del voltaje.
- Etiqueta de entrada para resistencia: Se utiliza una etiqueta para describir el campo de entrada de resistencia, y un campo de entrada de tipo number con el id "numero2" para ingresar el valor de la resistencia.
- Botón de cálculo: Se utiliza un botón con el texto "Calcular" y el id "btnCalcular" para iniciar el cálculo de la corriente.
- Enlace para volver atrás: Se utiliza un enlace con el texto "Volver Atrás" para regresar a la página anterior.

Código JavaScript:

- Función `\$(document).ready(function(){})`:
- Esta función se ejecuta cuando el documento HTML se ha cargado completamente.
- Asigna un evento de clic al botón "Calcular" (`\$('#btnCalcular').click(function(){})`).
- Función de cálculo:
- Obtiene los valores ingresados por el usuario para el voltaje y la resistencia.
- Convierte los valores a enteros utilizando parseInt().
- Realiza la operación de división ($I = V/R$) para calcular la corriente.
- Muestra el resultado del cálculo en la consola del navegador.
- Consideraciones adicionales:
- El código HTML y JavaScript proporcionado son relativamente simples y fáciles de entender.
- La página web tiene una interfaz de usuario básica y funcional.
- El código podría mejorarse con la adición de validación de entrada para asegurarse de que los usuarios ingresen valores válidos para el voltaje y la resistencia.
- Se podría agregar un campo de salida para mostrar el resultado del cálculo a los usuarios.
- Se podría considerar el uso de una hoja de estilos CSS externa para separar el estilo del código HTML.

```

5<! Contenido del documento-->
6<!-- Contenedor principal de Bootstrap para el diseño de la página.-->
7<div class="container">
8    <!-- Fila para organizar elementos en columnas-->
9    <div class="row">
10        <!-- Columna vacía con ancho de 4 en dispositivos pequeños y superiores.-->
11        <div class="col-sm-4"></div>
12        <!-- Columna vacía con ancho de 4 en dispositivos pequeños y superiores.-->
13        <!--height: Altura del elemento.
14        background-color: Color de fondo con transparencia.
15        border: Borde del elemento.
16        padding: Espaciado interno del elemento.
17        width: Ancho del elemento.
18        color: Color del texto dentro del elemento.
19        margin-top: Margen superior del elemento.-->
20        <div class="col-sm-4" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color de fondo con transparen
21            border: 5px solid #c09e9e; /* Define el borde del cuadro de texto */
22            padding: 10px; /* Define el espacio dentro del cuadro de texto */
23            width: 26px; /* Define el ancho del cuadro de texto */
24            color: #333; /* Define el color del texto dentro del cuadro de texto */
25            margin-top: 100px;">
26            <br>
27            <br>
28            <!-- Encabezado principal para el título-->
29            <h1 style="color: #rgb(23, 20, 20); text-align: center;">CORRIENTE</h1>
30            <!--Etiqueta para describir los campos de entrada.-->
31            <label for="voltaje1">Ingresa el valor de corriente:</label>
32            <!--Campos de entrada para números.-->
33            <input type="number" class="form-control" name="numero1" id="numero1" placeholder="solo valores numéricos" required>
34
35            <br>
36            <label for="voltaje2">Ingresa el valor de resistencia:</label>
37            <input type="number" class="form-control" name="numero2" id="numero2" placeholder="solo valores numéricos" required/>
38            <br>
39            <br>
40            <span class="btn btn-primary" id="btnCalcular" name="btnCalcular">calcular</span>
41            <a href="index.html"> Volver Atrás</a>
42            <br>
43        </div>
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59

```

Ilustración 40 estructura de HTML para la página de corriente

CÓDIGO CSS

Este código CSS importa dos fuentes personalizadas desde Google Fonts y define estilos para los encabezados de nivel 1 (`<h1>`) en la página web.

- Importación de fuentes:
- Se utiliza la directiva `@import` para importar dos fuentes personalizadas desde Google Fonts:
- Concert One: Esta fuente pertenece a la familia "Concert One" y se utiliza como fuente principal para los encabezados de nivel 1.
- Jersey 10 Charted: Esta fuente pertenece a la familia "Jersey 10 Charted" y se utiliza como fuente de respaldo en caso de que la fuente principal no esté disponible.
- Estilos para encabezados de nivel 1:
- Se seleccionan los encabezados de nivel 1 (`<h1>`) utilizando el selector `h1`.
- Se definen los siguientes estilos para los encabezados de nivel 1:
- `font-family: "Concert One", sans-serif;;`
- Se establece la fuente principal como "Concert One".
- En caso de que la fuente "Concert One" no esté disponible, se utilizará una fuente sans-serif predeterminada del sistema.
- `font-weight: 400;;`
- Se establece el grosor de la fuente como normal (400).
- `font-style: normal;;`
- Se establece el estilo de la fuente como normal.

CONSIDERACIONES ADICIONALES:

- El código CSS proporcionado es relativamente simple y fácil de entender.
- Los estilos definidos para los encabezados de nivel 1 son básicos pero deberían ser suficientes para mejorar la apariencia de los encabezados en la página web.
- Se podría considerar agregar más estilos para otros elementos de la página web, como párrafos, enlaces, etc.
- Se recomienda utilizar una hoja de estilos CSS externa para separar el estilo del código HTML.
- En general, el código CSS proporcionado es un buen punto de partida para personalizar la apariencia de los encabezados de nivel 1 en una página web.

```
memex.css /* ...
@import url('https://fonts.googleapis.com/css2?family=Concert+One&family=Jersey+10+Charted&display=swap');
/* Importa las fuentes 'Concert One' y 'Jersey 10 Charted' desde Google Fonts */

h1 {
    /* Estilos para los encabezados de nivel 1 */
    font-family: "Concert One", sans-serif;
    /* Establece la fuente principal como "Concert One" */
    font-weight: 400;
    /* Define el grosor de la fuente como normal (400) */
    font-style: normal;
    /* Define el estilo de la fuente como normal */
}
```

Ilustración 41 código de CSS de la página de corriente

RESISTENCIA

Estructura general:

El código HTML proporcionado está estructurado de la siguiente manera:

- Documento HTML: Se declara el tipo de documento como HTML5 (<!DOCTYPE HTML>).
- Head: Se incluye una sección para metadatos del documento (<head>), que contiene:
 - Metadatos: Se definen metadatos para la codificación de caracteres (<meta charset="utf-8">), la vista en dispositivos móviles (<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">) y el título del documento (<title>RESISTENCIA</title>).
 - Body: Se define la sección principal del contenido (<body>) que contiene:
 - Estilo en línea: Se aplica un estilo en línea al cuerpo del documento para establecer la imagen de fondo (url(img/resis.jpg)), la fuente (font-family:sans-serif), el tamaño de fuente (font-size: 16px), el tamaño de la imagen de fondo (background-size: cover), la fijación de la imagen de fondo (background-attachment: fixed) y la posición de la imagen de fondo (background-position: center center).
 - Contenedor principal: Se utiliza un contenedor principal de Bootstrap (<div class="container">) para organizar el contenido de la página.
 - Fila: Se crea una fila Bootstrap (<div class="row">) para distribuir el contenido en columnas.
 - Columna vacía: Se incluye una columna vacía con ancho de 4 en dispositivos pequeños (<div class="col-sm-4"></div>) para crear espacio entre el contenido y el borde de la página.
 - Columna principal: Se define una columna principal con ancho de 4 en dispositivos pequeños (<div class="col-sm-4">) para contener el formulario de cálculo.

- Estilo en línea: Se aplica un estilo en línea a la columna principal para establecer la altura (height:450px), el color de fondo semitransparente (background-color: rgba(255, 255, 255, 0.5)), el borde (border: 5px solid #c09e9e), el espaciado interno (padding: 10px), el ancho (width: 26px), el color del texto (color: #333) y el margen superior (margin-top: 100px).
- Contenido del formulario:
- Saltos de línea: Se incluyen saltos de línea (
) para crear espacio entre los elementos.
- Título: Se define un título de nivel 1 (<h1>) con el texto "RESISTENCIA" y estilo en línea para el color (color: rgb(23, 20, 20)) y la alineación central (text-align: center).
- Etiqueta de entrada para corriente: Se crea una etiqueta <label> con el texto "Ingresa el valor de corriente:" para describir el campo de entrada de corriente.
- Campo de entrada para corriente: Se incluye un campo de entrada de tipo number con el id "numero1", nombre "numero1", atributo placeholder para indicar el tipo de valor esperado ("solo valores numéricos") y el atributo required para indicar que es obligatorio.
- Salto de línea: Se incluye un salto de línea (
) para crear espacio entre los elementos.
- Etiqueta de entrada para resistencia: Se crea una etiqueta <label> con el texto "Ingresa el valor de resistencia:" para describir el campo de entrada de resistencia.
- Campo de entrada para resistencia: Se incluye un campo de entrada de tipo number con el id "numero2", nombre "numero2", atributo placeholder para indicar el tipo de valor esperado ("solo valores numéricos") y el atributo required para indicar que es obligatorio.
- Salto de línea: Se incluye un salto de línea (
) para crear espacio entre los elementos.
- Botón para calcular: Se crea un botón con el texto "Calcular", id "btnCalcular", nombre "btnCalcular" y clase "btn btn-primary".
- Enlace para regresar: Se crea un enlace <a> con el texto "Volver Atrás", href "index.html" para regresar a la página anterior.
- Salto de línea: Se incluye un salto de línea (
) para crear espacio entre los elementos.

```

1  <!--Declaración del tipo de documento y su versión--&gt;
2  &lt;!DOCTYPE html&gt;
3  &lt;!--Especifica el idioma del documento--&gt;
4  &lt;html lang="en"&gt;
5      &lt;!--Contiene metadatos sobre el documento HTML--&gt;
6      &lt;head&gt;
7          &lt;!--Define el título del documento que aparecerá en la pestaña del navegador.--&gt;
8          &lt;title&gt;Title&lt;/title&gt;
9          &lt;!-- Required meta tags --&gt;
10         &lt;!--Establece la codificación de caracteres del documento en UTF-8.--&gt;
11         &lt;meta charset="utf-8"&gt;
12         &lt;!--Define las propiedades del viewport para un diseño responsive.--&gt;
13         &lt;meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"&gt;
14         &lt;link rel="stylesheet" href="resistencia.css"&gt;
15         &lt;!-- Bootstrap CSS --&gt;
16         &lt;link rel="stylesheet" href="librerias/bootstrap.min.css"&gt;
17     &lt;/head&gt;
18     &lt;body style="background-image: url(img/resis.jpg); font-family:sans-serif;font-size: 16px; background-size: cover; background-attachment: fixed; background-position: center center;&gt;
19         &lt;!-- Cuerpo del documento con estilo de fondo, fuente y tamaño de fuente definidos --&gt;
20         &lt;div class="container"&gt;
21             &lt;!-- Contenedor principal de Bootstrap --&gt;
22             &lt;div class="row"&gt;
23                 &lt;!-- Fila para organizar elementos en columnas --&gt;
24                 &lt;div class="col-sm-4"&gt;&lt;/div&gt;
25                 &lt;!-- Columna vacía para espaciado --&gt;
26                 &lt;div class="col-sm-4" style="height:450px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color de fondo con transparencia */;
border: 5px solid #c09e9e; /* Define el borde del cuadro de texto */
padding: 10px; /* Define el espaciado dentro del cuadro de texto */
width: 26px; /* Define el ancho del cuadro de texto */
color: #333; /* Define el color del texto dentro del cuadro de texto */
margin-top: 100px; &gt;
27                     &lt;!-- Columna principal para contenido --&gt;
28                     &lt;br&gt;
29                     &lt;!-- Saltos de línea para espacio --&gt;
30                     &lt;h1 style="color: #rgb(23, 20, 20); text-align: center;"&gt;RESISTENCIA&lt;/h1&gt;
31                     &lt;!-- Título principal --&gt;
32                     &lt;label for="voltaje"&gt;Ingresa el valor de corriente:&lt;/label&gt;
33                     &lt;!-- Etiqueta para descripción del primer campo de entrada --&gt;
34                     &lt;input type="number" class="form-control" name="numero1" id="numero1" placeholder="solo valores numéricos" required&gt;
35                     &lt;!-- Campo de entrada para corriente --&gt;
36                     &lt;br&gt;
37                     &lt;!-- Salto de linea --&gt;
38                     &lt;label for="voltaje2"&gt;Ingresa el valor de resistencia:&lt;/label&gt;
39                     &lt;!-- Etiqueta para descripción del segundo campo de entrada --&gt;
40                     &lt;input type="number" class="form-control" name="numero2" id="numero2" placeholder="solo valores numéricos" required&gt;
41                     &lt;!-- Campo de entrada para resistencia --&gt;
42                     &lt;br&gt;
</pre>

```

Ilustración 42 estructura de HTML de la página de resistencia agregando sus campos de label y input

CÓDIGO CSS

En general, el código CSS proporcionado es un buen punto de partida para personalizar la apariencia de los encabezados de nivel 1 en una página web.

Sin embargo, es importante tener en cuenta que el código CSS proporcionado

Este código CSS importa dos fuentes personalizadas desde Google Fonts y define estilos para los encabezados de nivel 1 (`<h1>`) en la página web.

Importación de fuentes:

- Se utiliza la directiva `@import` para importar dos fuentes personalizadas desde Google Fonts:
- Concert One: Esta fuente pertenece a la familia "Concert One" y se utiliza como fuente principal para los encabezados de nivel
- Jersey 10 Charted: Esta fuente pertenece a la familia "Jersey 10 Charted" y se utiliza como fuente de respaldo en caso de que la fuente principal no esté disponible.
- Estilos para encabezados de nivel 1:
- Se seleccionan los encabezados de nivel 1 (`<h1>`) utilizando el selector `h1`.
- Se definen los siguientes estilos para los encabezados de nivel 1:
- `font-family: "Concert One", sans-serif;`
- Se establece la fuente principal como "Concert One".
- En caso de que la fuente "Concert One" no esté disponible, se utilizará una fuente sans-serif predeterminada del sistema.
- `font-weight: 400;`
- Se establece el grosor de la fuente como normal (400).
- `font-style: normal;`
- Se establece el estilo de la fuente como normal.

Consideraciones adicionales:

- El código CSS proporcionado es relativamente simple y fácil de entender.
- Los estilos definidos para los encabezados de nivel 1 son básicos pero deberían ser suficientes para mejorar la apariencia de los encabezados en la página web.
- Se podría considerar agregar más estilos para otros elementos de la página web, como párrafos, enlaces, etc.
- Se recomienda utilizar una hoja de estilos CSS externa para separar el estilo del código HTML.

```
@import url('https://fonts.googleapis.com/css2?family=Concert+One&family=Jersey+10+Charted&display=swap');
/* Importa las fuentes 'Concert One' y 'Jersey 10 Charted' desde Google Fonts */

h1 {
    /* Estilos para los encabezados de nivel 1 */
    font-family: "Concert One", sans-serif;
    /* Establece la fuente principal como "Concert One" */
    font-weight: 400;
    /* Define el grosor de la fuente como normal (400) */
    font-style: normal;
    /* Define el estilo de la fuente como normal */
}
```

VOLTAJE

Estructura general:

El código HTML proporcionado está estructurado de la siguiente manera:

- Documento HTML: Se declara el tipo de documento como HTML5 (<!DOCTYPE HTML>).
- Head: Se incluye una sección para metadatos del documento (<head>), que contiene:
 - Metadatos: Se definen metadatos para la codificación de caracteres (<meta charset="utf-8">), la vista en dispositivos móviles (<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">) y el título del documento (<title>RESISTENCIA</title>).
- Body: Se define la sección principal del contenido (<body>) que contiene:
 - Estilo en línea: Se aplica un estilo en línea al cuerpo del documento para establecer la imagen de fondo (url(img/resis.jpg)), la fuente (font-family:sans-serif), el tamaño de fuente (font-size: 16px), el tamaño de la imagen de fondo (background-size: cover), la fijación de la imagen de fondo (background-attachment: fixed) y la posición de la imagen de fondo (background-position: center center).
 - Contenedor principal: Se utiliza un contenedor principal de Bootstrap (<div class="container">) para organizar el contenido de la página.
 - Fila: Se crea una fila Bootstrap (<div class="row">) para distribuir el contenido en columnas.
 - Columna vacía: Se incluye una columna vacía con ancho de 4 en dispositivos pequeños (<div class="col-sm-4"></div>) para crear espacio entre el contenido y el borde de la página.
 - Columna principal: Se define una columna principal con ancho de 4 en dispositivos pequeños (<div class="col-sm-4">) para contener el formulario de cálculo.
 - Estilo en línea: Se aplica un estilo en línea a la columna principal para establecer la altura (height:450px), el color de fondo semitransparente (background-color: rgba(255, 255, 255, 0.5)), el borde (border: 5px solid #c09e9e), el espaciado interno (padding: 10px), el ancho (width: 26px), el color del texto (color: #333) y el margen superior (margin-top: 100px).
- Contenido del formulario:
 - Saltos de línea: Se incluyen saltos de línea (
) para crear espacio entre los elementos.
 - Título: Se define un título de nivel 1 (<h1>) con el texto "RESISTENCIA" y estilos en línea para el color (color: rgb(23, 20, 20)) y la alineación central (text-align: center).
 - Etiqueta de entrada para corriente: Se crea una etiqueta <label> con el texto "Ingresa el valor de corriente:" para describir el campo de entrada de corriente.
 - Campo de entrada para corriente: Se incluye un campo de entrada de tipo number con el id "numero1", nombre "numero1", atributo placeholder para indicar el tipo de valor esperado ("solo valores numéricos") y el atributo required para indicar que es obligatorio.
 - Salto de línea: Se incluye un salto de línea (
) para crear espacio entre los elementos.
 - Etiqueta de entrada para resistencia: Se crea una etiqueta <label> con el texto "Ingresa el valor de resistencia:" para describir el campo de entrada de resistencia.
 - Campo de entrada para resistencia: Se incluye un campo de entrada de tipo number con el id "numero2", nombre "numero2", atributo placeholder para indicar el tipo de valor esperado ("solo valores numéricos") y el atributo required para indicar que es obligatorio.
 - Salto de línea: Se incluye un salto de línea (
) para crear espacio entre los elementos.
 - Botón para calcular: Se crea un botón con el texto "Calcular", id "btnCalcular", nombre "btnCalcular" y clase "btn btn-primary".

- Enlace para regresar: Se crea un enlace <a> con el texto "Volver Atrás", href "index.html" para regresar a la página anterior.
- Salto de línea: Se incluye un salto de línea (
) para crear espacio entre los elementos.

```

1  <!-- Declaración del tipo de documento y su versión -->
2  <!DOCTYPE html>
3  <!-- Especifica el idioma del documento.-->
4  <html lang="en">
5      <!-- Contiene metadatos sobre el documento HTML-->
6      <head>
7          <!-- Define el título del documento que aparecerá en la pestaña del navegador.-->
8          <title>Title</title>
9          <!-- Required meta tags -->
10         <!-- Establece la codificación de caracteres del documento en UTF-8-->
11         <meta charset="utf-8">
12         <!-- Define las propiedades del viewport para un diseño responsive.-->
13         <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
14         <link rel="stylesheet" href="resistencia.css">
15         <!-- Bootstrap CSS -->
16         <link rel="stylesheet" href="librerias/bootstrap.min.css">
17     </head>
18     <body style="background-image: url(img/resis.jpg); font-family:sans-serif;font-size: 16px; background-size: cover; background-attachment: fixed; background-position: center center;>
19         <!-- Cuerpo del documento con estilo de fondo, fuente y tamaño de fuente definidos -->
20         <div class="container">
21             <!-- Contenedor principal de Bootstrap -->
22             <div class="row">
23                 <!-- Fila para organizar elementos en columnas -->
24                 <div class="col-sm-4">
25                     <!-- Columna vacía para espacio -->
26                     <div class="col-sm-4" Style="height:458px;background-color: #rgba(255, 255, 255, 0.5); /* Define el color de fondo con transparencia */ border: 5px solid #C0E9E9; /* Define el borde del cuadro de texto */ padding: 10px; /* Define el espacio dentro del cuadro de texto */ width: 26px; /* Define el ancho del cuadro de texto */ color: #333; /* Define el color del texto dentro del cuadro de texto */ margin-top: 100px;">
27                         <!-- Columna principal para contenido -->
28                         <br>
29                         <!-- Saltos de línea para espacio -->
30                         <h1 style="color: #rgb(23, 20, 20); text-align: center;">RESISTENCIA

```

Ilustración 43 estructura de HTML de la página de resistencia agregando sus campos de label y input

CÓDIGO CSS

En general, el código CSS proporcionado es un buen punto de partida para personalizar la apariencia de los encabezados de nivel 1 en una página web.

Sin embargo, es importante tener en cuenta que el código CSS proporcionado

Este código CSS importa dos fuentes personalizadas desde Google Fonts y define estilos para los encabezados de nivel 1 (<h1>) en la página web.

Importación de fuentes:

- Se utiliza la directiva @import para importar dos fuentes personalizadas desde Google Fonts:
- Concert One: Esta fuente pertenece a la familia "Concert One" y se utiliza como fuente principal para los encabezados de nivel
- Jersey 10 Charted: Esta fuente pertenece a la familia "Jersey 10 Charted" y se utiliza como fuente de respaldo en caso de que la fuente principal no esté disponible.
- Estilos para encabezados de nivel 1:
- Se seleccionan los encabezados de nivel 1 (<h1>) utilizando el selector h1.
- Se definen los siguientes estilos para los encabezados de nivel 1:
- font-family: "Concert One", sans-serif;
- Se establece la fuente principal como "Concert One".
- En caso de que la fuente "Concert One" no esté disponible, se utilizará una fuente sans-serif predeterminada del sistema.

- `font-weight: 400;`
- Se establece el grosor de la fuente como normal (400).
- `font-style: normal;`
- Se establece el estilo de la fuente como normal.

Consideraciones adicionales:

- El código CSS proporcionado es relativamente simple y fácil de entender.
- Los estilos definidos para los encabezados de nivel 1 son básicos pero deberían ser suficientes para mejorar la apariencia de los encabezados en la página web.
- Se podría considerar agregar más estilos para otros elementos de la página web, como párrafos, enlaces, etc.
- Se recomienda utilizar una hoja de estilos CSS externa para separar el estilo del código HTML.

```
@import url('https://fonts.googleapis.com/css2?family=Concert+One&family=Jersey+10+Charted&display=swap');

/* Importa las fuentes 'Concert One' y 'Jersey 10 Charted' desde Google Fonts */

h1 {
    /* Estilo para los encabezados de nivel 1 */
    font-family: "Concert One", sans-serif;
    /* Establece la fuente principal como "Concert One" */
    font-weight: 400;
    /* Define el grosor de la fuente como normal (400) */
    font-style: normal;
    /* Define el estilo de la fuente como normal */
}
```

Ilustración 44 Código de css de la página voltaje

PRACTICA 10: CARRUSEL

Practica 10: consiste en la aplicación de un carrusel, que su funcionamiento el cual es que al dar clic al mouse se cambie la imagen o ya sea automáticamente, para esto llevaremos a cabo empezando con el lenguaje estructurado de HTML en el cual son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, el elemento de <Canvas> Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <HTML> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

```
index.html # style.css
index.html > html > head
1  <!--Declara la versión de HTML que el documento está utilizando.-->
2  <!DOCTYPE html>
3  <!--Abre la etiqueta raíz del documento HTML y especifica el idioma principal del contenido (en este caso, inglés-->
4  <html lang="en">
5  <!--Abre la sección de encabezado del documento, donde se colocan metadatos y enlaces a recursos externos.-->
6  <head>
7      <!--Especifica el conjunto de caracteres (UTF-8) utilizado para codificar el contenido del documento-->
8      <meta charset="UTF-8">
9      <!--Define cómo el navegador debe controlar las dimensiones y la escala de la página en diferentes dispositivos-->
10     <meta name="viewport" content="width=device-width, initial-scale=1.0">
11     <!--Establece el título del documento, que se muestra en la barra de título del navegador o en la pestaña del navegador-->
12     <title>Document</title>
13     <!--Enlaza la hoja de estilos externa "style.css" con el documento HTML para aplicar estilos al contenido-->
14     <link rel="stylesheet" href="style.css">
15     <!--Cierra la sección de encabezado del documento.-->
16 </head>
17 <!--Abre la sección del cuerpo del documento, donde se coloca el contenido visible del sitio web.-->
18 <body>
19     <!--Inserta un elemento <canvas> en el cuerpo del documento con el ID "pizarra". Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript-->
20     <canvas id="pizarra"></canvas>
21 </body>
22 <!--Cierra la sección del cuerpo del documento.-->
23 </html>
24 <!--Cierra la etiqueta <html> del documento HTML.-->
```

Ilustración 45 ESTRUCTURA BASICA DE HTML

1. Barra superior con imagen:

- Un contenedor div con la clase container encierra toda la sección.
- Dentro de este contenedor, una fila (div con la clase row) contiene tres columnas (div con la clase col-sm-3, col-sm-4 y col-sm-4).
- La columna central contiene una imagen cargada desde el archivo imágenes/holi.png. El atributo src de la etiqueta especifica la ruta de la imagen, mientras que alt proporciona un texto alternativo para la accesibilidad.

2. Carrusel de imágenes:

- Otro contenedor div con la clase container encierra la sección del carrusel.
- Una fila (div con la clase row) contiene tres columnas (div con la clase col-sm-1, col y col-sm-1).
- La columna central contiene el carrusel, identificado por id="carouselExampleIndicators" y con la clase carousel slide y data-ride="carousel".
- Dentro del carrusel:
 - Una lista ordenada (ol con la clase carousel-indicators) contiene tres elementos li que funcionan como indicadores de las diapositivas.
 - Un contenedor principal (div con la clase carousel-inner) aloja las diapositivas del carrusel.
 - Cada diapositiva (div con la clase carousel-item) contiene una imagen (img). Las imágenes se cargan desde los archivos imágenes/dca59p6-ca187b0c-0c05-40cf-9150-e73a85fdc299.png, imágenes/desktop-wallpaper-whis-dragon-ball-super-by-nekoar-dragon-ball-super-whis-android.jpg, y imágenes/Captura.PNG.
 - Dos enlaces (a con las clases carousel-control-prev y carousel-control-next) sirven como controles para navegar entre las diapositivas.

Estilos CSS:

El código HTML y CSS proporcionado crea una página web con una barra superior con una imagen y un carrusel de imágenes. Se recomienda revisar y ajustar algunos aspectos como la accesibilidad, la optimización de imágenes y las pruebas en diferentes entornos para garantizar una experiencia de usuario óptima. El código CSS define los estilos visuales para los elementos HTML. Algunos aspectos destacados incluyen:

- Clases de Bootstrap: Se utilizan varias clases de Bootstrap para estructurar el diseño, como row, col, carousel slide, carousel-indicators, carousel-inner, carousel-item, d-block, w-100, carousel-control-prev, y carousel-control-next.
- Tamaño de las imágenes: Las imágenes del carrusel se ajustan para ocupar todo el ancho del contenedor (w-100) y tienen una altura fija de 550px.
- Márgenes: Se agregan márgenes para crear espacios entre los elementos.

Consideraciones adicionales:

- Accesibilidad: El texto alternativo (alt) para las imágenes es crucial para la accesibilidad en caso de que las imágenes no se carguen o sean difíciles de ver.
- Responsividad: El uso de clases de Bootstrap y estilos CSS adaptables debería asegurar que el diseño se ajuste correctamente a diferentes tamaños de pantalla.
- Optimización de imágenes: Se recomienda optimizar las imágenes para reducir su tamaño y mejorar la velocidad de carga de la página.
- Pruebas y ajustes: Es importante probar el código en diferentes navegadores y dispositivos para verificar su correcto funcionamiento y realizar ajustes si es necesario.

```

<div class="container">
  <!-- Esta clase de Bootstrap se utiliza para crear una fila en la que se colocarán las columnas. -->
  <div class="row">
    <!-- Esta clase de Bootstrap se utiliza para crear una columna -->
    <div class="col-sm-1"></div>
    <!-- Esta clase de Bootstrap se utiliza para crear una columna -->
    <div class="col">
      <!-- Define un identificador único para el carrusel, lo que permite que los controles y los indicadores se asocien correctamente-->
      <div id="carouselExampleIndicators" class="carousel slide" data-ride="carousel">
        <!-- Define una lista ordenada. -->
        <ol class="carousel-indicators">
          <!-- Especifica el carrusel al que se refiere el indicador. -->
          <li data-target="#carouselExampleIndicators" data-slide-to="0" class="active"></li>
          <li data-target="#carouselExampleIndicators" data-slide-to="1"></li>
          <li data-target="#carouselExampleIndicators" data-slide-to="2"></li>
        </ol>
        <!--Este es el contenedor principal del carrusel que contiene todas las diapositivas.-->
        <div class="carousel-inner">
          <!--Una imagen dentro de la diapositiva, con clases de Bootstrap para controlar su apariencia y comportamiento-->
          <div class="carousel-item active">
            <!--Una imagen dentro de la diapositiva, con clases de Bootstrap para controlar su apariencia y comportamiento-->
            
          </div>
          <!--esta es una diapositiva del carrusel. Contiene una imagen que ocupa todo el ancho del contenedor (clase "w-100")-->
          <div class="carousel-item">
            <!--Otra diapositiva del carrusel. También contiene una imagen con las mismas características que la anterior. La imagen ocupa todo el ancho del contenedor (clase "w-100")-->
            
          </div>
          <!--Un enlace que sirve como control para ir a la diapositiva anterior en el carrusel. Contiene un ícono de flecha hacia la izquierda-->
          <a class="carousel-control-prev" href="#carouselExampleIndicators" role="button" data-slide="prev">
            <span class="carousel-control-prev-icon" aria-hidden="true"></span>
            <span class="sr-only">Previous</span>
          </a>
        </div>
      </div>
    </div>
  </div>

```

Ilustración 46CONTENIDO DE CODIGO PAAR ESTRUCTURAR EL CORRUSEL

Función del código:

Este fragmento de código JavaScript se utiliza para controlar dinámicamente una propiedad CSS personalizada (variable CSS) llamada `--pos` en base al valor de un elemento range (probablemente un control deslizante de tipo input). Este código JavaScript permite vincular el valor de un control deslizante (elemento range) con una propiedad CSS personalizada (`--pos`) en tiempo real. Cualquier elemento que use la variable CSS `--pos` en su estilo se actualizará dinámicamente al mover el control deslizante.

Explicación paso a paso:

Evento oninput:

- Se asigna una función de flecha al evento oninput del elemento range.
- El evento oninput se dispara cada vez que el valor del elemento range cambia (por ejemplo, cuando el usuario mueve el control deslizante).

1. Acceso al valor del range:

- Dentro de la función de flecha, se accede al valor actual del elemento range utilizando la propiedad value.

2. Actualización de la propiedad CSS personalizada:

- Se utiliza el método document.body.style.setProperty para modificar el valor de la propiedad CSS personalizada `--pos`.
- El primer argumento ('`--pos`') indica el nombre de la propiedad CSS personalizada que se desea modificar.
- El segundo argumento (`range.value + '%'`) establece el nuevo valor de la propiedad. Se concatena el valor del range (un número) con el símbolo % para expresar un valor porcentual.

Consideraciones adicionales:

- Propósito de la propiedad CSS --pos: Sin el código CSS asociado, es difícil determinar con exactitud cómo se utiliza la propiedad --pos. Probablemente se use para controlar la posición de un elemento en la página web, como la transformación translateX o translateY.
- Seleccionar el elemento correcto: El código actualiza la propiedad en document.body. Si se desea aplicar el cambio a un elemento específico, se debe acceder a ese elemento mediante su identificador o clase y luego utilizar su propiedad style.

```
<!--Un script JavaScript que parece estar incompleto o fuera de lugar. No hay una variable range definida en este fragmento de código, lo que puede
<script>
    range.oninput = () =>
        document.body.style.setProperty('--pos', range.value + '%')
</script>
```

Ilustración 47 CODIGO DE JAVASCRIPT

PRÁCTICA 11: JUEGO DE ADIVINA

Practica 11 consiste en la aplicación de un juego de adivinar el número que el usuario tiene que adivinar este juego. tienen 5 intentos, también cuenta con una pista de un número cercano para adivinar, para comenzar va a ocupar el lenguaje de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, vídeos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <HTML> envuelve todo el contenido del documento HTML. El atributo langa especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

```

1 <!--Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
2 <!DOCTYPE html>
3 <!-- La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma: en-->
4 <html lang="en">
5 <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción-->
6 <head>
7   <!-- Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente-->
8   <meta charset="UTF-8">
9   <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos-->
10  <meta name="viewport" content="width=device-width, initial-scale=1.0">
11   <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda-->
12  <title>Document</title>
13 </head>
14 <!--Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.-->
15 <body>
16
17
18
19 </body>
20 </html>

```

Ilustración 48 ESTURUCTURA BASICA DE HTML

Estructura de la página:

El código HTML define la estructura de una página web para un juego de "Adivina el número". La página está dividida en tres secciones principales utilizando filas (div con la clase row) y columnas (div con las clases col-sm-4):

1. Sección central:

- ✓ La columna central (col-sm-4) con estilo background-position: center center; contiene el título del juego "Adivina el Número" centrado (h1).
- ✓ Debajo del título, se muestra una imagen cargada desde el archivo IMAGENES/fb2cbe0595615d8e873a485df8cad81e.png con un ancho de 400px.

2. Instrucciones:

- ✓ La columna derecha (col-sm-4) contiene un formulario (div con la clase form) con texto e instrucciones para el juego:
- ✓ Se explica el objetivo de encontrar un número entre 1 y 10.
- ✓ Se muestra una etiqueta centrada "ADIVINA EL NUMERO:" indicando el campo de entrada.

3. Entrada y resultados:

- ✓ La columna derecha (col-sm-4) también contiene un contenedor adicional (div con la clase resultParas):
- ✓ Un campo de texto (input con tipo "text", id="guessField" y clase guessField) permite ingresar la suposición del usuario.
- ✓ Un botón (input con tipo "submit", valor "Enviar respuesta" y clase guessSubmit) envía la suposición.
- ✓ Tres párrafos vacíos (p con clases guesses, lastResult, y lowOrHi) se reservan para mostrar resultados del juego (intentos, último resultado, mayor o menor).

Consideraciones adicionales:

- ✓ **Falta de código JavaScript:** Este código HTML proporciona la estructura de la página, pero requiere código JavaScript adicional para generar el número aleatorio, procesar las suposiciones del usuario, actualizar los resultados y mostrar mensajes de acierto o error.
- ✓ **Accesibilidad:** El texto alternativo (`alt`) para la imagen podría ser útil para describir la imagen en caso de que no se cargue o para usuarios con lectores de pantalla.
- ✓ **Optimización de imágenes:** Se recomienda optimizar la imagen para reducir su tamaño y mejorar la velocidad de carga de la página.
- ✓ **Estilos CSS:** Se puede agregar una hoja de estilos CSS separada para definir estilos visuales personalizados para los elementos de la página.

El código HTML proporcionado crea una estructura básica para un juego de "Adivina el número". Se necesita código JavaScript adicional para implementar la funcionalidad del juego y potencialmente una hoja de estilos CSS para mejorar la presentación visual. Este código CSS define estilos básicos para la página web del juego. Importa una fuente personalizada para los encabezados, establece el ancho y centra el contenido del cuerpo, y da formato al texto que muestra el último resultado del usuario.

Funcionalidad del juego: Comenzamos con definir las variables de nuestro código, este código JavaScript implementa la lógica del juego "Adivina el número" para la página web con la estructura HTML descrita anteriormente.

Variables clave:

- ✓ **randomNumber:** Almacena el número aleatorio generado entre 1 y 6 (cambiado a 1 y 3 para este ejemplo).
- ✓ **guesses:** Referencia al elemento HTML con la clase `guesses` para mostrar los intentos del usuario.
- ✓ **lastResult:** Referencia al elemento HTML con la clase `lastResult` para mostrar el resultado de cada intento.
- ✓ **lowOrHi:** Referencia al elemento HTML con la clase `lowOrHi` para indicar si el número ingresado es muy bajo o muy alto.
- ✓ **guessSubmit:** Referencia al elemento HTML con la clase `guessSubmit` (botón "Enviar respuesta").
- ✓ **guessField:** Referencia al elemento HTML con la clase `guessField` (campo de texto para ingresar la suposición).
- ✓ **guessCount:** Contador para llevar registro de los intentos realizados.
- ✓ **resetButton:** Variable para almacenar el botón de reinicio del juego (creado dinámicamente).

Funciones principales:

- **checkGuess():**
 - ✓ Esta función se ejecuta al hacer clic en el botón "Enviar respuesta".
 - ✓ Obtiene la suposición del usuario como número y la almacena en `userGuess`.
 - ✓ Actualiza el texto de `guesses` para mostrar el historial de intentos.
- **Comprueba si la suposición es correcta:**
 - ✓ Si se adivina el número, muestra un mensaje de felicitación, cambia el color de fondo a verde y desactiva el juego.
 - ✓ Si se llega al límite de intentos (5), muestra un mensaje de "Fin del juego" y desactiva el juego.
 - ✓ Si la suposición es incorrecta, indica si el número es muy bajo o muy alto y cambia el color de fondo a rojo.
 - ✓ Incrementa el contador de intentos (`guessCount`).
 - ✓ Limpia el campo de texto para la siguiente suposición y establece el foco en él.

- **setGameOver():**
- ✓ Esta función se llama cuando se adivina el número o se acaban los intentos.
- ✓ Deshabilita el campo de texto y el botón de enviar.
- ✓ Crea un botón nuevo "Jugar de nuevo" y lo agrega a la página.
- ✓ Agrega un evento clic al botón de reinicio para llamar a la función resetGame.
- ✓ resetGame(): Esta función se ejecuta al hacer clic en el botón "Jugar de nuevo".
- ✓ Reinicia el contador de intentos.
- ✓ Limpia el texto de los elementos que muestran intentos, resultados e indicaciones.
- ✓ Habilita el campo de texto y el botón de enviar.
- ✓ Elimina el botón de reinicio de la página.

Eventos:

- Se agrega un escuchador de eventos click al botón "Enviar respuesta" para ejecutar la función checkGuess cuando se hace clic.

Mejoras potenciales:

- ✓ El código original generaba un número aleatorio entre 1 y 6. Se ha modificado para generar un número entre 1 y 3 para fines de demostración y facilitar ganar el juego en pocas iteraciones.
- ✓ Se puede mejorar la interfaz de usuario agregando estilos CSS para dar formato a los elementos HTML y mejorar la presentación visual del juego.
- ✓ Se podría permitir a los usuarios elegir el rango del número aleatorio para aumentar la dificultad.
- ✓ Se puede implementar un contador para mostrar el tiempo restante para adivinar el número y agregar un elemento de presión al juego.

El código JavaScript proporcionado complementa el código HTML para crear un juego funcional de "Adivina el número". El juego genera un número aleatorio, permite a los usuarios ingresar suposiciones, proporciona retroalimentación e indica si se adivinó el número o se agotaron los intentos.

Este código CSS define estilos para la página web del juego "Adivina el número":

Importación de fuentes:

- ✓ `@import url('https://fonts.googleapis.com/css2?family=Jersey+10+Charted&display=swap');` Importa la fuente "Jersey 10 Charted" de Google Fonts.

Estilos generales:

- ✓ `HTML { font-family: sans-serif; }`: Establece la fuente predeterminada para la página como sans-serif.

Estilos para el cuerpo (body):

- ✓ body { : Define estilos para el cuerpo de la página;
- ✓ width: 50%;: Establece el ancho del cuerpo al 50% del ancho del viewport (ventana del navegador).
- ✓ max-width: 800px;: Define un ancho máximo de 800px para el cuerpo.
- ✓ min-width: 480px;: Define un ancho mínimo de 480px para el cuerpo.
- ✓ margin: 0 auto;: Centra el contenido del cuerpo horizontalmente dentro de la ventana del navegador.

Estilos para la clase .lastResult:

- ✓ .lastResult { : Define estilos para la clase .lastResult, probablemente utilizada en un elemento que muestra el resultado del último intento del usuario;
- ✓ color: white;: Establece el color del texto en blanco.
- ✓ padding: 3px;: Agrega un margen interior de 3px al elemento.

Estilos para encabezados <h1>:

- ✓ h1{ : Define estilos para los encabezados <h1>;
- ✓ font-family: "Jersey 10 Charted", sans-serif;: Establece la fuente del encabezado como "Jersey 10 Charted". Si esta fuente no está disponible, se usará una fuente sans-serif predeterminada.

PRÁCTICA 12: MEMORAMA

Practica 1: consiste en la aplicación de una pizarra, que su funcionamiento el cual es que al dar clic al mouse empezara a dibujar en la pizarra, para esto llevaremos a cabo comenzando con el lenguaje estructurado de HTML en el cual son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, el elemento de <Canvas> Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <HTML> envuelve todo el contenido del documento HTML. El atributo langa especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

```

1 <!--Declara la versión de HTML que el documento está utilizando.-->
2 <!DOCTYPE html>
3 <!--Abre la etiqueta raíz del documento HTML y especifica el idioma principal del contenido (en este caso, inglés-->
4 <html lang="en">
5 <!--Abre la sección de encabezado del documento, donde se colocan metadatos y enlaces a recursos externos.-->
6 <head>
7   <!--Especifica el conjunto de caracteres (UTF-8) utilizado para codificar el contenido del documento-->
8   <meta charset="UTF-8">
9   <!--Define cómo el navegador debe controlar las dimensiones y la escala de la página en diferentes dispositivos-->
10  <meta name="viewport" content="width=device-width, initial-scale=1.0">
11  <!--Establece el título del documento, que se muestra en la barra de título del navegador o en la pestaña del-->
12  <title>Document</title>
13  <!--Enlaza la hoja de estilos externa "style.css" con el documento HTML para aplicar estilos al contenido.-->
14  <link rel="stylesheet" href="style.css">
15  <!--Cierra la sección de encabezado del documento.-->
16 </head>
17 <!--Abre la sección del cuerpo del documento, donde se coloca el contenido visible del sitio web.-->
18 <body>
19   <!--Inserta un elemento <canvas> en el cuerpo del documento con el ID "pizarra". Este elemento se utilizará p-->
20   <canvas id="pizarra"></canvas>
21 </body>
22 <!--Cierra la sección del cuerpo del documento.-->
23 </html>
24 <!--Cierra la etiqueta raíz del documento HTML.-->

```

Ilustración 49 Estructura básica de HTML

Este código HTML representa la sección de un juego de memoria basado en tarjetas. Veamos la estructura y elementos principales:

Estructura del código:

- El código está contenido dentro de un contenedor principal con la clase "container" de Bootstrap.
- Dentro del contenedor principal, se define una fila con la clase "row" de Bootstrap para distribuir el contenido en columnas.
- La fila contiene dos columnas:
- La primera columna (<div class="col"></div>) está vacía en este código.
- La segunda columna (<div class="col">) contiene la sección del juego de memoria.
- Sección del juego de memoria:
- La sección del juego de memoria está contenida dentro de un elemento <div> con la clase "interaction".
- Dentro del elemento "interaction", se incluye otra sección con la clase "memory-game" que presumiblemente define el juego.
- Se utilizan saltos de línea (
) para espaciar los elementos del juego.

Tarjetas del juego:

- Se utilizan múltiples elementos <div> con la clase "memory-card" para representar cada tarjeta del juego.
- Cada tarjeta tiene los siguientes elementos:
- Un atributo data-group que presumiblemente define el grupo o pareja de la tarjeta.
- Dos elementos :
- El primer img con la clase "front-face no-lazyload" representa la imagen frontal de la tarjeta. Su atributo src define la ruta a la imagen.
- El segundo img con la clase "back-face no-lazyload" representa la imagen trasera de la tarjeta que se muestra inicialmente. Su atributo src también define la ruta a la imagen.

- En este código se presentan 10 tarjetas divididas en 5 parejas con las imágenes.
 - Se repiten dos veces las imágenes de cada personaje para formar las parejas.

Consideraciones adicionales:

- El código HTML proporcionado solo representa una sección del juego de memoria.
 - Sería necesario analizar el código CSS para entender los estilos aplicados a los elementos del juego.
 - Posiblemente exista código JavaScript adicional para controlar la funcionalidad del juego, como voltear las cartas, detectar parejas y verificar el fin del juego.
 - En resumen, este código HTML muestra la estructura básica de un juego de memoria con tarjetas que tienen imágenes frontales y traseras. El código utiliza clases de Bootstrap para el diseño y atributos para definir las características de cada tarjeta.

```
18 </div>
19 <div class="memory-card">
20   
21   
22 </div>
23 <br>
24
25 <div class="memory-card" data-group="balero">
26   
27   
28 </div>
29 <br>
30
31 <div class="memory-card" data-group="maracas">
32   
33   
34 </div>
35 <br>
36 <div class="memory-card" data-group="maracas">
37   
38   
39 </div>
40 <br>
41 <div class="memory-card" data-group="munecca">
42   
43   
44 </div>
45 <br>
46 <div class="memory-card" data-group="munecca">
47   
48   
49 </div>
50 <br>
51 <div class="memory-card" data-group="pirinola">
52   
53   
54 </div>
55 <br>
56 <div class="memory-card" data-group="pirinola">
```

Ilustración 50 contenido de body en el estructura del juego de memorama

CÓDIGO CSS

Este código CSS define los estilos para los elementos del juego de memoria que vimos en el código HTML anterior.

Estilos generales:

.interaction:

- Se comenta la propiedad height: 100vh; que posiblemente se utilizaba para ocupar el alto total de la ventana.
- Se establece display: flex; para que los elementos hijos se posicionen en fila.

.memory-game:

- Se define un ancho de 640px para el contenedor principal del juego.
- Se comenta la propiedad height: 640px; que posiblemente se utilizaba para fijar el alto del contenedor.
- Se aplica margin: auto; para centrar el contenedor horizontalmente.
- Se establece display: flex; para que los elementos hijos (tarjetas) se posicionen en fila.
- Se usa flex-wrap: wrap; para permitir que las tarjetas se envuelvan en varias filas si el espacio no es suficiente.
- Se aplican propiedades perspective para crear un efecto 3D en las tarjetas al voltearlas.

Estilos para las tarjetas:

.memory-card:

- Se calcula el ancho de la tarjeta restando un margen de 10px al 25% del ancho disponible.
- Se asigna un alto fijo de 300px a la tarjeta.
- Se agrega un margen de 5px alrededor de cada tarjeta.
- Se establece position: relative; para permitir el posicionamiento absoluto de los elementos front-face y back-face (caras de la tarjeta).
- Se aplica una transformación inicial transform: scale(1); para establecer el tamaño inicial de la tarjeta.
- Se usa transform-style: preserve-3d; para mantener el efecto 3D en las caras de la tarjeta.
- Se define una transición transition: transform .5s; para animar el cambio de tamaño de la tarjeta al hacer clic.
- Se comentan propiedades para aplicar sombra a las tarjetas.
- Se agrega un margen superior de -20px para ajustar la posición vertical de las tarjetas.

.memory-card:active:

- Define el estilo de la tarjeta cuando está presionada (click sostenido).
- Aplica una transformación transform: scale(0.97); para reducir ligeramente el tamaño de la tarjeta.
- Define una transición más corta transition: transform .2s; para la animación al presionar.

Estilos para el botón de reinicio:

#reset:

- Define el estilo para un botón de reinicio (posiblemente no mostrado en el código HTML proporcionado).
- Se posiciona de forma absoluta (position: absolute) en la esquina superior izquierda (top: 0; left: 0;).
- Se aplican estilos para el fondo (background-color: skyblue), color de texto (color: white), relleno (padding: 10px), bordes redondeados (border-radius: 5px), ausencia de borde (border: none), cursor (cursor: pointer) que indica interacción y margen (margin: 10px).
- Se alinea el texto al centro (text-align: center).
- Estilos para el contador y temporizador:
- #counter, #timer:

- Se agrega un margen superior de -350px (valor negativo) para posicionar estos elementos fuera de la vista superior probablemente para ocultarlos inicialmente.
- Se establece un tamaño de fuente grande (font-size: large;) para el contador y temporizador.
- Estilos para pantallas pequeñas:
- @media (max-width: 767.98px):
- Esta media query aplica estilos para pantallas con ancho máximo de 767.98px (tablets y celulares).
- Se ajusta el ancho de las tarjetas a un tercio del ancho disponible restando el margen (calc(33.333% - 10px)) para una mejor visualización en pantallas pequeñas.
- Estilos para las caras de las tarjetas:
- .front-face, .back-face:
- Se define el ancho al 100% del contenedor de la tarjeta.
- Se comenta la propiedad height: 100%; que posiblemente se utilizaba para ocupar todo el alto de la tarjeta.
- Se establece un alto automático (height: auto;) para que las imágenes se ajusten proporcional

```

EXPLORADOR    androidchim - holics.css
MEMORAMA
  > vscode
  > IMG
  # holics
  ○ index.html
  JS script.js

# holics > ...
  1  /* Interacción */
  2  .interaction {
  3    display: flex;
  4  }
  5
  6  /* Juego de memoria */
  7  .memory-game {
  8    width: 640px;
  9    margin: auto;
10    display: flex;
11    flex-wrap: wrap;
12    perspective: 1000px;
13  }
14
15  /* Tarjeta de memoria */
16  .memory-card {
17    width: calc(25% - 10px);
18    height: 100px;
19    margin: 5px;
20    position: relative;
21    transform: scale(1);
22    transform-style: preserve-3d;
23    transition: transform 0.5s;
24  }
25
26  /* Estilos para el botón de reset */
27  #reset {
28    position: absolute;
29    top: 0;
30    left: 0;
31    background-color: skyblue;
32    color: white;
33    padding: 10px;
34    border-radius: 5px;
35    border: none;
36    cursor: pointer;
37    margin: 10px;
38    text-align: center;
39  }
40
41  /* Estilos para el contador y timer */
42  #counter, #timer {
43    margin-top: -350px;
44    font-size: large;
45  }

```

Ilustración 51 código de css para el memorama

CÓDIGO JAVASCRIPT

Este código JavaScript define la funcionalidad del juego de memoria que vimos anteriormente. Se encarga de controlar el comportamiento de las tarjetas, el temporizador, el contador de aciertos y la victoria del juego.

Variables principales:

- cards: Almacena una lista con todas las tarjetas del juego obtenidas usando `querySelectorAll('.memory-card')`.
- hasFlippedCard: Bandera booleana (true o false) que indica si se ha volteado una carta actualmente.
- lockBoard: Bandera booleana que indica si el tablero está bloqueado para evitar interacciones mientras se comprueban parejas.
- firstCard y secondCard: Variables para almacenar las referencias a las dos últimas cartas volteadoras.
- counter: Variable entera para llevar la cuenta de las parejas acertadas.
- timer: Variable entera para almacenar el tiempo transcurrido en segundos.
- intervalId: Variable para almacenar el identificador del intervalo usado en el temporizador.
- Funciones principales:
- startTimer:
- Inicia el temporizador.

Utiliza la función `setInterval` para ejecutar una función cada 1 segundo (1000 milisegundos).

La función interna incrementa el valor de timer y actualiza el contenido del elemento con id "timer" mostrando el tiempo transcurrido.

resetGame:

Reinicia el juego.

- Recorre todas las tarjetas (`cards.forEach`) eliminando la clase "flip" que las voltea.
- Vuelve a vincular el evento click a la función `flipCard` para todas las tarjetas, habilitando la interacción.
- Mezcla las tarjetas aleatorizando su orden mediante un ciclo for interno que asigna una posición aleatoria (`Math.floor(Math.random() * 12)`) a la propiedad `style.order` de cada carta.
- Reinicia los valores de counter y timer a 0.
- Actualiza el contenido de los elementos con id "counter" y "timer" mostrando el contador de aciertos y el tiempo.
- Detiene el temporizador anterior si existía (`clearInterval(intervalId)`).
- Inicia un nuevo temporizador llamando a `startTimer`.
- Oculta brevemente todas las cartas volteándolas (`classList.add('flip')`) con un `setTimeout` de 500 milisegundos y luego las desvoltea (`classList.remove('flip')`) con otro `setTimeout` de 3000 milisegundos para mostrarlas temporalmente al usuario.

flipCard:

- Maneja el evento click de cada tarjeta.
- Comprueba si el tablero está bloqueado (`lockBoard`) para evitar interacciones mientras se verifica una pareja.
- Evita que se pueda hacer clic en la misma carta dos veces.
- Agrega la clase "flip" a la carta volteándola.
- Si no se ha volteado ninguna carta (`!hasFlippedCard`), establece la bandera `hasFlippedCard` a true y almacena la referencia a la carta en `firstCard`.

- Si ya hay una carta volteada (hasFlippedCard), almacena la referencia a la segunda carta en secondCard y llama a la función checkForMatch para verificar si son pareja.

checkForMatch:

- Comprueba si las dos últimas cartas volteadas (firstCard y secondCard) pertenecen al mismo grupo (dataset.group).
- Si son pareja (isMatch), llama a la función disableCards para deshabilitar las cartas y aumentar el contador.
- Si no son pareja (!isMatch), llama a la función unflipCards para voltearlas nuevamente.

disableCards:

- Elimina el evento click de las dos últimas cartas para evitar voltearlas nuevamente.
- Llama a la función resetBoard para reiniciar variables internas.
- Aumenta el contador de aciertos (counter) y actualiza el contenido del elemento con id "counter".
- Comprueba si se han encontrado todas las parejas (contador igual a la mitad del número de tarjetas).
- Si se encontraron todas las parejas, muestra una alerta de victoria con un setTimeout de 500 milisegundos.

unflipCards:

- Bloquea el tablero (lockBoard = true) para evitar interacciones mientras se voltean las cartas que no son pareja.
- Utiliza un setTimeout de 1500 milisegundos para voltear nuevamente las dos últimas cartas ('classList'

```

7   // Indica si el tablero esta bloqueado
8   let lockBoard = false;
9   // Guarda la primera tarjeta volteada
10  let firstCard = null;
11  // Guarda la segunda tarjeta volteada
12  let secondCard = null;
13  // Contador de aciertos
14  let counter = 0;
15  // Temporizador
16  let timer = 0;
17  // Identificador del intervalo del temporizador
18  let intervalId = null;
19
20
21  // Función para iniciar el temporizador
22  function startTimer() {
23    intervalId = setInterval(function() {
24      timer++;
25      document.getElementById('timer').innerText = "Tiempo: " + timer;
26    }, 1000);
27  }
28
29  // Función para reiniciar el juego
30  function resetGame() {
31    // Quita la clase 'flip' de todas las tarjetas
32    cards.forEach(function(card) {
33      card.classList.remove('flip');
34      // Agrega el evento 'click' a todas las tarjetas
35      card.addEventListener('click', flipCard);
36    });
37    // Baraja las tarjetas
38    shuffle();
39    // Reinicia el contador y el temporizador
40    counter = 0;
41    timer = 0;
42    // Actualiza el texto del contador y el temporizador
43    document.getElementById('counter').innerText = "Cartas acertadas: " + counter;
44    document.getElementById('timer').innerText = "Tiempo: " + timer;
45    // Si el intervalo del temporizador existe, lo detiene
46    if (intervalId) {
47      clearInterval(intervalId);
48    }
49    // Inicia el temporizador
50    startTimer();
51
52    // Después de 500ms, volteá todas las tarjetas
53    setTimeout(function() {

```

PRACTICA 13 TARJETAS DE VOLTEO

La práctica trata de hacer un programa más dinámico, es decir trata de agregar tres imágenes con la finalidad de que se volteen la imágenes pasando el cursor, para comenzar va a ocupar el lenguaje de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, videos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Esta declaración indica que el documento es de tipo HTML5.
- ✓ <HTML lang="en">: Inicia el elemento raíz del documento HTML y establece el idioma de la página como inglés ("en").
- ✓ <head>: Esta sección contiene metadatos y enlaces a recursos externos.
- ✓ <meta charset="UTF-8">: Especifica el conjunto de caracteres utilizado en el documento como UTF-8.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Define la configuración de la vista para dispositivos móviles.
- ✓ <title>imagenes cool 01</title>: Establece el título de la página que se mostrará en la pestaña del navegador.
- ✓ <link rel="icon" href="icono/images.png">: Enlaza un favicon para la página.
- ✓ <body>: Contiene todo el contenido visible de la página.
- ✓ : Inserta una imagen con un mapa de imagen llamado "rectangle". Si la imagen no se puede cargar, se mostrará el texto alternativo "imagen no encontrada".
- ✓ <map name="rectangle">: Define el mapa de imagen llamado "rectangle" que contiene áreas activas vinculadas a diferentes páginas web.
- ✓ <area shape="rect" coords="34,44,270,350" href="pajinas/uno_index.html" name="rectangle"/>: Define un área rectangular con coordenadas y un enlace a "pajinas/uno_index.html".
- ✓ <area shape="circle" coords="337,300,44" href="pajinas/dos_index.html"/>: Define un área circular con coordenadas y un enlace a "pajinas/dos_index.html".
- ✓ <area shape="rect2" coords="333,280,290,172" href="pajinas/tres_index.html" name="rectangle"/>: Define un área rectangular con coordenadas y un enlace a "pajinas/tres_index.html".
- ✓ : Inserta otra imagen con un mapa de imagen llamado "triangulo".
- ✓ <map name="triangulo">: Define el mapa de imagen llamado "triangulo" que contiene áreas activas vinculadas a diferentes páginas web.
- ✓ Hay múltiples <area> dentro del mapa, cada uno define un área poligonal con coordenadas y un enlace a "pajinas/cuatro_index.html".

```
1 <!--Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
2 <!DOCTYPE html>
3 <!-- La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma-->
4 <html lang="en">
5 <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción-->
6 <head>
7   <!-- Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente-->
8   <meta charset="UTF-8">
9   <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos-->
10  <meta name="viewport" content="width=device-width, initial-scale=1.0">
11  <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda-->
12  <title>PAGINAS RESPONSIVAS</title>
13  <!--Link de css-->
14  <link rel="stylesheet" href="style.css">
15 </head>
16 <!--Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc-->
17 <body>
```

Ilustración 52 ESTRUCTURA BASICA DE HTML

Las tarjetas se diseñan con CSS y se pueden voltear usando JavaScript. Este código HTML crea una página web con un grid de elementos de tarjetas volteables. Cada tarjeta tiene dos caras: un frente con una imagen de fondo y un reverso con un texto poético. Al hacer clic en una tarjeta, esta rota para mostrar el lado opuesto. HTML proporciona la estructura básica para crear un efecto de tarjetas volteables con imágenes y texto. Sin embargo, se requiere JavaScript adicional para implementar la funcionalidad interactiva de voltear las tarjetas.

1. Estructura general (líneas 1-7):

- ✓ La página web comienza con la etiqueta <!DOCTYPE HTML>, que indica que se trata de un documento HTML5.
- ✓ La etiqueta <HTML> define el elemento raíz del documento HTML.
- ✓ La etiqueta <head> contiene metadatos de la página, como la codificación de caracteres (<meta charset="UTF-8">), la configuración del visor (<meta name="viewport" content="width=device-width, initial-scale=1.0">) y el título de la página (<title>Efecto Voltear Tarjetas</title>).

2. Estilo de fondo (línea 8):

- ✓ Se establece un color de fondo para el cuerpo de la página utilizando la propiedad background-color.

3. Contenedor principal (línea 10):

- ✓ Se crea un contenedor principal con la clase container para organizar el contenido de la página.

4. Grid de tarjetas (línea 12):

- ✓ Se crea un elemento contenedor con la clase cards-grid para definir el diseño en forma de cuadrícula de las tarjetas volteables.

5. Tarjetas volteables (líneas 14-38):

- ✓ Se crea un elemento individual con la clase flip-card para representar cada tarjeta volteable dentro del grid.
- ✓ Contenedor interno (líneas 15-16):
- ✓ Se crea un elemento contenedor con la clase flip-card-inner para sostener los lados frontal y posterior de la tarjeta.
- ✓ Lado frontal (líneas 17-18):
- ✓ Se crea un elemento contenedor con la clase flip-card-front para representar el frente de la tarjeta.
- ✓ Se establece una imagen de fondo para el frente de la tarjeta utilizando la propiedad background-image y la ruta a la imagen correspondiente.
- ✓ Lado posterior (líneas 19-22):
- ✓ Se crea un elemento contenedor con la clase flip-card-back para representar el reverso de la tarjeta.
- ✓ Se agrega el texto poético que se mostrará en el reverso de la tarjeta.

6. Repetición de las tarjetas (líneas 24-38):

- ✓ Se repite el código de las líneas 14-22 para crear dos tarjetas volteables adicionales con diferentes imágenes y textos poéticos.

1. Limitaciones del código proporcionado:

- ✓ El código actual no incluye la funcionalidad JavaScript para manejar la animación de volteo de las tarjetas. Se necesitaría un código JavaScript adicional para detectar clics en las tarjetas y aplicar las transformaciones CSS para rotarlas.

```
<body>
    <!--Establece un color de fondo para toda la página web (lila claro)-->
    <body style="background-color: #rgb(213, 100, 179);">
        <!--Elemento de encabezado para el título de la página web-->
        <h1>
            <!--El texto del encabezado con un elemento span para darle un estilo diferente a la palabra "Voltear tarjas".-->
            Efecto: <span>Voltear tarjas</span>
        </h1>
        <!--Un elemento contenedor con la clase "cards-grid" para contener las tarjetas volteables.-->
        <div class="cards-grid">
            <!--Cada elemento .flip-card representa una sola tarjeta volteable.
            Dentro de cada .flip-card-->
            <div class="flip-card">
                <!--Un contenedor interno para sostener los lados frontal y posterior de la tarjeta.-->
                <div class="flip-card-inner">
                    <!--El lado frontal de la tarjeta, utilizando una imagen de fondo de un archivo llamado imágenes/7db4d7d8-c29a-4ded-84b5-a52ee6ab3fa8.jpg'-->
                    <div class="flip-card-front" style="background-image: url('imágenes/7db4d7d8-c29a-4ded-84b5-a52ee6ab3fa8.jpg');">
                    </div>
                    <!--El lado posterior de la tarjeta, que contiene texto.Notas Adicionales:-->
                    <div class="flip-card-back">
                        Habré de levantar la vasta vida
                        que aún ahora es tu espejo:
                        cada mañana habré de reconstruirla.
                        Desde que te alejaste,
                        cuántos lugares se han tornado vanos
                        y sin sentido, iguales
                        a luces en el día
                    </div>
                </div>
            </div>
        </div>
```

Ilustración 53 contenido de html la estructura de las tarjetas

Este fragmento de código JavaScript agrega una funcionalidad interactiva a las tarjetas volteables creadas con HTML, permitiendo que se volteen al hacer clic en ellas. Este código JavaScript implementa la interacción necesaria para que las tarjetas volteables funcionen correctamente, respondiendo a los clics del usuario y aplicando las transformaciones CSS para mostrar el lado opuesto de la tarjeta.

1. Selección de la tarjeta (línea 1):

- ✓ Se utiliza la función `document.querySelector()` para seleccionar el primer elemento que tenga la clase "card_inner".
- ✓ El elemento seleccionado se almacena en la variable `card`.

2. Event listener para el clic (líneas 3-6):

- ✓ Se agrega un evento "click" a la variable `card` utilizando la función `addEventListener()`.
- ✓ Cuando se hace clic en la tarjeta, se ejecuta la función anónima definida dentro del paréntesis.

3. Funcionamiento del clic (líneas 4-5):

- ✓ Dentro de la función anónima, se utiliza el método `classList.toggle()` para manipular las clases del elemento `card`.
- ✓ El método `classList.toggle()` recibe un argumento, que en este caso es la cadena "is-flipped".
- ✓ Si la clase "is-flipped" ya está presente en el elemento, se elimina.
- ✓ Si la clase "is-flipped" no está presente en el elemento, se agrega.

4. Efecto visual (línea 5):

- ✓ La clase "is-flipped" se utiliza en CSS para definir el estilo del lado volteado de la tarjeta.
- ✓ Al alternar la presencia de esta clase, se controla la visibilidad del lado frontal y posterior de la tarjeta, creando el efecto de volteo.

```
JS main.js > ...
1 // Selecciona el primer elemento que tenga la clase 'card_inner' y lo almacena en la variable 'card'
2 const card = document.querySelector('.card_inner');
3
4 // Agrega un event listener que se activa cuando se hace clic en 'card'
5 card.addEventListener('click', function() {
6     // Cuando se hace clic en 'card', se ejecuta esta función anónima
7     // Toggle es un método que agrega una clase si no está presente y la quita si ya está presente
8     card.classList.toggle('is-flipped');
9     // Con esto, se alterna entre agregar y quitar la clase 'is-flipped' en 'card'
10 });
11 |
```

Ilustración 54 código de JavaScript

Este código CSS define los estilos para una página web que presenta un grid de tarjetas volteables con imágenes y texto. Se aplican estilos generales para la página, encabezados, párrafos, tarjetas volteables y sus lados frontal y posterior.

1. Importación de fuentes (líneas 1-2):

- ✓ Se importan las fuentes "Kurale" y "Staatliches" de Google Fonts utilizando la directiva `@import`.
- ✓ Esto permite utilizar estas fuentes personalizadas en los estilos posteriores.

2. Configuración general (líneas 4-10):

- ✓ Se establece el box-sizing en border-box para todos los elementos y sus pseudoelementos, lo que asegura que el ancho y alto de los elementos incluyan sus bordes y rellenos.
- ✓ Se definen estilos base para el elemento HTML y el cuerpo (body), incluyendo el tamaño de fuente predeterminado (16px), márgenes y rellenos a 0.

3. Estilos adicionales para HTML y body (líneas 12-14):

- ✓ Se establecen propiedades display: block para HTML y body, asegurando que se comporten como elementos de bloque.

4. Estilos principales para el cuerpo (líneas 16-22):

- ✓ Se define una altura de línea (line-height) de 1.317101995 para el cuerpo.
- ✓ Se establece un comportamiento de desplazamiento suave (scroll-behavior: smooth) para mejorar la experiencia de usuario.
- ✓ Se define una familia de fuentes predeterminada (font-family) como sans-serif.
- ✓ Se establece un color de fondo blanco (background-color) para el cuerpo.

5. Eliminación del resultado de enfoque en botones (línea 24):

- ✓ Se eliminan los bordes y rellenos asociados al estado de enfoque (focus) en los botones.

6. Estilos para elementos multimedia (líneas 26-32):

- ✓ Se asegura que los elementos multimedia (imágenes, videos, etc.) se ajusten dentro de su contenedor sin sobrepasar sus dimensiones.
- ✓ Se establece una altura automática (height: auto) para estos elementos.
- ✓ Se alinean verticalmente (vertical-align: middle) dentro de su contenedor.

7. Estilos para la línea horizontal (línea 34):

- ✓ Se elimina la altura predeterminada de la línea horizontal (hr).
- ✓ Se establece un borde inferior de 1px sólido con color gris claro (#dfdfdf).
- ✓ Se eliminan los márgenes superior e inferior de la línea horizontal.

8. Estilos para el cuerpo principal (líneas 36-45):

- ✓ Se establece un color de texto principal (color: #5c7470) para el cuerpo.
- ✓ Se define un ancho y altura del 100% para que el cuerpo ocupe toda la ventana gráfica.
- ✓ Se utiliza un diseño flexbox (display: flex) para alinear los elementos internamente.
- ✓ Se establece una dirección de flexbox vertical (flex-direction: column) para apilar los elementos uno sobre otro.
- ✓ Se alinean los elementos horizontalmente al centro (align-items: center).
- ✓ Se agrega un relleno superior e inferior de 32px.

9. Estilos para el título h1 (líneas 47-55):

- ✓ Se define un tamaño de fuente de 4rem para el título.
- ✓ Se elimina el margen superior (margin-top: 0).
- ✓ Se establece un margen inferior de 80px (margin-bottom: 80px).
- ✓ Se define un color de texto negro (color: black).
- ✓ Se utiliza la fuente personalizada "Staatliches" (font-family: "Staatliches", sans-serif).
- ✓ Se establece un peso de fuente normal (font-weight: 400).
- ✓ Se define un estilo de fuente normal (font-style: normal).

10. Estilos para el span dentro de h1 (línea 53):

- ✓ Se aplican los mismos estilos de color de texto, fuente y peso de fuente que en el título h1 al span dentro del mismo.

11. Estilos para los párrafos (línea 57):

- ✓ Se eliminan los márgenes de los párrafos.
- ✓ Se utiliza la fuente personalizada "Staatliches" (font-family: "Staatliches", sans-serif).
- ✓ Se establece un peso de fuente normal (`font-weight: 400`)

```
/* Importa las fuentes 'Kurale' y 'Staatliches' desde Google Fonts */
@import url('https://fonts.googleapis.com/css2?family=Kurale&display=swap');
@import url('https://fonts.googleapis.com/css2?family=Kurale&family=Staatliches&display=swap');

/* Establece el box-sizing en border-box para todos los elementos y sus pseudoelementos */
*, *:before, *::after {
    box-sizing: border-box;
}

/* Estilos base para el elemento root (html) y el cuerpo (body) */
html, body {
    font-size: 16px; /* Tamaño base de la fuente */
    margin: 0; /* Sin márgenes */
    padding: 0; /* Sin relleno */
}

/* Estilos adicionales para html y body */
html, body {
    display: block; /* Asegura que html y body se comporten como bloques */
}

/* Estilos principales para el cuerpo */
body {
    line-height: 1.317101995; /* Altura de línea */
    scroll-behavior: smooth; /* Comportamiento de desplazamiento suave */
    font-family: sans-serif; /* Familia de fuentes predeterminada */
    background-color: #ffffff; /* Color de fondo */
}

/* Elimina el resultado de enfoque en los botones */
button:focus {
    outline: 0;
}

/* Estilos para elementos multimedia para que se ajusten dentro de su contenedor */
img, embed, svg, audio, canvas, iframe, video {
    max-width: 100%;
    height: auto;
    vertical-align: middle;
}

/* Estilos para la línea horizontal */
hr {
    height: 0;
    border: 0;
    border-bottom: 1px solid #dfdfdf;
    margin: 0;
}
```

Ilustración 55 ESTILOS DE LA PAGINA WEB CODIGO DE CSS

```

        color: #5c7470; /* Color de texto principal */
        width: 100%; /* Ancho completo */
        height: 100vh; /* Altura de la ventana gráfica */
        display: flex; /* Flexbox para alinear elementos */
        flex-direction: column; /* Dirección de flexión */
        align-items: center; /* Alinea los elementos al centro horizontal */
        padding: 32px 0; /* Relleno superior e inferior */
    }

    /* Estilos para el título h1 */
    h1 {
        font-size: 4rem; /* Tamaño del título */
        margin-top: 0; /* Sin margen superior */
        margin-bottom: 80px; /* Margen inferior */
        color: black; /* Color del texto */
        font-family: "Staatliches", sans-serif; /* Fuente personalizada */
        font-weight: 400; /* Peso de la fuente */
        font-style: normal; /* Estilo de la fuente */
    }

    /* Estilos para el span dentro de h1 */
    h1 span {
        color: black; /* Color del texto */
        font-family: "Staatliches", sans-serif; /* Fuente personalizada */
        font-weight: 400; /* Peso de la fuente */
        font-style: normal; /* Estilo de la fuente */
    }

    /* Estilos para los párrafos */
    p {
        margin: 0; /* Sin márgenes */
        font-family: "Staatliches", sans-serif; /* Fuente personalizada */
        font-weight: 400; /* Peso de la fuente */
        font-style: normal; /* Estilo de la fuente */
    }

    /* Estilos para el contenedor de las tarjetas */
    .cards-grid {
        display: flex; /* Flexbox para los elementos internos */
        grid-gap: 80px; /* Espacio entre tarjetas */
        padding-bottom: 60px; /* Relleno inferior */
    }

    /* Estilos para las tarjetas de volteo */
    .flip-card {
        width: 320px; /* Ancho de la tarjeta */
        height: 504px; /* Altura de la tarjeta */
        perspective: 2000px; /* Perspectiva para el efecto de volteo */
    }

```

Ilustración 56ESTILOS DE LA PAGINA WEB CODIGO DE CSS

```

1    padding-bottom: 60px; /* Relleno inferior */
2  }
3
4  /* Estilos para las tarjetas de volteo */
5  .flip-card {
6    width: 320px; /* Ancho de la tarjeta */
7    height: 504px; /* Altura de la tarjeta */
8    perspective: 2000px; /* Perspectiva para el efecto de volteo */
9  }
10
11 /* Estilos para el contenedor interno de las tarjetas de volteo */
12 .flip-card-inner {
13   position: relative; /* Posición relativa para los hijos absolutos */
14   width: 100%; /* Ancho completo */
15   height: 100%; /* Altura completa */
16   box-shadow: 0 4px 20px □rgba(0, 0, 0, 0.2); /* Sombra de la tarjeta */
17   border-radius: 28px; /* Radio de borde */
18   transition: all 550ms cubic-bezier(0.1, 0.22, 0.8, 1.13); /* Transición */
19   transform-style: preserve-3d; /* Estilo de transformación */
20 }
21
22 /* Estilos para el frente y el reverso de las tarjetas */
23 .flip-card-front, .flip-card-back {
24   position: absolute; /* Posición absoluta para superponer */
25   width: 100%; /* Ancho completo */
26   height: 100%; /* Altura completa */
27   border-radius: 28px; /* Radio de borde */
28   backface-visibility: hidden; /* Oculta el reverso */
29 }
30
31 /* Estilos para el frente de las tarjetas */
32 .flip-card-front {
33   background-position: center; /* Posición del fondo */
34   background-repeat: no-repeat; /* No repetir el fondo */
35   background-size: cover; /* Cubrir el fondo */
36 }
37
38 /* Estilos para el reverso de las tarjetas */
39 .flip-card-back {
40   font-size: 1.25rem; /* Tamaño de fuente */
41   text-align: center; /* Alineación de texto */
42   display: grid; /* Grid layout */
43   place-items: center; /* Centra los elementos en el grid */
44   padding: 32px; /* Relleno */
45   background-color: ■#81CFC8; /* Color de fondo */
46   transform: rotateY(180deg); /* Gira 180 grados para ocultar el reverso */
47 }
48
49

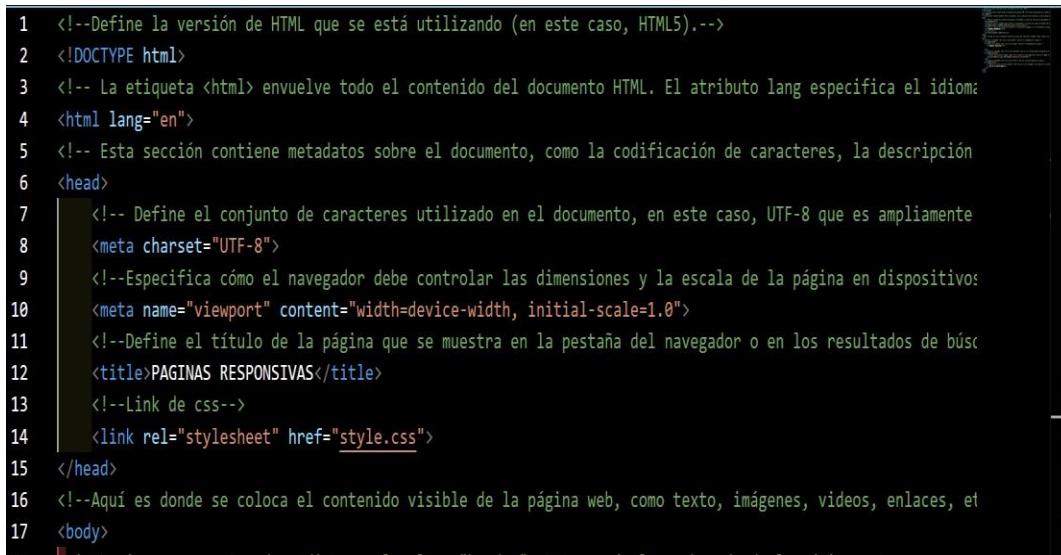
```

Ilustración 57 ESTILOS DE LA PAGINA WEB CODIGO DE CSS

PRACTICA 14: KIMBY

Practica 14 se realizara un diseño de un kirby elaborado de HTML son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, imágenes, textos, videos y todo tipo de material multimedia de forma que se pueda visualizar correctamente, para la estructura básica de HTML se agregara como primer punto :

- ✓ <!DOCTYPE HTML>: Esta es una declaración que especifica que el documento es de tipo HTML5.
- ✓ <HTML lang="en">: Inicia el elemento raíz del documento HTML y especifica el idioma de la página como inglés ("en").
- ✓ <head>: Dentro de este elemento se colocan metadatos, como el conjunto de caracteres, la configuración de la vista y enlaces a hojas de estilo y scripts.
- ✓ <meta charset="UTF-8">: Define el juego de caracteres del documento como UTF-8, que incluye la mayoría de los caracteres utilizados en la mayoría de los idiomas escritos.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Este metaetiqueta establece la escala inicial y el ancho de la ventana gráfica cuando se carga la página en dispositivos móviles.
- ✓ <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">: Este enlace carga el archivo CSS de Bootstrap desde un CDN (Content Delivery Network), que proporciona estilos predefinidos para la maquetación y los componentes.
- ✓ <title>Document</title>: Define el título de la página que se mostrará en la barra de título del navegador.
- ✓ <link rel="stylesheet" href="style.css">: Este enlace carga tu archivo CSS personalizado llamado "style.css", que puede contener estilos adicionales específicos para tu página.
- ✓ <body>: Inicia el cuerpo del documento HTML, que contiene todo el contenido visible de la página.



```
1 <!--Define la versión de HTML que se está utilizando (en este caso, HTML5).-->
2 <!DOCTYPE html>
3 <!-- La etiqueta <html> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma-->
4 <html lang="en">
5 <!-- Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción-->
6 <head>
7   <!-- Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente-->
8   <meta charset="UTF-8">
9   <!--Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos-->
10  <meta name="viewport" content="width=device-width, initial-scale=1.0">
11  <!--Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda-->
12  <title>PAGINAS RESPONSIVAS</title>
13  <!--Link de css-->
14  <link rel="stylesheet" href="style.css">
15 </head>
16 <!--Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc-->
17 <body>
```

Ilustración 58Estructura básica de Html5

Este código HTML representa la estructura básica de una página web con un diseño de dos columnas. La página contiene principalmente elementos circulares dispuestos de forma jerárquica.

En resumen, este código HTML proporciona una estructura básica con elementos circulares. Sin embargo, se requiere la implementación de estilos CSS para darles una apariencia visual y un posicionamiento adecuado.

Contenedor principal (`<div class="container">`):

- ✓ Define el contenedor principal que engloba el resto del contenido de la página.

Fila principal (`<div class="row">`):

- ✓ Dentro del contenedor principal, se crea una fila que abarca el ancho completo de la página.

Columnas (`<div class="col">`):

- ✓ La fila principal se divide en dos columnas con la clase col.
- ✓ En teoría, estas columnas deberían ocupar el 50% del ancho de la página cada una (asumiendo un sistema de grid de Bootstrap u otro framework similar).

Elementos circulares:

- ✓ Cada columna contiene varios elementos div con clases que parecen representar círculos de diferentes tamaños y posiciones.
- ✓ Se utilizan atributos de estilo inline (style) para definir propiedades como margin-top y float para posicionar los círculos relativamente dentro de las columnas.

Limitaciones del código proporcionado:

- ✓ El código carece de estilos CSS para definir la apariencia visual de los círculos (color, tamaño, fondo, etc.).
- ✓ No hay referencias a clases CSS que se puedan utilizar para aplicar estilos consistentes a los círculos.
- ✓ El código utiliza float para posicionar los círculos, lo cual puede generar problemas de responsividad en pantallas más pequeñas.

```
8 <div class="container">
9   <!-- Fila principal -->
10  <div class="row">
11    <!-- Columna -->
12    <div class="col">
13      <!-- Título principal -->
14      <h1 style="margin-top: 250px; text-align: center;"> hola</h1>
15      <!-- Primer círculo secundario -->
16      <div class="circulo09" style="margin-top: 80px; float: right;"></div>
17    </div>
18    <!-- Columna -->
19    <div class="col">
20      <!-- Segundo círculo principal -->
21      <div class="circulo" style="margin-top: 80px;">
22        <!-- Tres círculos secundarios -->
23        <div class="circulo7" style="margin-top: 100px; float: right;"></div>
24        <div class="circulo6" style="margin-top: 450px; float: right;"></div>
25        <div class="circulo8" style="margin-top: 100px; float: left;"></div>
26        <!-- Círculo interno con dos círculos -->
27        <div class="circulo2" style="margin-top: 40px;">
28          <div class="circulo22">
29            <div class="circulo21"></div>
30            <div class="circulo23"></div>
31          </div>
32        </div>
33        <!-- Otros dos círculos secundarios -->
34        <div class="circulo3" style="margin-top: -280PX; float: right;"></div>
35        <div class="circulo4" style="margin-top: 40px;">
36          <div class="circulo41" style="margin-top: 50px;"></div>
37        </div>
38      </div>
39    </div>
40  </div>
41</div>
```

Ilustración 59 CONTENIDO DE HTML PARA LA ESTRUCTURA DEL KIRBY

Este código CSS define estilos para varios elementos circulares que se presentan en una página web. Los estilos se aplican a clases CSS específicas como .circulo, .circulo2, .circulo3, etc., para controlar la apariencia de cada elemento circular. Define estilos para varios elementos circulares en una página web.

1. Estilos generales para círculos (circulo):

- ✓ Se establece un ancho y altura de 500px para todos los elementos con la clase .circulo.
- ✓ Se aplica un radio de borde de 50%, lo que crea un efecto de borde redondeado.
- ✓ Se establece un color de fondo que se ve rojo, pero falta el valor hexadecimal exacto.
- ✓ Se define un estilo de borde sólido.
- ✓ Se agrega un margen inferior de 10px.
- ✓ Se establece display: block para que los elementos se comporten como bloques.
- ✓ Se aplican valores de margen izquierdo y derecho de auto, lo que centra los elementos horizontalmente dentro de su contenedor padre.

2. Estilos para círculos más pequeños (circulo2, circulo3, circulo4, circulo5):

- ✓ Se aplican estilos similares a la clase .circulo, pero con diferentes valores de ancho, altura, color de fondo y margen inferior.
- ✓ El radio de borde varía entre 50% y 80% para crear círculos de diferentes tamaños.
- ✓ Los colores de fondo también varían, utilizando valores RGB específicos.

3. Estilos para círculos más grandes y anidados (circulo6, circulo7, circulo8):

- ✓ Se aplican estilos similares a la clase .circulo, pero con valores de ancho, altura y margen inferior más grandes.
- ✓ El radio de borde se mantiene en 50%.
- ✓ Los colores de fondo varían, utilizando valores RGB específicos.

4. Estilos para círculos pequeños dentro de un círculo más grande (circulo21, circulo22):

- ✓ Se aplican estilos similares a la clase .circulo, pero con valores de ancho, altura y margen inferior aún más pequeños.
- ✓ El radio de borde varía entre 50% y 60%.
- ✓ Los colores de fondo varían, utilizando valores RGB específicos.

5. Estilo para un círculo pequeño dentro de otro círculo más pequeño (circulo51):

- ✓ Se aplican estilos similares a la clase .circulo, con un ancho y altura de 30px y 50px respectivamente.
- ✓ El radio de borde es de 60%.
- ✓ Se establece un color de fondo que se ve blanco, pero falta el valor hexadecimal exacto.

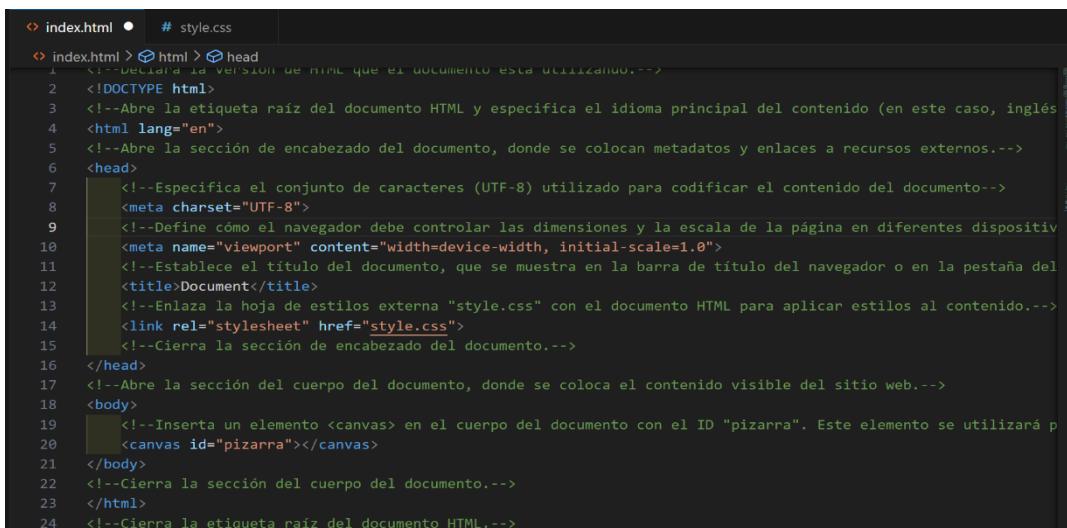
6. Estilo para un círculo pequeño dentro de otro círculo (circulo41):

- ✓ Se aplican estilos similares a la clase .circulo, con un ancho y altura de 40px y 50px respectivamente.
- ✓ El radio de borde es de 70%.
- ✓ Se establece un color de fondo que se ve rojo, pero falta el valor hexadecimal exacto.

PRÁCTICA 15: JEUGO DE AVION

Practica 15: consiste en la aplicación de un juego con una nave, que su funcionamiento el cual es que al dar clic al mouse empezara a disparar y matar a los enemigos, para esto llevaremos a cabo empezando con el lenguaje estructurado de HTML en el cual son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, el elemento de <Canvas> Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <HTML> envuelve todo el contenido del documento HTML. El atributo langa especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.



The screenshot shows a code editor with two tabs: 'index.html' and '# style.css'. The 'index.html' tab is active and displays the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <canvas id="pizarra"></canvas>
  </body>
</html>
```

The '# style.css' tab is visible but contains no code. To the right of the code editor, there is a small preview window showing a dark gray canvas area, which is likely a placeholder or a preview of the game's graphics.

Este código HTML define la interfaz gráfica de un juego espacial, presumiblemente utilizando JavaScript y otros recursos para la lógica y el funcionamiento del juego. este código HTML configura los elementos visuales para mostrar las vidas, el tiempo y la nave espacial del jugador, además de incluir los elementos de audio para los sonidos del juego. El archivo JavaScript principal "app.js" se encarga de la lógica y la interacción del juego.

Elementos principales:

span#lives:

- Muestra el número de vidas restantes del jugador.
- El texto "Vidas:" se muestra antes del número de vidas.
- El número de vidas inicial es 5, representado por la etiqueta <i>5</i>.

span#times:

- Muestra el tiempo restante del jugador.
- Inicialmente no se muestra texto ni información en este elemento.

div.nave#nave:

- Representa la nave espacial del jugador.
- Se utiliza la clase "nave" para estilizar la apariencia de la nave.
- Se le asigna un identificador "nave" para poder manipularla con JavaScript.

audio#laser:

- Define un elemento de audio para reproducir el sonido del láser cuando la nave dispara.
- La fuente de audio es el archivo "audio/laser.mp3".

audio#explosion:

- Define un elemento de audio para reproducir el sonido de la explosión cuando la nave es destruida.
- La fuente de audio es el archivo "audio/explosion.mp3".

script src="js/app.js":

- Carga el archivo JavaScript principal "app.js" que presumiblemente contiene la lógica del juego, el control de la nave, la gestión del tiempo y las vidas, y la reproducción de los sonidos.

Consideraciones adicionales:

- El código HTML proporcionado solo muestra los elementos básicos de la interfaz gráfica del juego.
- Se requiere el código JavaScript "app.js" para entender la funcionalidad completa del juego.
- Es probable que existan otros elementos HTML y CSS para definir el resto de la interfaz del juego, como el fondo espacial, enemigos, disparos, etc.

```

// Inicializa el juego de memoria
function initBlockScript() {
    // Selecciona todas las tarjetas de memoria
    const cards = document.querySelectorAll('.memory-card');
    // Indica si se ha volteado una tarjeta
    let hasFlippedCard = false;
    // Indica si el tablero está bloqueado
    let lockBoard = false;
    // Guarda la primera tarjeta volteada
    let firstCard = null;
    // Guarda la segunda tarjeta volteada
    let secondCard = null;
    // Contador de aciertos
    let counter = 0;
    // Temporizador
    let timer = 0;
    // Identificador del intervalo del temporizador
    let intervalId = null;

    // Función para iniciar el temporizador
    function startTimer() {
        intervalId = setInterval(function() {
            timer++;
            document.getElementById('timer').innerText = "Tiempo: " + timer;
        }, 1000);
    }

    // Función para reiniciar el juego
    function resetGame() {
        // Quita la clase 'flip' de todas las tarjetas
        cards.forEach(function(card) {
            card.classList.remove('flip');
            // Agrega el evento 'click' a todas las tarjetas
            card.addEventListener('click', flipCard);
        });
        // Baraja las tarjetas
        shuffle();
        // Reinicia el contador y el temporizador
        counter = 0;
        timer = 0;
        // Actualiza el texto del contador y el temporizador
        document.getElementById('counter').innerText = "Cartas acertadas: " + counter;
        document.getElementById('timer').innerText = "Tiempo: " + timer;
        // Si el intervalo del temporizador existe, lo detiene
        if (intervalId) {
            clearInterval(intervalId);
        }
        // Inicia el temporizador
        ...
    }
}

```

Ilustración 60 Código de java en el juego

Este código JavaScript se encarga de generar enemigos (meteoritos) y verificar las colisiones con la nave espacial del jugador en un juego espacial.

Explicación paso a paso:

- Intervalo para generar meteoritos:
- Se declara una variable aparecer inicializada en 0.
- Se utiliza la función setInterval para ejecutar un bloque de código cada cierto tiempo (en este caso, cada 100 milisegundos).

Generar enemigos:

- Dentro del intervalo, se incrementa la variable aparecer en 1 en cada ciclo.
- Se usa un condicional if para verificar si el valor de aparecer es divisible entre 10 (aparecer % 10 == 0).
- Si la condición se cumple, significa que han pasado múltiplos de 10 ciclos del intervalo (cada 1 segundo).
- Se crea un nuevo elemento div usando document.createElement('div').
- Se agrega la clase "enemigo" al elemento para aplicar estilos que representen el meteoro.
- Se agrega el elemento div creado (enemigo) como hijo del elemento body usando body.append(enemigo).
- Se establece la posición inicial del enemigo en el eje horizontal (left) utilizando Math.random() * window.innerWidth - 100. Esto genera un número aleatorio entre 0 y el ancho de la ventana (menos 100 pixeles) para posicionar el enemigo dentro del área visible del juego.

Mover y verificar colisiones:

- Se seleccionan todos los elementos con la clase "enemigo" usando querySelectorAll('.enemigo') y se almacenan en la variable enemigos.
- Se recorre la lista de enemigos (enemigos.forEach(element => { ... })) para aplicar lógica a cada enemigo individualmente.
- Se incrementa la posición vertical (top) del enemigo en 10 pixeles sumando el valor actual element.getBoundingClientRect().top para simular el movimiento hacia abajo.
- Se obtiene el rectángulo delimitador del enemigo (element.getBoundingClientRect()) para verificar la posición.
- Se compara la posición superior del enemigo (element.getBoundingClientRect().top) con la posición superior de la nave (nave.getBoundingClientRect().top).
- Si la posición superior del enemigo es mayor a la posición superior de la nave (el enemigo ha sobrepasado la nave), significa que hubo una colisión.
- Se reduce la cantidad de vidas del jugador (lives--).
- Se actualiza el contenido del elemento con id "live" para mostrar la cantidad de vidas restantes (live.textContent = lives).
- Se verifica si las vidas restantes son menores a -1 (lo que probablemente sea un error).
- Si las vidas son menores a -1, se muestra una alerta de "Game Over" con alert('Game Over').
- Se recarga la página del juego con location.reload() para reiniciar el juego.
- Se elimina el elemento del enemigo del DOM utilizando element.remove() para que desaparezca de la pantalla.

```

    }
    // Inicia el temporizador
    startTimer();

    // Después de 500ms, voltear todas las tarjetas
    setTimeout(function() {
        cards.forEach(function(card) {
            card.classList.add('flip');
        });
    }, 500);

    // Después de 3000ms, vuelve a voltear todas las tarjetas
    setTimeout(function() {
        cards.forEach(function(card) {
            card.classList.remove('flip');
        });
    }, 3000);
}

// Agrega el evento 'click' al botón de reinicio
document.getElementById('reset').addEventListener('click', resetGame);

// Función para voltear una tarjeta
function flipCard() {
    // Si el tablero está bloqueado, no hace nada
    if (lockBoard) return;
    // Si esta tarjeta es la primera tarjeta volteada, no hace nada
    if (this === firstCard) return;

    // Agrega la clase 'flip' a esta tarjeta
    this.classList.add('flip');

    // Si no se ha volteado ninguna tarjeta aún
    if (!hasFlippedCard) {
        // Indica que se ha volteado una tarjeta
        hasFlippedCard = true;
        // Guarda esta tarjeta como la primera tarjeta volteada
        firstCard = this;
        // No hace nada más
        return;
    }

    // Guarda esta tarjeta como la segunda tarjeta volteada
    secondCard = this;
    // Comprueba si las dos tarjetas volteadas son iguales
    checkForMatch();
}

```

Ilustración 61 código de JavaScript

CÓDIGO CSS

Este código CSS define los estilos para los elementos del juego espacial que vimos anteriormente en el código HTML.

Estilos generales:

- Se aplica un reset de estilos a todos los elementos (*) con margin:0; padding: 0; box-sizing: border-box; para una base estandarizada.

Estilos para la nave:

.nave:

- Define el ancho (width: 80px;) y alto (height: 80px;) de la nave espacial del jugador.
- Establece una imagen de fondo (background-image: url('../img/nave.png')) para la nave.
- Las propiedades background-repeat: no-repeat; y background-size: cover; controlan cómo se muestra la imagen de fondo.
- Se posiciona la nave de forma absoluta (position: absolute;) permitiendo ubicarla libremente sobre el fondo.
- La propiedad bottom: 50px; la posiciona a 50 pixeles desde el borde inferior de la ventana.
- calc(50vw - 40px) centra la nave horizontalmente restando la mitad del ancho de la nave (40px) al 50% del ancho de la ventana (50vw).

Estilos para el fondo:

- Define una imagen de fondo (background-image: url('../img/espacio.jpg')) para el cuerpo del documento que representará el espacio.
- Las propiedades background-repeat: no-repeat; y background-size: cover; controlan cómo se muestra la imagen de fondo a pantalla completa.
- Se establece el alto (height: 100vh;) y ancho (width: 100vw;) del cuerpo al 100% del alto y ancho de la ventana respectivamente para que la imagen ocupe toda la pantalla.
- overflow: hidden; oculta cualquier elemento que se desborde del cuerpo del documento.

Estilos para los disparos (balas):

.bala:

- Define el ancho (width: 10px;) y alto (height: 50px;) de los disparos (balas) de la nave.
- Establece un color de fondo (background-color:rgb(234,243,150);) para las balas.
- Aplica un borde redondeado (border-radius: 15px;) a las balas.
- Agrega una sombra (box-shadow: 0px 0px 40px 10px #0ff;) para dar volumen a las balas.
- Se posiciona de forma absoluta (position: absolute;) para permitir ubicar los disparos libremente sobre el resto de elementos.

Estilos para los enemigos (meteoritos):

.enemigo:

- Define el ancho (width: 80px;) y alto (height: 50px;) de los enemigos (meteoritos).
- Establece una imagen de fondo (background-image: url('../img/roca.png')) para los meteoritos.
- Las propiedades background-repeat: no-repeat; y background-size: cover; controlan cómo se muestra la imagen de fondo.
- Se posiciona de forma absoluta (position: absolute;) permitiendo ubicarlos libremente sobre el fondo.
- La propiedad top: 0; los posiciona en la parte superior de la pantalla.

Estilos para las vidas:

#lives:

- Define el estilo para el elemento con id "lives" que muestra las vidas restantes del jugador.
- Se posiciona de forma absoluta (position: absolute;) permitiendo ubicarlo libremente sobre el resto de elementos.
- La propiedad top: 10px; lo ubica a 10 pixeles del borde superior de la ventana.
- left: 10px; lo posiciona a 10 pixeles del borde izquierdo de la ventana.
- Establece un tamaño de fuente grande (font-size: 2em;) para las vidas.
- Define el color de texto (color:rgb(245,222,19);) para las vidas.
- Aplica un estilo específico a la etiqueta <i> que probablemente se usa dentro del elemento #lives para mostrar el número actual de vidas.
- Cambia el color del texto (color: red;) a rojo para resaltar las vidas restantes.

```
1  /* Interacción */
2  .interaction {
3      display: flex;
4  }
5
6  /* Juego de memoria */
7  .memory-game {
8      width: 640px;
9      margin: auto;
10     display: flex;
11     flex-wrap: wrap;
12     perspective: 1000px;
13 }
14
15  /* Tarjeta de memoria */
16  .memory-card {
17      width: calc(25% - 10px);
18      height: 300px;
19      margin: 5px;
20      position: relative;
21      transform: scale(1);
22      transform-style: preserve-3d;
23      transition: transform 0.5s;
24 }
25
26  /* Estilos para el botón de reset */
27  #reset {
28      position: absolute;
29      top: 0;
30      left: 0;
31      background-color: skyblue;
32      color: white;
33      padding: 10px;
34      border-radius: 5px;
35      border: none;
36      cursor: pointer;
37      margin: 10px;
38      text-align: center;
39 }
40
41  /* Estilos para el contador y timer */
42  #counter, #timer {
43      margin-top: -350px;
44      font-size: large;
45 }
46
47  /* Media query para pantalla pequeña */
48  @media (max-width: 767.98px) {
```

Ilustración 62 Código css

PRACTICA FOCO 16

Practica 16: consiste en la aplicación de crear un foco, que su funcionamiento el cual es que al dar clic a un botón de encendido que se encienda el foco y crear otro botón que tenga el funcionamiento de apagado, para esto llevaremos a cabo empezando con el lenguaje estructurado de HTML en el cual son elementos utilizados para definir la estructura jerárquica y organizada el contenido de una página web. Para esta práctica utilizaremos HTML5 ya que nos permite crear la estructura de una página web, por así decirlo, su esqueleto, donde vamos a incorporar posteriormente, el elemento de <Canvas> Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.

- ✓ <!DOCTYPE HTML>: Define la versión de HTML que se está utilizando (en este caso, HTML5).
- ✓ <HTML lang="es">: La etiqueta <HTML> envuelve todo el contenido del documento HTML. El atributo lang especifica el idioma del documento, en este caso, español (es).
- ✓ <head>: Esta sección contiene metadatos sobre el documento, como la codificación de caracteres, la descripción de la página (a través de etiquetas meta), el título de la página y enlaces a hojas de estilo CSS, fuentes y otros recursos.
- ✓ <meta charset="UTF-8">: Define el conjunto de caracteres utilizado en el documento, en este caso, UTF-8 que es ampliamente utilizado y admite una amplia gama de caracteres.
- ✓ <meta name="viewport" content="width=device-width, initial-scale=1.0">: Especifica cómo el navegador debe controlar las dimensiones y la escala de la página en dispositivos móviles. width=device-width indica que el ancho de la página se ajustará al ancho del dispositivo, y initial-scale=1.0 establece el nivel inicial de zoom cuando se carga la página.
- ✓ <title>Título de la página</title>: Define el título de la página que se muestra en la pestaña del navegador o en los resultados de búsqueda.
- ✓ <body>: Aquí es donde se coloca el contenido visible de la página web, como texto, imágenes, videos, enlaces, etc.

```
index.html # style.css
index.html > html > head
1  <!--Declara la versión de HTML que el documento está utilizando.-->
2  <!DOCTYPE html>
3  <!--Abre la etiqueta raíz del documento HTML y especifica el idioma principal del contenido (en este caso, inglés-->
4  <html lang="en">
5  <!--Abre la sección de encabezado del documento, donde se colocan metadatos y enlaces a recursos externos.-->
6  <head>
7      <!--Especifica el conjunto de caracteres (UTF-8) utilizado para codificar el contenido del documento-->
8      <meta charset="UTF-8">
9      <!--Define cómo el navegador debe controlar las dimensiones y la escala de la página en diferentes dispositivos-->
10     <meta name="viewport" content="width=device-width, initial-scale=1.0">
11     <!--Establece el título del documento, que se muestra en la barra de título del navegador o en la pestaña del navegador-->
12     <title>Document</title>
13     <!--Enlaza la hoja de estilos externa "style.css" con el documento HTML para aplicar estilos al contenido.-->
14     <link rel="stylesheet" href="style.css">
15     <!--Cierra la sección de encabezado del documento.-->
16 </head>
17 <!--Abre la sección del cuerpo del documento, donde se coloca el contenido visible del sitio web.-->
18 <body>
19     <!--Inserta un elemento <canvas> en el cuerpo del documento con el ID "pizarra". Este elemento se utilizará para dibujar gráficos, animaciones, o cualquier otro contenido gráfico dinámico utilizando JavaScript.-->
20     <canvas id="pizarra"></canvas>
21 </body>
22 <!--Cierra la sección del cuerpo del documento.-->
23 </html>
```

Ilustración 63 ESTRUCTURA BASICA DE HTML

Continuando con la estructura de HTML comenzaremos agregando un contenedor div con la clase container encierra toda la página. Dentro de este contenedor, la primera fila (div con la clase row) contiene una columna (div con la clase col). La columna contiene un encabezado

con el texto "HOLA MUNDO" centrado y con color blanco.

Continuando con las instrucciones:

- ✓ Otra fila (div con la clase row) contiene una columna (div con la clase col).
 - ✓ La columna contiene un subtítulo

con el texto "CAMBIA ELEMENTOS ALGUNOS ATRIBUTOS" centrado y con color blanco.
 - ✓ Debajo del subtítulo, un párrafo

explica la intención de cambiar atributos con un texto centrado y color blanco.

Espacio vacío:

- ✓ Una fila (div con la clase row) contiene una columna vacía (div con la clase col-sm-4). Esta columna no tiene contenido visible.

Lámpara interactiva:

- ✓ La última fila (div con la clase row) contiene tres columnas:
 - ✓ La primera columna (div con la clase col-sm-2) contiene un botón con la etiqueta "Prender".
 - ✓ El botón tiene las clases btn btn-primary de Bootstrap que le dan estilo visual.
 - ✓ El atributo onclick ejecuta código JavaScript cuando se hace clic en el botón.
 - ✓ La segunda columna (div con la clase col-sm-8) contiene la imagen de una lámpara.
 - ✓ La imagen se carga desde el archivo img/pic_bulbon.gif.
 - ✓ El atributo id="foto" permite acceder a la imagen mediante JavaScript.
 - ✓ Se establecen atributos para el ancho (width) y alto (height) de la imagen.
 - ✓ La tercera columna (div con la clase col-sm-2) contiene otro botón con la etiqueta "Apagar".
 - ✓ El estilo y funcionalidad del botón son similares al primero, pero ejecuta código para apagar la lámpara.

```
<div class="container">
  <div class="row">
    <!-- Columna vacía a la izquierda -->
    <div class="col-sm-4">
      <!-- Columna central con el título y el formulario -->
      <div class="col-sm-4" style="background-position: center center;">
        <!-- Espacio en blanco -->
        <br>
        <br>
        <!-- Título del juego -->
        <h1 style="text-align: center ;"> Adivina el Número</h1>
        <!-- Imagen del juego -->
        
      </div>
      <!-- Columna vacía a la derecha -->
      <div class="col-sm-4">
        <!-- Formulario para adivinar el número -->
        <div class="form">
          <!-- Espacio en blanco -->
          <br>
          <p>ENCUENTRA EL NUMERO DEL 1 AL 10 </p>
          <br>
          <!-- Campo de texto para introducir la respuesta -->
          <label for="guessField" style="text-align: center;">ADIVINA EL NUMERO: </label>
          <br>
          <br>
          <input type="text" id="guessField" class="guessField" />
          <br>
          <br>
          <!-- Botón para enviar la respuesta -->
          <input type="submit" value="Enviar respuesta" class="guessSubmit" />
        </div>
        <!-- Área para mostrar los resultados -->
        <div class="resultPars">
          <p class="guesses"></p>
          <p class="lastResult"></p>
          <p class="lowOrHi"></p>
        </div>
      </div>
    </div>
  </div>
</div>
```

Ilustración 64 Contenido estructurado de HTML para la realización del juego

Código JavaScript:

El código JavaScript se ejecuta al hacer clic en los botones y modifica el atributo src de la imagen con el identificador "foto": y el código HTML y JavaScript crean una página web con un título, instrucciones, y una lámpara interactiva que se puede encender y apagar al hacer clic en los botones correspondientes. Comenzaremos con los botones de encendido y apagado para esto vamos a ser útil de onclick que especificar qué código ejecutar cuando se hace clic en el botón

- ✓ onclick="document.getElementById('foto').src='img/pic_bulbon.gif)": Al hacer clic en el botón "Prender", se obtiene la referencia a la imagen mediante document.getElementById('foto') y se actualiza el atributo src con la ruta de la imagen encendida (img/pic_bulbon.gif).
- ✓ onclick="document.getElementById('foto').src='img/pic_bulboff.gif)": Al hacer clic en el botón "Apagar", se sigue el mismo proceso para actualizar el atributo src con la ruta de la imagen apagada (img/pic_bulboff.gif).

Consideraciones adicionales:

- ✓ Optimización de imágenes: Se recomienda optimizar las imágenes para reducir su tamaño y mejorar la velocidad de carga de la página.
- ✓ Accesibilidad: El texto alternativo (alt) en la etiqueta podría ser útil para describir la imagen en caso de que no se cargue o para usuarios con lectores de pantalla.
- ✓ Mejoras visuales: Se pueden agregar estilos CSS adicionales para personalizar la apariencia de la página y los botones.

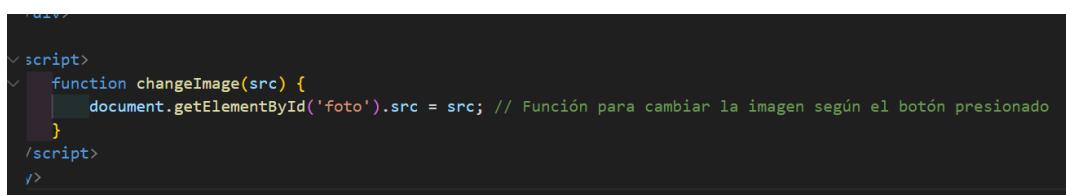


Ilustración 65 CODIGO ESTRUCTURADO DE JAVASCRIPT

En resumen, este código CSS define estilos personalizados para la página web utilizando dos fuentes de Google Fonts: "Pacifico" y "Roboto". El estilo general es minimalista con un fondo negro, mientras que los encabezados y párrafos utilizan la fuente "Pacifico" en cursiva.

importación de fuentes:

- ✓ @import url('https://fonts.googleapis.com/css2?family=Pacifico&display=swap');: Importa la fuente "Pacifico" de Google Fonts.
- ✓ @import url('https://fonts.googleapis.com/css2?family=Pacifico&family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,900;1,100;1,300;1,400;1,500;1,700;1,900&display=intercambio');: Importa la fuente "Roboto" de Google Fonts con diferentes estilos de cursiva y grosor.

Estilo general:

- ✓ body{ background-color:black ; }: Establece el color de fondo del cuerpo de la página en negro.

- ✓ Estilos para encabezados (h2, h3):
- ✓ h2{ font-family: "Pacifico", cursiva; peso de fuente: 400; estilo de fuente: normal; }:
- ✓ Define la fuente "Pacifico" para los encabezados h2.
- ✓ Establece el estilo de fuente en cursiva.
- ✓ El peso de fuente se establece en 400 (normal).
- ✓ El estilo de fuente se establece en normal.
- ✓ h3{ font-family: "Pacifico", cursiva; peso de fuente: 400; estilo de fuente: normal; }:
- ✓ Define la fuente "Pacifico" para los encabezados h3.
- ✓ Establece el estilo de fuente en cursiva.
- ✓ El peso de fuente se establece en 400 (normal).
- ✓ El estilo de fuente se establece en normal.

Estilos para párrafos (p):

- ✓ p{ font-family: "Pacifico", cursiva; peso de fuente: 400; estilo de fuente: normal; }:
- ✓ Define la fuente "Pacifico" para los párrafos.
- ✓ Establece el estilo de fuente en cursiva.
- ✓ El peso de fuente se establece en 400 (normal).
- ✓ El estilo de fuente se establece en normal.

Consideraciones adicionales:

- ✓ **Combinación de fuentes:** La elección de dos fuentes diferentes puede generar algunas consideraciones de legibilidad y armonía visual. Es importante asegurarse de que las fuentes se complementen entre sí y no generen una sensación de desorden o desequilibrio en la página.
- ✓ **Jerarquía visual:** Los estilos para los encabezados (h2, h3) y párrafos (p) deberían crear una clara jerarquía visual en la página, diferenciando los diferentes tipos de contenido.
- ✓ **Personalización:** Este código CSS es un punto de partida. Se pueden ajustar y personalizar los estilos para adaptarlos a los gustos y necesidades específicas del proyecto.

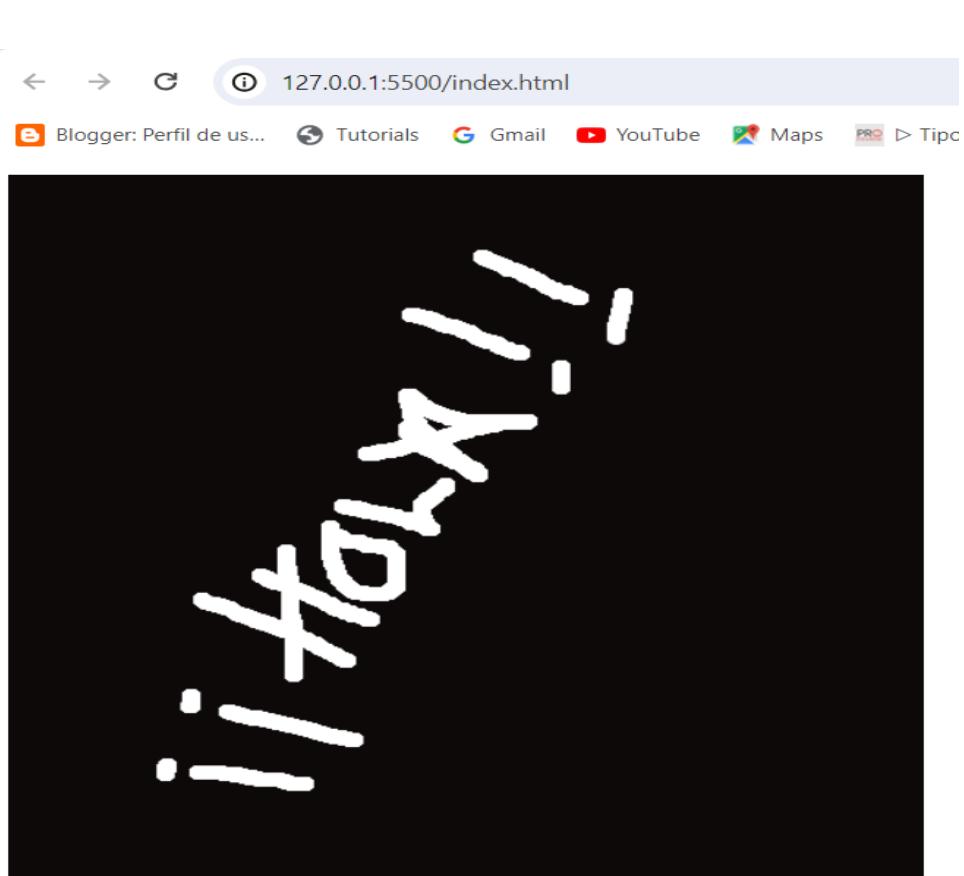
RESULTADOS

PRÁCTICA 1 DE HTML5: PIZARRA

Objetivo: crear una pizarra que al hacer clic dibuje en la pizarra

Descripción de la página web: La página web creada consiste en una estructura básica con elementos HTML5, CSS Y JavaScript

Contenido de la página:



PRÁCTICA 2 DE HTML5: PIZARRA_2

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5, CSS Y JAVASRIPT

Descripción de la página web: Creación de una pizarra que al pasar el mouse pinte en la pizarra.

Contenido de la página:



PRÁCTICA 3 DE HTML5: FORMULARIO EN PDF

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5,CSS Y JAVASCRIPT

Descripción de la página web: La página web creada consiste en un botón que se botón tenga la funcionalidad de convertirse en PDF

Contenido de la página:

The image shows a web page with a light gray background. In the center, there is a white rectangular form with a thin orange border. At the top of the form, the word "FORMULARIO" is written in a large, bold, dark font. Below the title, there are four input fields, each consisting of a label followed by a white horizontal input box. The labels are "Nombre", "Apellido", "Correo Electronico", and "Registro Federal de Contribuyentes". At the bottom of the form, there is a single button labeled "GUARDAR como PDF" in a dark font, enclosed in a white rounded rectangle.



FORMULARIO

Nombre

FATIMA

Apellido

ALCANTARA

Correo Electronico

CVGDBHNJLMKC@JDBC

Registro Federal de Contribuyentes

IYE32

GUARDAR como PDF

Datos Registrados Correctamente:

Nombre:FATIMA

Apellido:ALCANTARA

Direccion de Correo ElectronicoCVGDBHNJLMKC@JDBC

Numero de Registro Federal deContribuyentes:IYE32

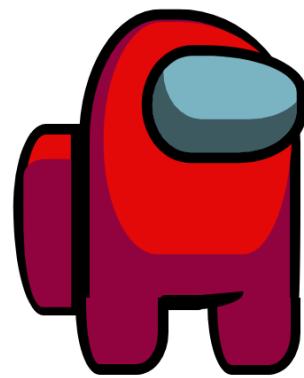
ALCANTARA

PRÁCTICA 4 DE HTML5: AMONGOS EN PDF

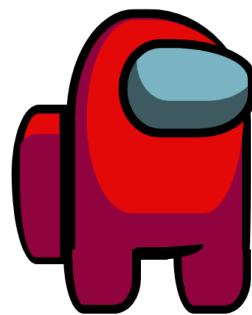
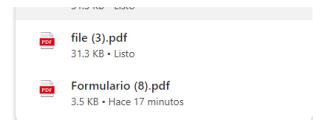
Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5,CSS Y JAVASCRIPT

Descripción de la página web: La página web creada consiste en que la imagen de amongos se cree en PDF

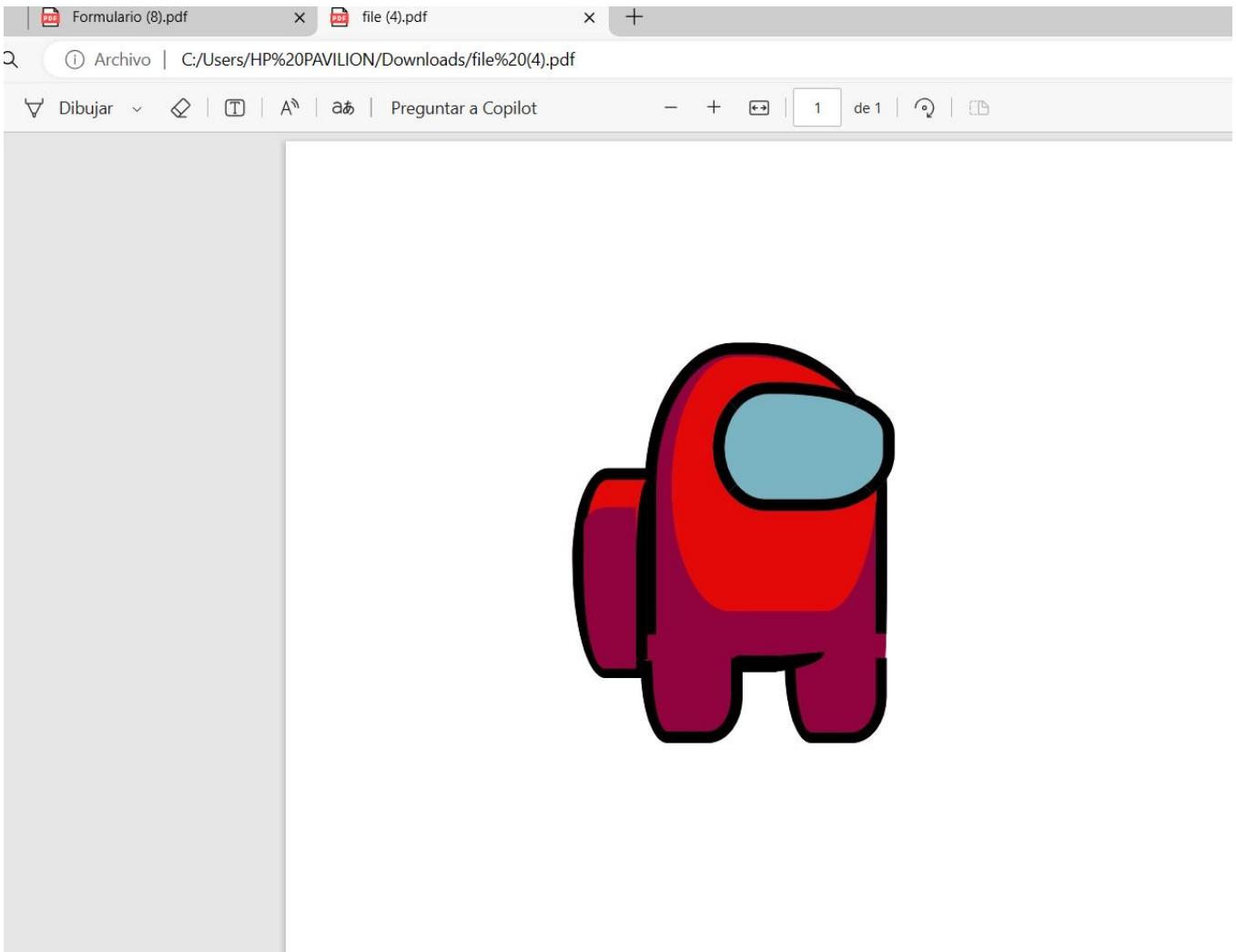
Contenido de la página:



[Generar PDF](#)



[Generar PDF](#)



PRÁCTICA 5 DE HTML5: TABLAS DE MULTIPLICAR

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5, CSS Y JAVASCRIPT

Descripción de la página web: La página web creada consiste en que el usuario ingrese un número y automáticamente cree la tabla de multiplicar del 1 al 9

Contenido de la página:

The screenshot shows a web browser window with a modal dialog box. The dialog has a light gray background and a white content area. At the top left, it says "Esta página dice". Below that is a text input field with the placeholder "Ingrese el número para generar la tabla de multiplicar:". To the right of the input field are two buttons: a blue rounded rectangle labeled "Aceptar" and a white rounded rectangle labeled "Cancelar". The browser's address bar at the top shows "index.html". On the right side of the browser, there are some interface elements: a "Maps" button, a "PRO" button, a "Tipos d" link, a "p Ph" icon, and a "inglés" button. A "Google Translate" button is also visible.

This screenshot shows the same web browser setup as the previous one, but with a key difference: the text input field now contains the number "8". The rest of the dialog, including the message, placeholder, and buttons, remains the same. The browser's address bar still shows "index.html" and the right-side interface elements are present.

Tablas de multiplicar

$$8 \times 1 = 8$$

$$8 \times 2 = 16$$

$$8 \times 3 = 24$$

$$8 \times 4 = 32$$

$$8 \times 5 = 40$$

$$8 \times 6 = 48$$

$$8 \times 7 = 56$$

$$8 \times 8 = 64$$

$$8 \times 9 = 72$$

PRÁCTICA 6 DE HTML5: NÚMEROS PRIMOS

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos de HTML5, CSS YJAVASCRIPT

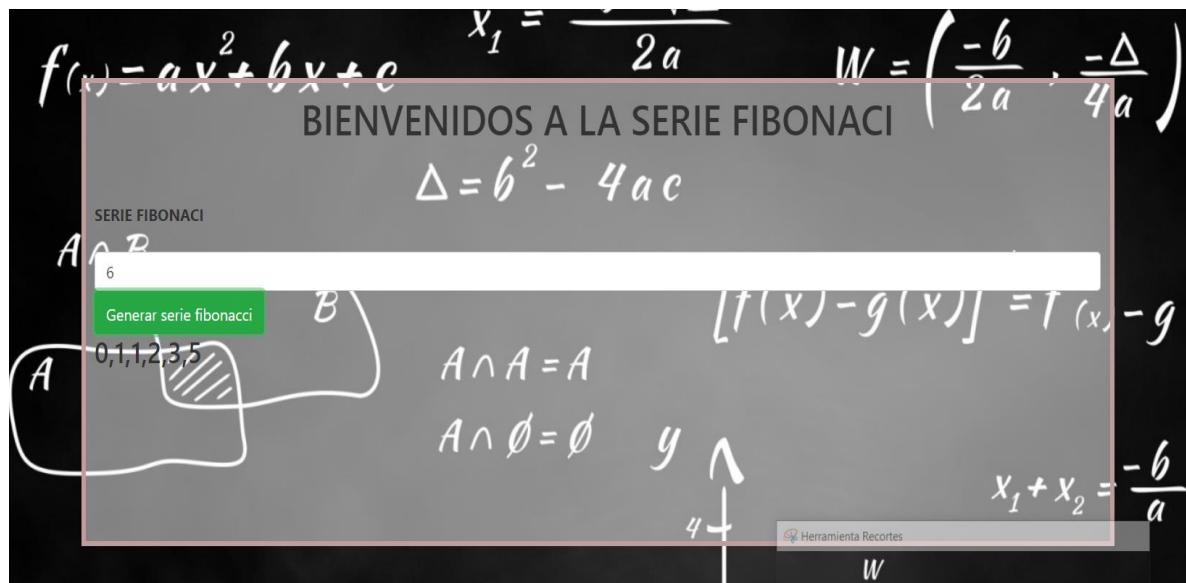
Descripción de la página web: La página web creada consiste en ingresar un número aleatorio y que el programa le diga si es primo o no

The screenshot shows a teal-colored web page with the title "NUMEROS PRIMIOS" in bold black capital letters at the top center. Below the title, there is a text input field containing the number "8". To the right of the input field is a button labeled "checar numero". Underneath the input field, the text "El NUMERO 8 NO ES PRIMO!" is displayed in a large, bold, blue font.

PRÁCTICA 7: SERIE FIBONACI

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos de HTML5, CSS Y JAVASCRIPT

Descripción de la página web: La página web creada consiste en pedirle al usuario un numero al lazar y que genere su serie Fibonacci



Práctica 8 de HTML5: Creación de “leyes de ohm,colum,newton”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5 y CSS y JavaScript para la creación de páginas web relacionales.

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 ,CSS y JavaScript como encabezado, párrafos, imágenes locales.

Contenido de la página:



NEWTON

Ingresa el valor de la aceleracion :

Ingresa el valor de la masa:

EL RESULTADO ES newton :

[Calcular](#)[Volver Atrás](#)

COULOMB

Ingresa el valor de corriente:

54

Ingresa el valor de resistencia:

4

EL RESULTADO ES coulomb:

13.5

[Calcular](#)

[Volver Atrás](#)

OHM

Ingresa el valor de corriente:

solos valores numéricos

Ingresa el valor de resistencia:

solos valores numéricos

EL RESULTADO ES ohm:

calcula

[Volver Atrás](#)

sicos de HTML5 y CSS para la creación de páginas

Práctica 9 de HTML5:leyes de física voltaje, resistencia,aceleracion”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML,CSS Y JAVASCRIPT.

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de una pagina web de leyes de fisica

Contenido de la página:



VOLTAJE

Ingresa el valor de corriente:

4

Ingresa el valor de resistencia:

4

Calcular

[Volver Atrás](#)

El resultado [valotaje.html:83](#)
del voltaje es:16



Iter Default levels ▼

Issues: ✖ 2 |

E1 [corriente.html:91](#)
resultado del CORRIENTE
es:0.4444444444444444

RESISTENCIA

Ingresá el valor de corriente:

Ingresá el valor de resistencia:

Calcular

[Volver Atrás](#)



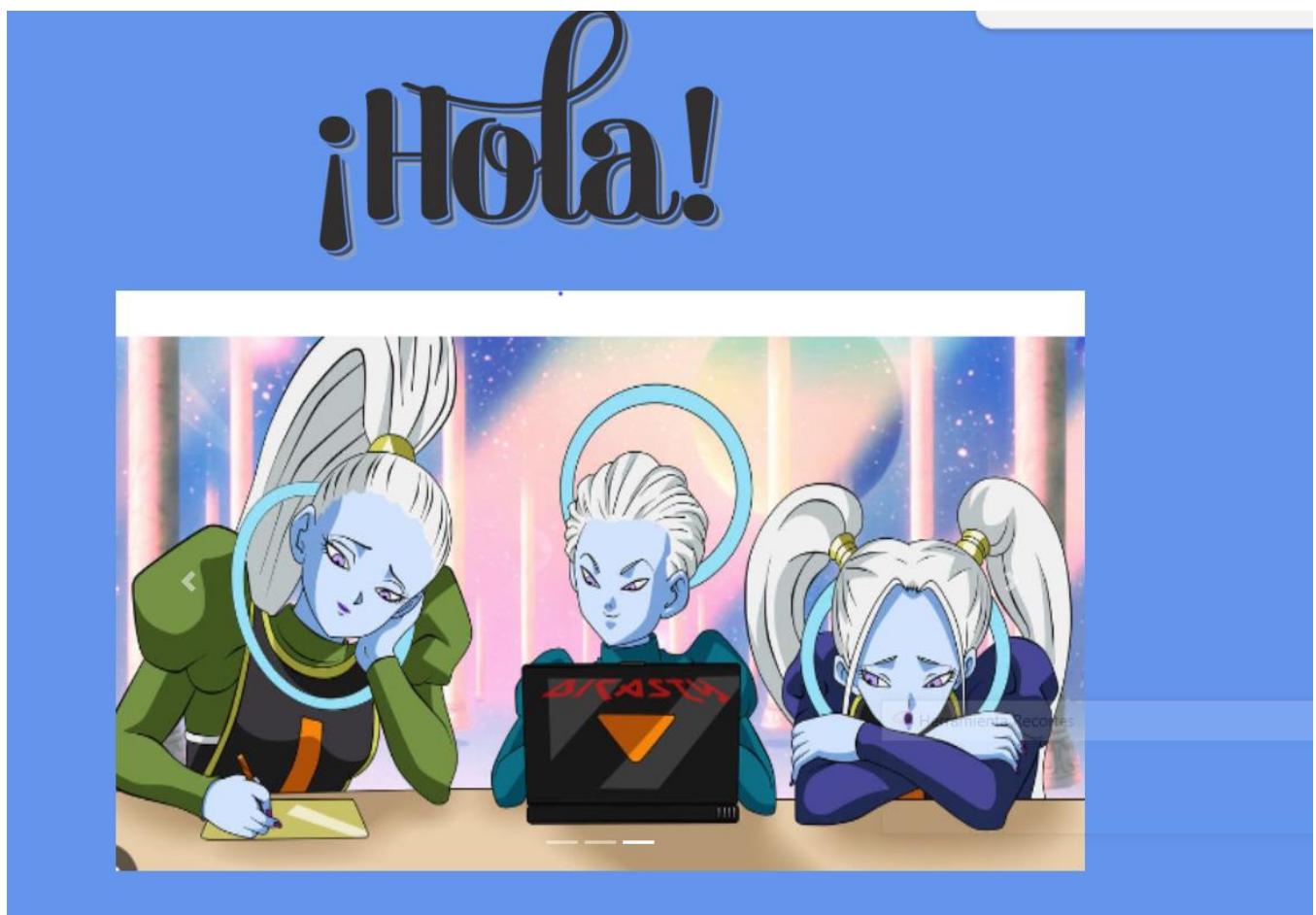
```
resistencia.html:89
resultado del RESISTENCIA
es:1.1666666666666667
```

Práctica 10 de HTML5: Creación de “Carrusel”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML web con una creación de un carrusel.

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de carrusel

Contenido de la página:



Práctica 11 de HTML5: Creación de “juego adivina”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML, CSS y JAVASCRIPT web con una creación de juego

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de un juego adivina el número

Contenido de la página:

ENCUENTRA EL NUMERO DEL
1 AL 10

**Adivina el
Número**

ADIVINA EL NUMERO:

Intentos anteriores: 4 5 7 1 2 3

!!!Fin del juego!!!



Práctica 13 de HTML5: Creación de “tarjetas de volteo”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5 , CSS y JAVASCRIPT para la creación de páginas web con una creación de tarjetas de volteo.

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de tarjetas de volteo

Contenido de la página:

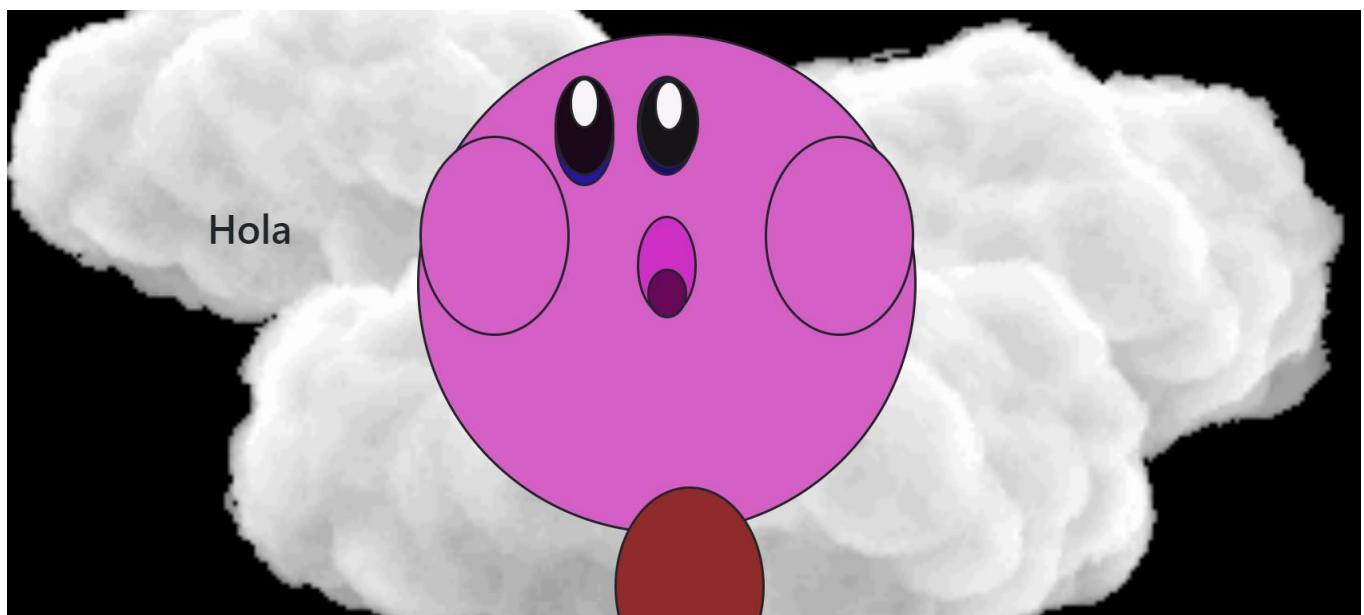


Práctica 14 de HTML5: Creación de “kirby”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5 y CSS para la creación de páginas web con una creación de un kirby

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de una figura de un kirby.

Contenido de la página:



Práctica 15 de HTML5: Creación de “foco”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5 ,CSS y JAVASCRIPT para la creación de páginas web con una creación un foco dinámico

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de un foco dinámico

Contenido de la página:

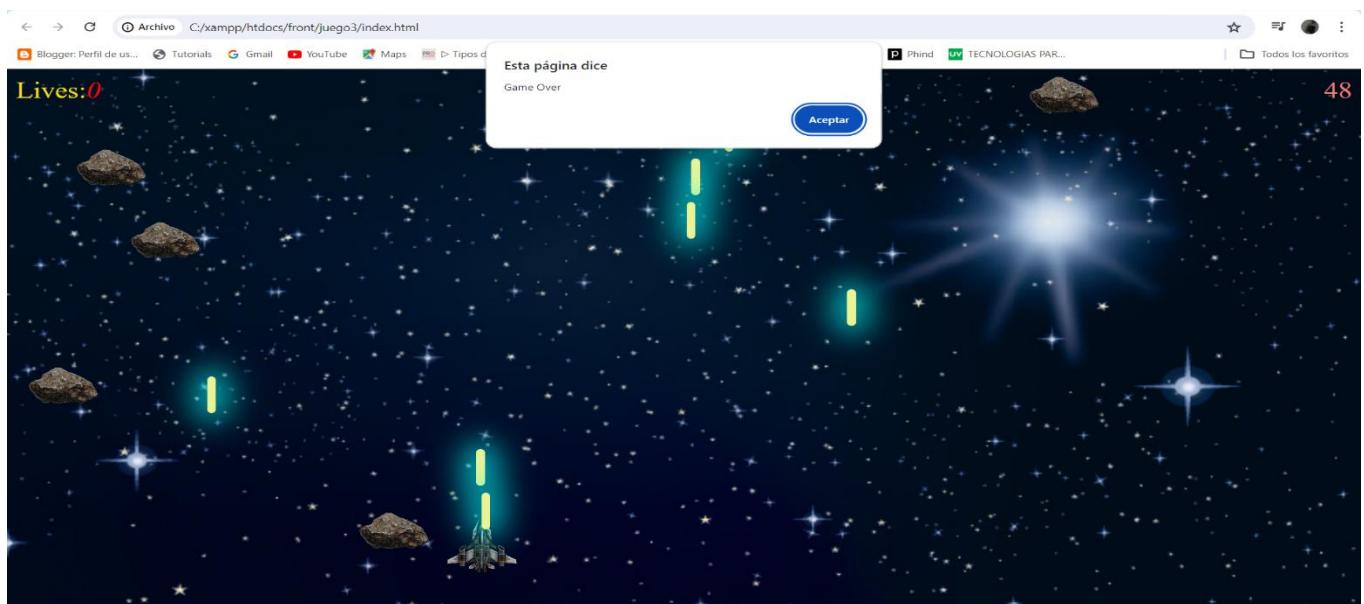


Práctica 16 de HTML5: Creación de “juego de naves”

Objetivo: El objetivo de esta práctica era familiarizarse con los elementos básicos de HTML5 ,CSS y JAVASCRIPT para la creación de un juego

Descripción de la página web: La página web creada consiste en una estructura de elementos HTML5 y con la creación de un juego

Contenido de la página:



2. CONCLUSIÓN

Creo que es interesante observar la aplicación de diferentes conceptos básicos dentro del desarrollo Web. Aprendí bastante sobre el porqué de algunas etiquetas, sus funciones, sus limitantes, así mismo la optimización al momento de comenzar a desarrollar código e implementarlo en una serie de buenas prácticas. Importante siempre mantener una buena sintaxis y el código limpio, así como hacer una creación responsable de nuestras etiquetas y los nombres con los que codificamos las distintas clases a usar.

3. BIBLIOGRAPHY (APA FORMAT)
