

Informe de Laboratorio 03

Tema: Javascript

Nota

Estudiante	Escuela	Asignatura
Miguel Angel Alvarez Choque malvarezcho	Escuela Profesional de Ingeniería de Sistemas	Programación Web Semestre: I Código: 20230477

Laboratorio	Tema	Duración
03	Javascript	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 15 de mayo 2024	Al 17 de mayo 2024

1. Actividades

- Programar en JavaScript sobre una página web html básica.

Listing 1: página web básica

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
</head>
<body>
<!-- no html, generate it with javascript -->
<script src="script_ejercicio_01.js"></script>
</body>
</html>
```


- Ejercicio 01: Cree un teclado random para banca por internet.
- Ejercicio 02: Cree una calculadora básica como la de los sistemas operativos, que pueda utilizar la función eval() y que guarde todos las operaciones en una pila. Mostrar la pila al pie de la página web.
- Ejercicio 03: Cree una versión de el juego 'el ahorcado' que grafique con canvas paso a paso desde el evento onclick() de un botón.

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/miguelnodjan/pw2_24a.git`
- URL para el laboratorio 3 en el Repositorio GitHub.
- `https://github.com/miguelnodjan/pw2_24a/tree/main/lab_03`

2. Ejercicio 1:

2.1. creación de archivo *html*, *css*:

- Para esta parte se creó un archivo *html* base que se irá adaptando según avancemos con el ejercicio.
- Adicional a ello, se necesita de un *css* para que de estilo al *html*.



Listing 2: css

```
body {
  font-family: Arial, sans-serif;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  height: 100vh;
  background-color: #f4f4f4;
}
.container {
```

```
background-color: white;
padding: 20px;
border-radius: 8px;
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
max-width: 400px;
width: 100%;
}
.container h2 {
margin-bottom: 20px;
color: #333;
}
.form-group {
margin-bottom: 15px;
}
.form-group label {
display: block;
margin-bottom: 5px;
color: #555;
}
.form-group input[type="text"],
.form-group input[type="password"],
.form-group button {
width: 100%;
padding: 10px;
border: 1px solid #ccc;
border-radius: 5px;
box-sizing: border-box;
}
.form-group button {
background-color: #007bff;
color: white;
font-size: 16px;
cursor: pointer;
}
.form-group button:hover {
background-color: #0056b3;
}
#keyboard {
display: grid;
grid-template-columns: repeat(3, 1fr);
gap: 10px;
max-width: 300px;
margin: 20px auto 0;
}
.key {
background-color: #007bff;
color: white;
border: none;
padding: 20px;
font-size: 20px;
border-radius: 5px;
cursor: pointer;
user-select: none;
}
.key:hover {
background-color: #0056b3;
```

```
}  
#inputDisplay {  
  margin-bottom: 20px;  
  padding: 10px;  
  border: 2px solid #ccc;  
  border-radius: 5px;  
  text-align: center;  
  font-size: 20px;  
  background-color: white;  
}  
.respuesta{  
  transition: 0.5s;  
}  
.respuesta:hover{  
  transform: scale(1.1);  
}  
}
```

2.2. crear archivo *Javascript*

- para poder hacer funcional este ejemplo de banca por internet se necesita de un archivo *js* para que genere la interacción con el cliente.
- El archivo resultante es el siguiente que tienen como funciones:
 - **generateKey():** Lo que hace es resetea los valores a *input* y da inicio al método *generateRandomKeys*.
 - **generateRandomKeys():** Lo que hace este método es generar el tablero para colocar los números en un orden aleatorio.
 - **ingresar():** Lo que hace es una vez le das al botón ingresar, te muestra un mensaje respectivo si los espacios están en blanco.

Listing 3: script.js

```
const inputDisplay = document.getElementById('inputDisplay');  
const keyboard = document.getElementById('keyboard');  
let input = '';  
  
function generateKey() {  
  input = '';  
  inputDisplay.textContent = '****';  
  generateRandomKeys();  
}  
  
function generateRandomKeys() {  
  const keys = [...Array(10).keys()];  
  keys.sort(() => Math.random() - 0.5);  
  keyboard.innerHTML = '';  
  keys.forEach(key => {  
    const keyElement = document.createElement('button');  
    keyElement.className = 'key';  
    keyElement.textContent = key;  
    keyElement.addEventListener('click', () => {  
      if (input.length < 4) {
```

```
        input += key;
        inputDisplay.textContent = '*'.repeat(input.length);
    }
});
keyboard.appendChild(keyElement);
});
}
function ingresar() {
    const cardNumber = document.getElementById('cardNumber').value;
    const documentId = document.getElementById('documentId').value;
    if (cardNumber && documentId && input.length === 4) {
        alert('Ingresando...');
    } else {
        alert('Por favor, complete todos los campos correctamente.');
```

```
    }
}
```

```
window.onload = generateRandomKeys;
```

Ingreso a Banca por Internet

Número de Tarjeta

Documento de Identidad

Generar Clave de Internet

Clave de Internet

7

3

1

0

8

5

2

9

6

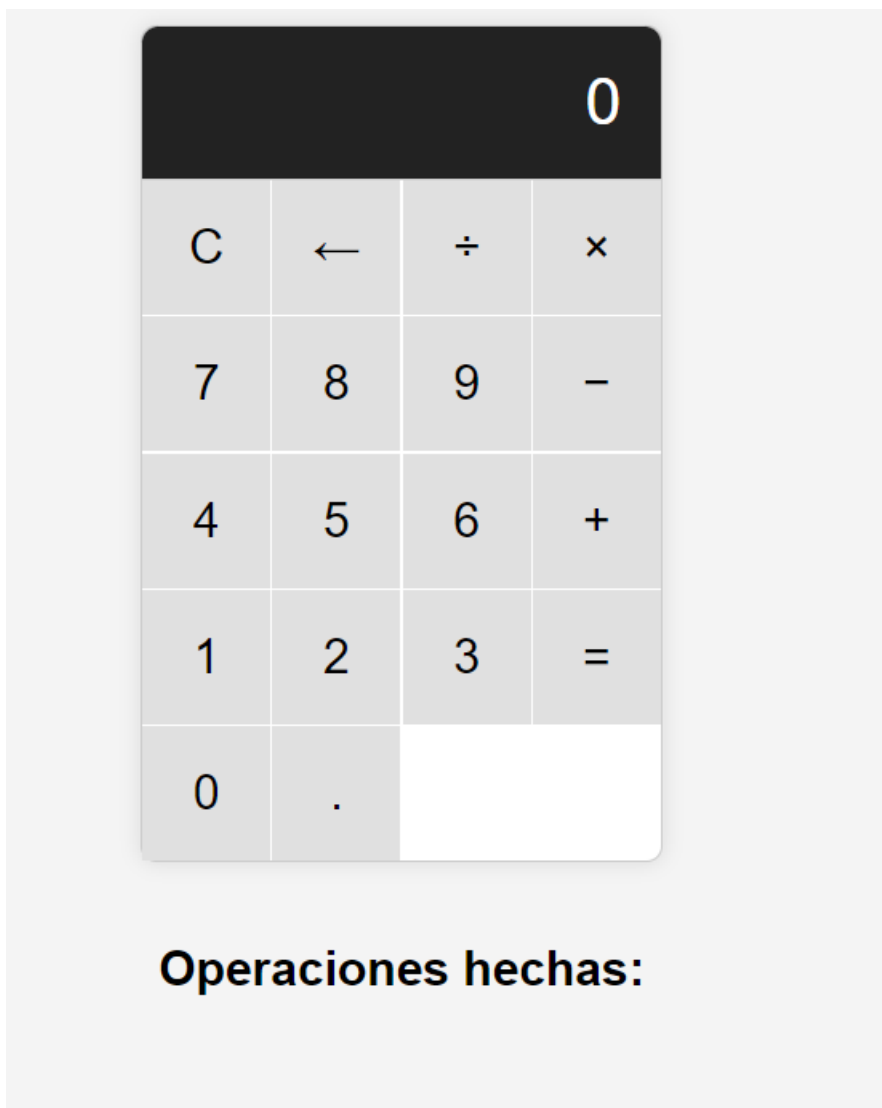
4

Ingresar

3. Ejercicio2 :

3.1. creando los archivos html y css

- Se procedió a crear los archivos *css* y *html* para la calculadora,
- luego se procedera a darle funcionalidad con el archivo *js*.
- El resultado se muestra a continuación.



3.2. Completando scripts.js

- Se añadió el código al *js* para que tenga funcionalidad el juego y que se pueda usar *canva*.
- Se procede a presentar el resultado final con la implementación del archivo *scripts.js*.

Listing 4: scripts.js

```
const display = document.getElementById('display');
const historyList = document.getElementById('history');
let displayValue = '';
let history = [];

function clearDisplay() {
    displayValue = '';
    updateDisplay();
}

function deleteLast() {
    displayValue = displayValue.slice(0, -1);
    updateDisplay();
}

function appendNumber(number) {
    if (displayValue === '0') {
        displayValue = number;
    } else {
        displayValue += number;
    }
    updateDisplay();
}

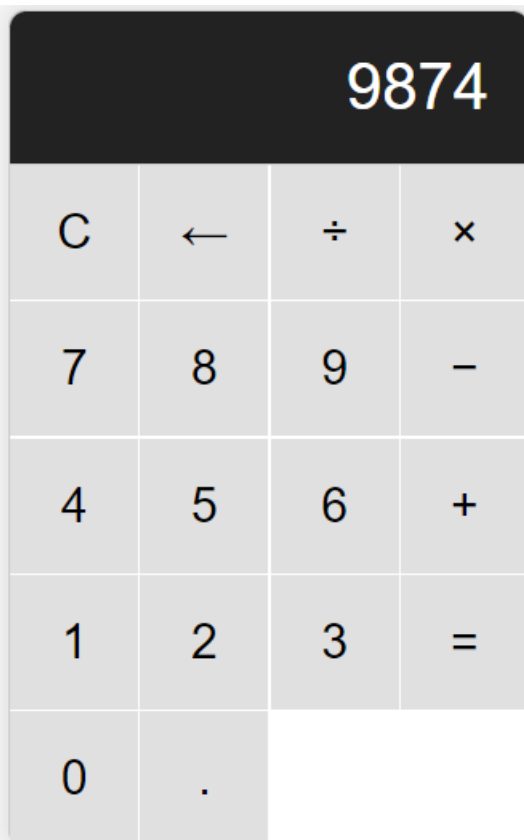
function appendOperator(operator) {
    if (displayValue === '') {
        return;
    }
    const lastChar = displayValue[displayValue.length - 1];
    if ('+-*/'.includes(lastChar)) {
        displayValue = displayValue.slice(0, -1);
    }
    displayValue += operator;
    updateDisplay();
}

function calculate() {
    try {
        const result = eval(displayValue).toString();
        history.push(`${displayValue} = ${result}`);
        displayValue = result;
        updateHistory();
    } catch (e) {
        displayValue = 'Error';
    }
    updateDisplay();
}

function updateDisplay() {
```



```
    display.textContent = displayValue || '0';  
}  
  
function updateHistory() {  
    historyList.innerHTML = '';  
    history.forEach(operation => {  
        const li = document.createElement('li');  
        li.textContent = operation;  
        historyList.appendChild(li);  
    });  
}
```



Operaciones hechas:

$$1 * 9 = 9$$

$$96 / 9 = 10.666666666666666$$

$$1 * 9874 = 9874$$

4. Ejercicio 3 :

4.1. creando los archivos html y css

- Se procedió a crear los archivos *css* y *html* para el juego del ahorcado,
- luego se procedera a darle funcionalidad con el archivo *js*.
- El resultado se muestra a continuación.



4.2. Completando scripts.js

- Se añadió el código al *js* para que tenga funcionalidad el juego.
- Para este ejercicio se buscó ayuda en internet por lo complejo que resultó la funcionalidad del gráfico del ahorcado
- Se procede a presentar el resultado final con la implementación del archivo *scripts.js*.

Listing 5: scripts.js

```
const canvas = document.getElementById('hangmanCanvas');
const ctx = canvas.getContext('2d');
const wordElement = document.getElementById('word');
const letterInput = document.getElementById('letterInput');
const messageElement = document.getElementById('message');

const words = ['computadora', 'ardilla', 'programacion', 'ahorcado', 'holamundo',
  'sistemas', 'jugar', 'palta', 'esternocleidomastoideo'];
const word = words[Math.floor(Math.random() * words.length)];
let guessedWord = '_'.repeat(word.length).split('');
let incorrectGuesses = 0;

function drawHangman() {
  ctx.clearRect(0, 0, canvas.width, canvas.height);
  ctx.lineWidth = 2;

  if (incorrectGuesses > 0) {
    ctx.beginPath();
    ctx.moveTo(10, 390);
    ctx.lineTo(190, 390);
    ctx.stroke();
  }
  if (incorrectGuesses > 1) {
    ctx.beginPath();
    ctx.moveTo(50, 390);
    ctx.lineTo(50, 50);
    ctx.stroke();
  }
  if (incorrectGuesses > 2) {
    ctx.beginPath();
    ctx.moveTo(50, 50);
    ctx.lineTo(150, 50);
    ctx.stroke();
  }
  if (incorrectGuesses > 3) {
    ctx.beginPath();
    ctx.moveTo(150, 50);
    ctx.lineTo(150, 100);
    ctx.stroke();
  }
  if (incorrectGuesses > 4) {
    ctx.beginPath();
    ctx.arc(150, 130, 30, 0, Math.PI * 2, true);
    ctx.stroke();
  }
  if (incorrectGuesses > 5) {
    ctx.beginPath();
    ctx.moveTo(150, 160);
    ctx.lineTo(150, 250);
    ctx.stroke();
  }
  if (incorrectGuesses > 6) {
    ctx.beginPath();
    ctx.moveTo(150, 180);
    ctx.lineTo(120, 230);
  }
}
```

```
        ctx.stroke();
    }
    if (incorrectGuesses > 7) {
        ctx.beginPath();
        ctx.moveTo(150, 180);
        ctx.lineTo(180, 230);
        ctx.stroke();
    }
    if (incorrectGuesses > 8) {
        ctx.beginPath();
        ctx.moveTo(150, 250);
        ctx.lineTo(120, 320);
        ctx.stroke();
    }
    if (incorrectGuesses > 9) {
        ctx.beginPath();
        ctx.moveTo(150, 250);
        ctx.lineTo(180, 320);
        ctx.stroke();
    }
}

function guessLetter() {
    const letter = letterInput.value.toLowerCase();
    letterInput.value = '';
    messageElement.textContent = '';

    if (!letter.match(/[a-z]/) || letter.length !== 1) {
        messageElement.textContent = 'Por favor, ingresa una letra vlida.';
        return;
    }

    if (word.includes(letter)) {
        for (let i = 0; i < word.length; i++) {
            if (word[i] === letter) {
                guessedWord[i] = letter;
            }
        }
    } else {
        incorrectGuesses++;
        drawHangman();
    }

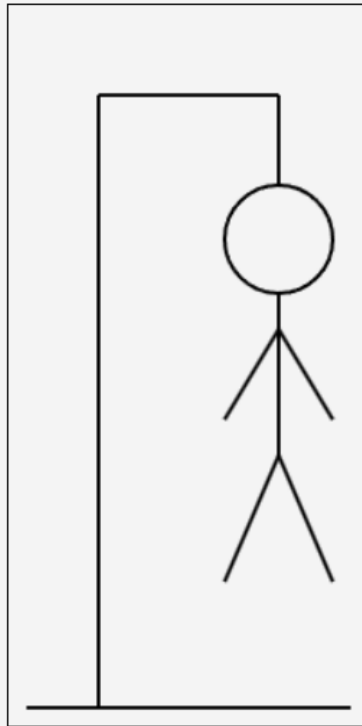
    wordElement.textContent = guessedWord.join(' ');

    if (!guessedWord.includes('_')) {
        messageElement.textContent = 'Felicidades, ganaste :)!';
    } else if (incorrectGuesses > 9) {
        messageElement.textContent = 'Perdiste :C. La palabra era: ${word}, ms suerte a la proxima';
    }
}

function init() {
    wordElement.textContent = guessedWord.join(' ');
    drawHangman();
}
```

```
}  
  
init();
```

El Juego del Ahorcado



— — — a — — — — —

Ingresa una letra

Adivinar

Perdiste :C. La palabra era: holamundo, más suerte a la proxima

4.3. Commits:

- A continuación se muestra una captura de pantalla de los principales *commits* para la creación de la página web

añadiendo las funciones a scripts.js del ejercicio 3 miguelnodjan committed 7 minutes ago	0952a07		
completado el html y css del ejercicio 3 miguelnodjan committed 18 minutes ago	374604c		
añadiendo las funciones en scripts.js y creando los archivos necesarios para el ejercicio3 miguelnodjan committed 24 minutes ago	10bc5a7		
arreglando el html y css del ejercicio2 miguelnodjan committed 29 minutes ago	119a326		
creando primeras versiones del html y css para la calculadora del ejercicio 2 miguelnodjan committed 36 minutes ago	7350a1f		
creando los archivos html, css y js para el ejercicio 2 miguelnodjan committed 40 minutes ago	68ce707		
corrigiendo script.js miguelnodjan committed 48 minutes ago	271c995		
se implemento el método generateRandomKeys() en scripts.js miguelnodjan committed 1 hour ago	378d90b		
añadiendo las funciones generatekey() e ingresar() en script.js de ejercicio 1 miguelnodjan committed 5 hours ago	66be4c3		
creando el archivo script.js para el ejercicio 1 miguelnodjan committed 5 hours ago	6ca5833		
añadiendo el contenido a html y css del ejercicio 1 miguelnodjan committed 5 hours ago	a970f30		

4.4. Cuestionario

- Explique una herramienta para ofuzcar código JavaScript.
 - Una de las herramientas más utilizadas en *JavaScript Obfuscator*. Esta herramienta permite ofuscar el código que creamos de manera facil y rapida, inclusive atraves de una intefaz web. Otra opción un poco menos especializada es *UglifyJS* que puede servir para una ofuscación básica.
- Muestre un ejemplo de su uso en uno de los ejercicios de la tarea.

Ingreso a Banca por Internet

Número de Tarjeta

Documento de Identidad

Generar Clave de Internet

Clave de Internet

7

3

1

0

8

5

2

9

6

4

Ingresar

9874			
C	←	÷	×
7	8	9	-
4	5	6	+
1	2	3	=
0	.		

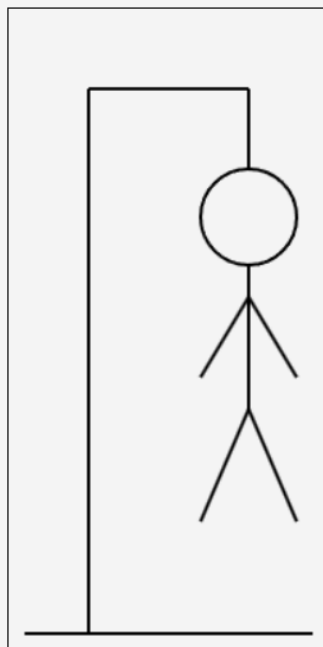
Operaciones hechas:

1*9 = 9

96/9 = 10.6666666666666666

1*9874 = 9874

El Juego del Ahorcado



— — — a — — — — —

Ingresa una letra

Adivinar

Perdiste :C. La palabra era: holamundo, más suerte a la proxima

-
- Adjunte a su repositorio ambas versiones:
 - `script_ejercicio01.js(development).script_ejercicio01.min.js(production)`.

4.5. Estructura de laboratorio 2

```
lab_3
|-----ejercicio_1
|       |-----index.html
|       |-----script.js
|       |-----style.css
|
|-----ejercicio_2
|       |-----index.html
|       |-----scripts.js
|       |-----style.css
|
|-----ejercicio_3
|       |-----index.html
|       |-----scripts.js
|       |-----style.css
|
+-----images
|       |-----cap1.png
|       |-----cap2.png
|       |-----cap5.png
|       |-----cap6.png
|       |-----cap7.png
|       |-----cap8.png
|       |-----cap9.png
```

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe		
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y fácil de leer.	Nota
Observaciones	Respetar la estructura de organización para la ubicación de los entregables. Por cada observación dentro del informe se le descontará puntos. Se debe incluir el código fuente latex del informe	

5.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

Nivel				
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	1	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	1	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	x	2	
Total		20		12	

6. Bibliografía:

<http://mally.stanford.edu/~sr/computing/basic-unix.html>

<https://www.technology.pitt.edu/help-desk/how-to-documents/basic-unix-commands>