



**UNIVERSIDAD DE LAS FUERZAS ARMADAS
ESPE UNIDAD DE EDUCACIÓN A DISTANCIA**



Asignatura:

Programación Integrativa Componentes Web

Ing.

Vilmer David Criollo Chanchicocha

Tema:

Números Pares e Impares con Web Components

Nombre:

Carlos Ramiro Yáñez Yazán

Fecha:

Quito.....15.....julio de 2025

TEMA: Números Pares e Impares con Web Components

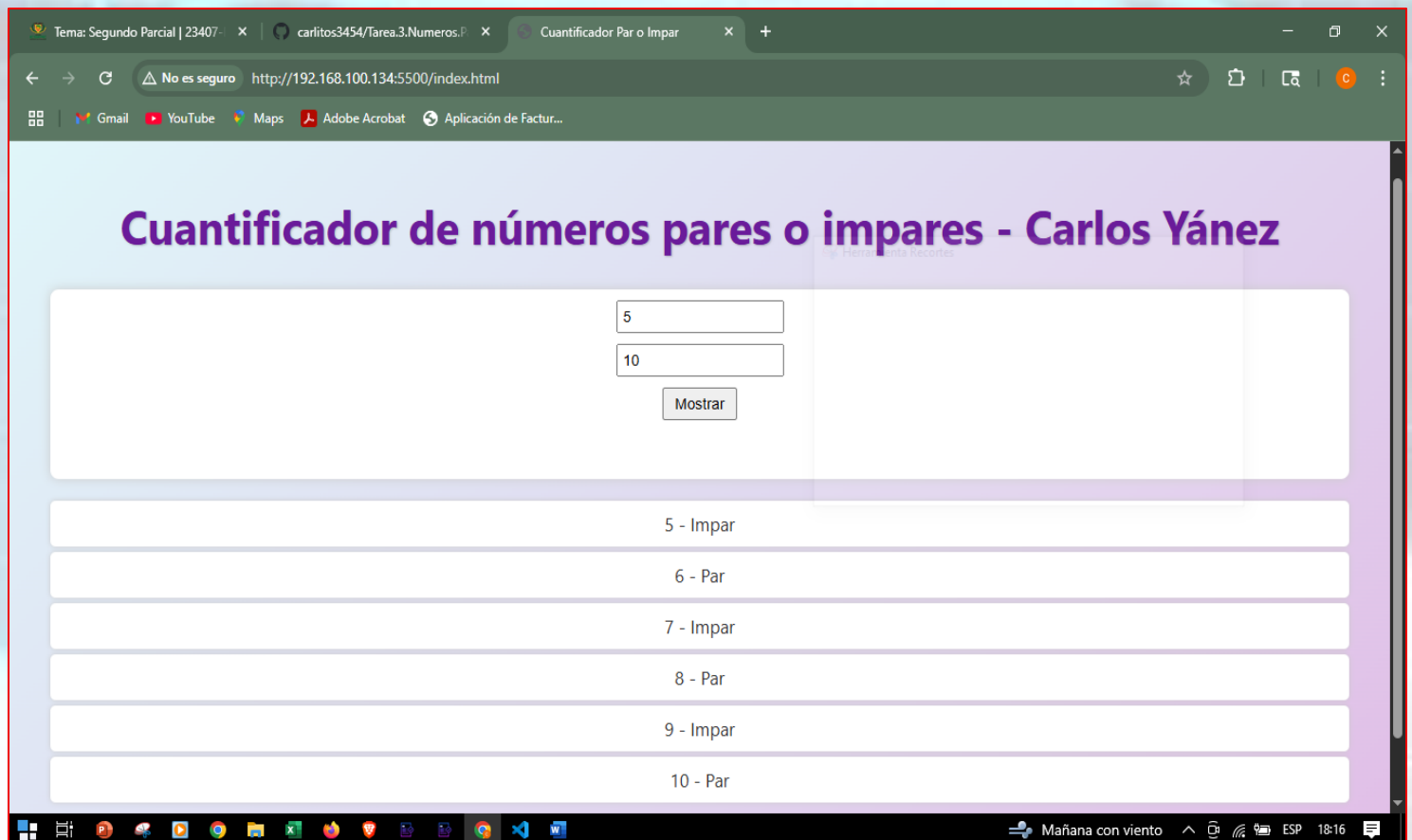
ESTUDIANTE: Carlos Ramiro Yáñez Yazán

FECHA: Martes 15 de Julio del 2025

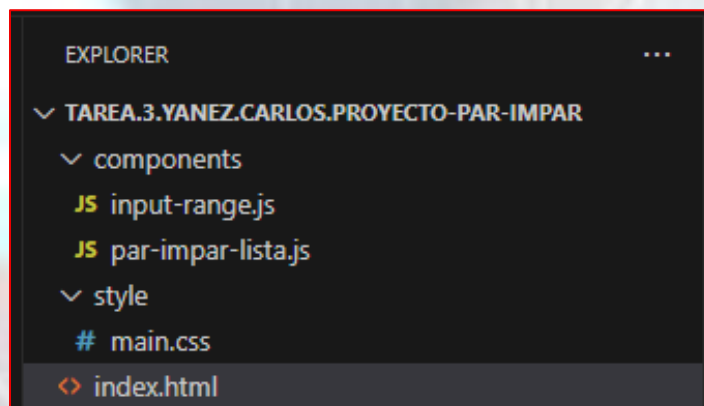
1. ENLACE GITHUB

<https://github.com/carlitos3454/Tarea.3.Numeros.Pares.Impares-con-Web-Components.git>

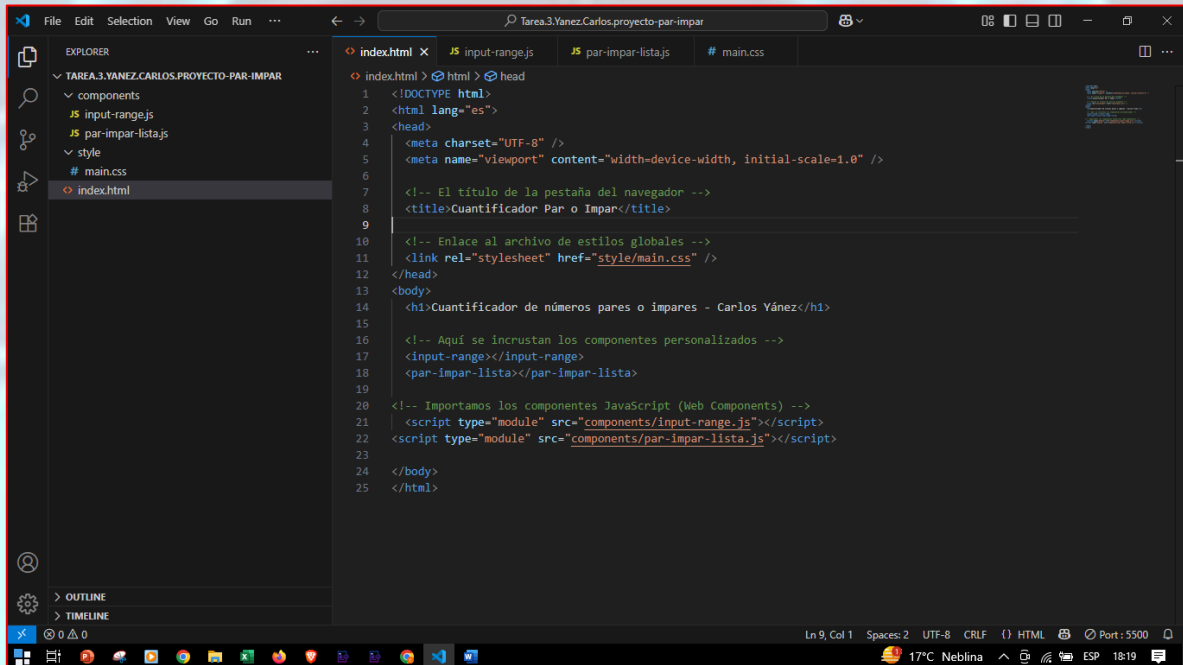
2. CODIGO FUENTE O EJECUCIÓN.



ESTRUCTURA DEL PROYECTO.



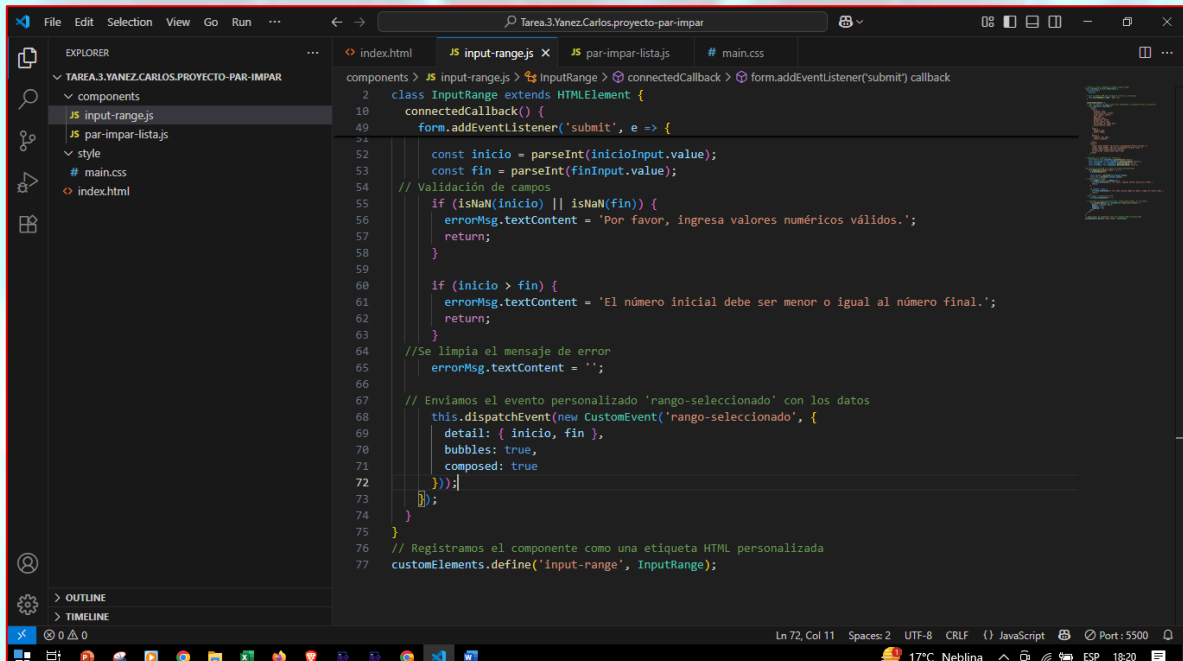
Index.html

A screenshot of the Visual Studio Code editor. The Explorer sidebar on the left shows a project named 'TAREA.3.YANEZ.CARLOS.PROYECTO-PAR-IMPAT' with subfolders 'components', 'style', and 'main.css'. The 'index.html' file is selected. The main editor area displays the content of 'index.html'. The code is in Spanish and includes a DOCTYPE declaration, HTML lang attribute, meta tags for charset and viewport, a title 'Cuantificador Par o Impar', a link to 'style/main.css', and a body containing an h1 title 'Cuantificador de números pares o impares - Carlos Yáñez', a comment about custom components, and two custom components: 'input-range' and 'par-impar-lista'. It also includes script tags for 'components/input-range.js' and 'components/par-impar-lista.js'.

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8" />
5   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6
7   <!-- El título de la pestaña del navegador -->
8   <title>Cuantificador Par o Impar</title>
9
10  <!-- Enlace al archivo de estilos globales -->
11  <link rel="stylesheet" href="style/main.css" />
12 </head>
13 <body>
14   <h1>Cuantificador de números pares o impares - Carlos Yáñez</h1>
15
16   <!-- Aquí se incrustan los componentes personalizados -->
17   <input-range></input-range>
18   <par-impar-lista></par-impar-lista>
19
20  <!-- Importamos los componentes JavaScript (Web Components) -->
21  <script type="module" src="components/input-range.js"></script>
22  <script type="module" src="components/par-impar-lista.js"></script>
23
24 </body>
25 </html>
```

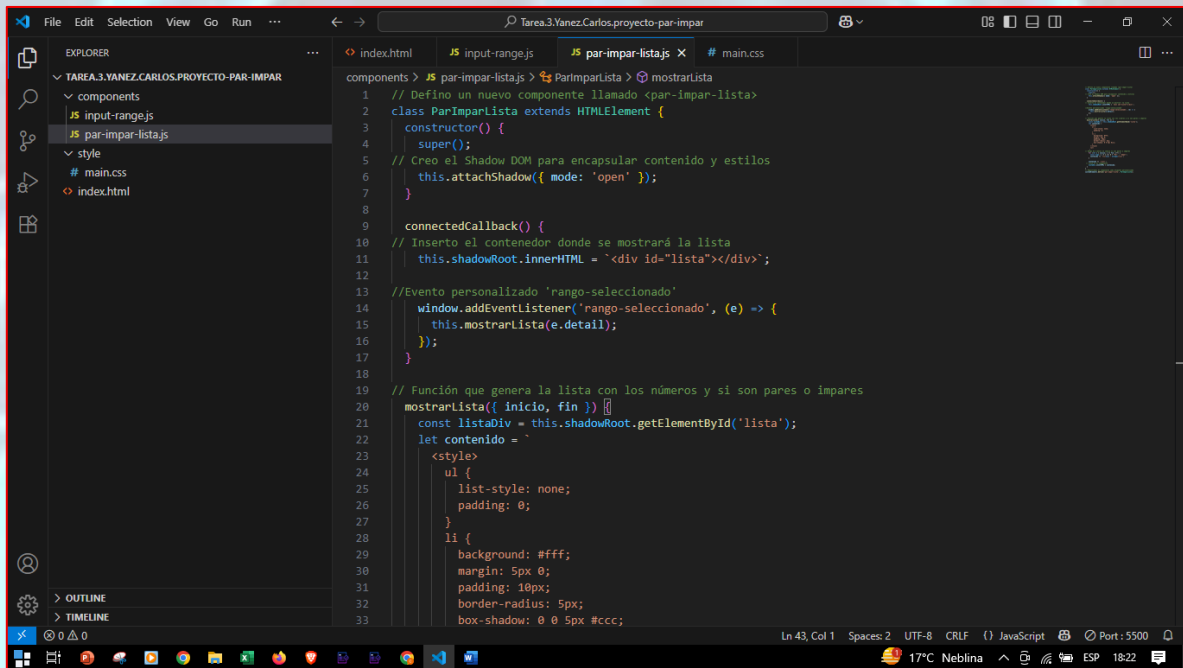
Archivos java/js_COMPONENTS.

Input_range.js

A screenshot of the Visual Studio Code editor. The Explorer sidebar on the left shows the same project structure as the previous screenshot. The 'input-range.js' file is selected. The main editor area displays the content of 'input-range.js'. The code is in Spanish and defines a custom web component 'InputRange' that extends 'HTMLElement'. It includes a 'connectedCallback' method that adds an event listener for the 'submit' event. The event listener calls 'form.addEventListeners' with 'submit' and a callback function. The callback function parses the input values, validates them (checking for NaN and range), and dispatches a custom event 'rango-seleccionado' with details {inicio, fin}. It also registers the component as a custom HTML element.

```
1 components > JS input-range.js > InputRange > connectedCallback > form.addEventListeners('submit') callback
2 class InputRange extends HTMLElement {
3   connectedCallback() {
4     form.addEventListeners('submit', e => {
5
6       const inicio = parseInt(inicioInput.value);
7       const fin = parseInt(finInput.value);
8
9       // Validación de campos
10      if (isNaN(inicio) || isNaN(fin)) {
11        errorMsg.textContent = 'Por favor, ingresa valores numéricos válidos.';
12        return;
13      }
14
15      if (inicio > fin) {
16        errorMsg.textContent = 'El número inicial debe ser menor o igual al número final.';
17        return;
18      }
19
20      //Se limpia el mensaje de error
21      errorMsg.textContent = '';
22
23      // Enviamos el evento personalizado 'rango-seleccionado' con los datos
24      this.dispatchEvent(new CustomEvent('rango-seleccionado', {
25        detail: { inicio, fin },
26        bubbles: true,
27        composed: true
28      }));
29    });
30  }
31
32  // Registramos el componente como una etiqueta HTML personalizada
33  customElements.define('Input-range', InputRange);
34 }
```

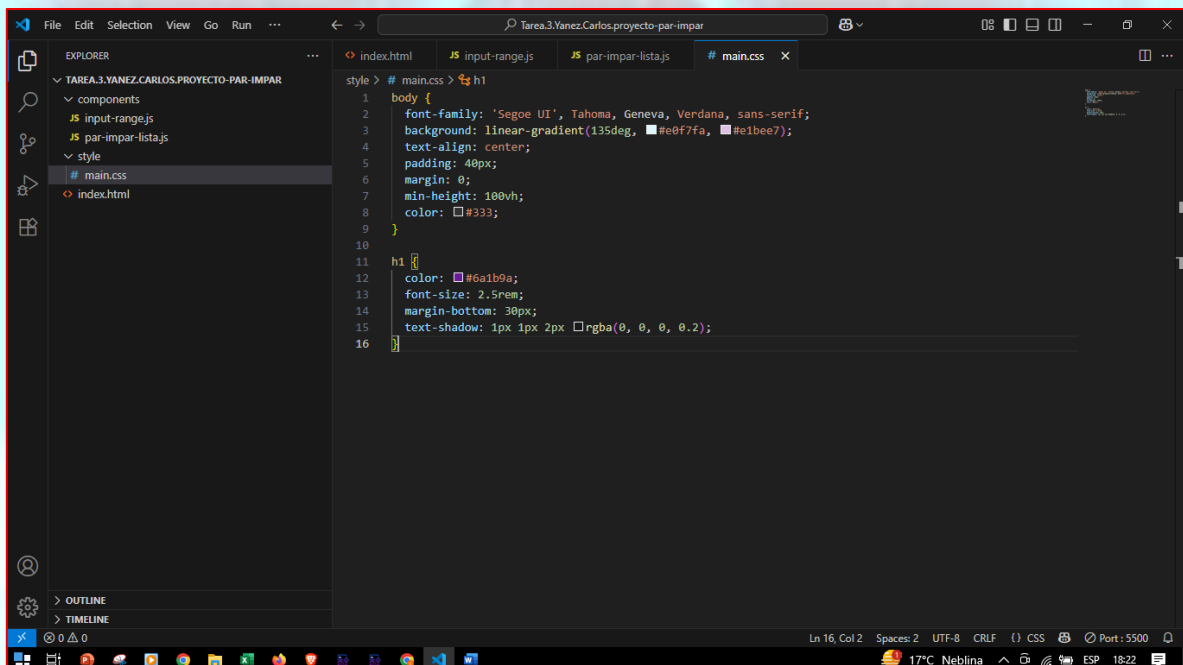
par-impar-list.js



The screenshot shows the VS Code editor with the file explorer on the left displaying the project structure: TAREA.3.YANEZ.CARLOS.PROYECTO-PAR-IMPAP, components, JS input-range.js, JS par-impar-listajs, style, # main.css, and index.html. The main editor displays the JavaScript code for the ParImparLista web component in the file par-impar-listajs.js. The code defines a class ParImparLista extending HTMLElement, with a constructor, a connectedCallback method, and a custom event 'rango-seleccionado'. It also includes a mostrarLista function that generates a list of numbers based on the selected range.

```
1 // Defino un nuevo componente llamado <par-impar-lista>
2 class ParImparLista extends HTMLElement {
3   constructor() {
4     super();
5   }
6   // Creo el Shadow DOM para encapsular contenido y estilos
7   this.attachShadow({ mode: 'open' });
8
9   connectedCallback() {
10    // Inserto el contenedor donde se mostrará la lista
11    this.shadowRoot.innerHTML = `<div id="lista"></div>`;
12
13    //Evento personalizado 'rango-seleccionado'
14    window.addEventListener('rango-seleccionado', (e) => {
15      this.mostrarLista(e.detail);
16    });
17
18    // Función que genera la lista con los números y si son pares o impares
19    mostrarLista(( inicio, fin )) {
20      const listaDiv = this.shadowRoot.getElementById('lista');
21      let contenido = `
22      <style>
23        ul {
24          list-style: none;
25          padding: 0;
26        }
27        li {
28          background: #fff;
29          margin: 5px 0;
30          padding: 10px;
31          border-radius: 5px;
32          box-shadow: 0 0 5px #ccc;
33      `
```

Hoja de estilo css



The screenshot shows the VS Code editor with the file explorer on the left displaying the project structure: TAREA.3.YANEZ.CARLOS.PROYECTO-PAR-IMPAP, components, JS input-range.js, JS par-impar-listajs, style, # main.css, and index.html. The main editor displays the CSS code for the web component in the file # main.css. The code defines the styles for the body and the h1 element, including font-family, background, text-align, padding, margin, min-height, color, font-size, margin-bottom, and text-shadow.

```
1 body {
2   font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
3   background: linear-gradient(135deg, #e0f7fa, #e1bee7);
4   text-align: center;
5   padding: 40px;
6   margin: 0;
7   min-height: 100vh;
8   color: #333;
9 }
10
11 h1 {
12   color: #6a1b9a;
13   font-size: 2.5rem;
14   margin-bottom: 30px;
15   text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.2);
16 }
```