

Informe de Laboratorio 04

Tema: Ajax

Nota

Estudiante	Escuela	Asignatura
Luis Guillermo Luque Condori, Fernando Miguel Garambel Marín, William Herderson Choquehuanca Berna, Jeans Anthony Ajra Huacso lluquecon@unsa.edu.pe fgarambel@unsa.edu.pe ajrahuacso@unsa.edu.pe wchoquehuancab@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Laboratorio de P web Semestre: III Código: 20233478

Laboratorio	Tema	Duración
04	Ajax	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 4 de mayo55555	Al 8 de mayo 2024

1. Tarea Documento

- Listar los archivos Markdown disponibles
- Ver el contenido de un archivo Markdown traducido a HTML
- Crear nuevos archivos MarkDown y almacenarlos en el servidor

2. Tarea Plataforma

- Liste todas las “regiones”
- Muestre el número total de confirmados por región
- Encuentre las 10 regiones cuya suma total sea la mayor
- Visualice un gráfico en el tiempo de los valores para la región de Arequipa
- Haga gráficos comparativos entre regiones usando líneas
- Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao
- Haga gráficos comparativos entre regiones elegidas por el usuario.
- Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao, mostrando el número de confirmados por cada día

3. URL GitHub

- <https://github.com/FernandoGarambelM/Ajax.git>

4. Tareas Documento

- 4.1. Listar los archivos Markdown disponibles
- 4.2. Ver el contenido de un archivo Markdown traducido a HTML
- 4.3. Crear nuevos archivos MarkDown y almacenarlos en el servidor

5. Tareas Plataforma

5.1. Liste las regiones

- HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="style.css">
  <title>Ejercicio1</title>
</head>
<body>
  <div id="datos">
    <h1>LISTA DE REGIONES</h1>
    <button id="btnMostrarRegiones">Mostrar Regiones</button>
    <div class="contenedor">
      <ul id="regionList"></ul>
    </div>
  </div>
  <script src="ejercicio1.js"></script>
</body>
</html>
```

- Script

```
document.addEventListener("DOMContentLoaded", () => {
  btnMostrarRegiones.addEventListener('click', () => {
  });
});

function mostrarRegiones(data) {
  const regionList = document.getElementById('regionList');
  data.forEach(regionData => {
    const li = document.createElement('li');
    li.textContent = regionData.region;
    regionList.appendChild(li);
  });
}
```

- Página

LISTA DE REGIONES

Mostrar Regiones

- Resultado

LISTA DE REGIONES

Amazonas
Ancash
Apurímac
Arequipa
Ayacucho
Cajamarca
Callao
Cusco
Huancavelica
Huanuco
Ica
Junín
La Libertad
Lambayeque
Lima
Loreto
Madre de Dios
Moquegua
Pasco
Piura
Puno
San Martín
Tacna
Tumbes
Ucayali

5.2. Muestre el número total de confirmados por región

- HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Confirmados por Región</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Número Total de Confirmados por Región</h1>
  <div id="regiones"></div>
  <button id="btnConfirmadosRegion">Mostrar Confirmados</button>

  <script src="ejercicio2.js"></script>
</body>
</html>
```

- Script

```
document.addEventListener("DOMContentLoaded", function() {
  const btnConfirmadosRegion = document.getElementById('btnConfirmadosRegion');

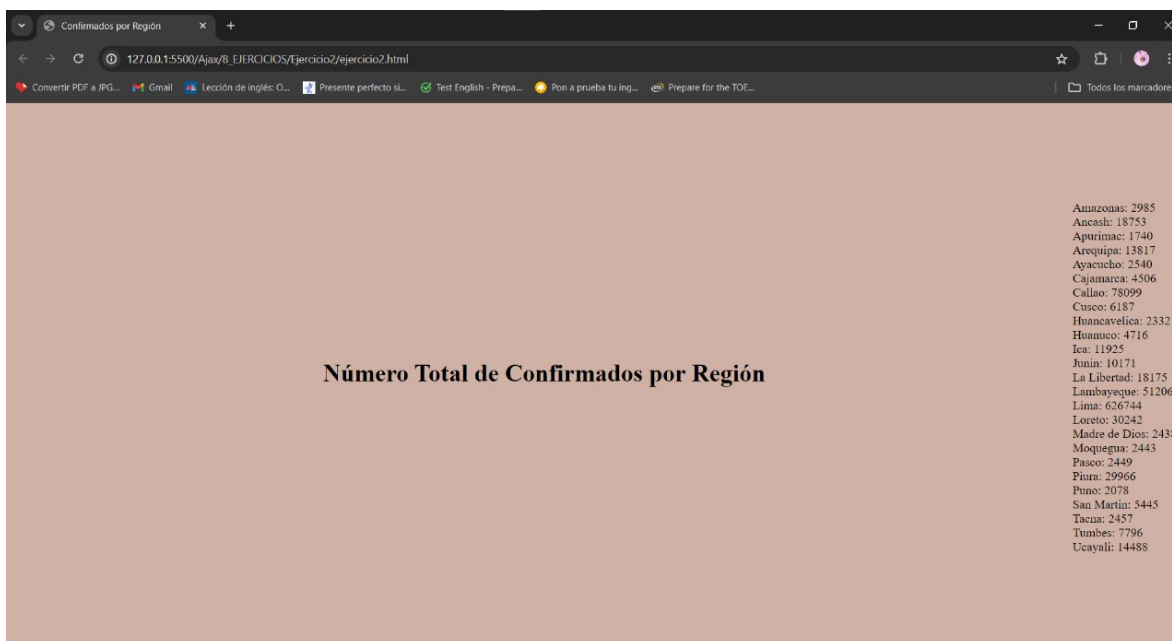
  btnConfirmadosRegion.addEventListener('click', () => {
    // Realizar la solicitud
    let xhr = new XMLHttpRequest();
    xhr.open("GET", "../data.json", true);
    xhr.onload = function() {
      if (xhr.status == 200) {
        var data = JSON.parse(xhr.responseText);
        mostrarConfirmadosPorRegion(data);
        btnConfirmadosRegion.style.display = 'none';
      } else {
        console.error('Error en la red:', xhr.statusText);
      }
    };
    xhr.onerror = function() {
      console.error('Error en la red.');
```

■ Página

**Número Total de Confirmados por
Región**

Mostrar
Confirmados

■ Resultado



5.3. Encuentre las 10 regiones cuya suma total sea la mayor

■ HTML


```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="style.css">
  <title>Ejercicio3</title>
</head>
<body>
  <div id="datos">
    <h1>LISTA DE 10 REGIONES CON LA MAYOR SUMA TOTAL</h1>
    <button id="btnMostrarRegiones">Mostrar regiones</button>
    <div class="contenedor">
      <ul id="regionList"></ul>
    </div>
  </div>
  <script src="ejercicio3.js"></script>
</body>
</html>
```

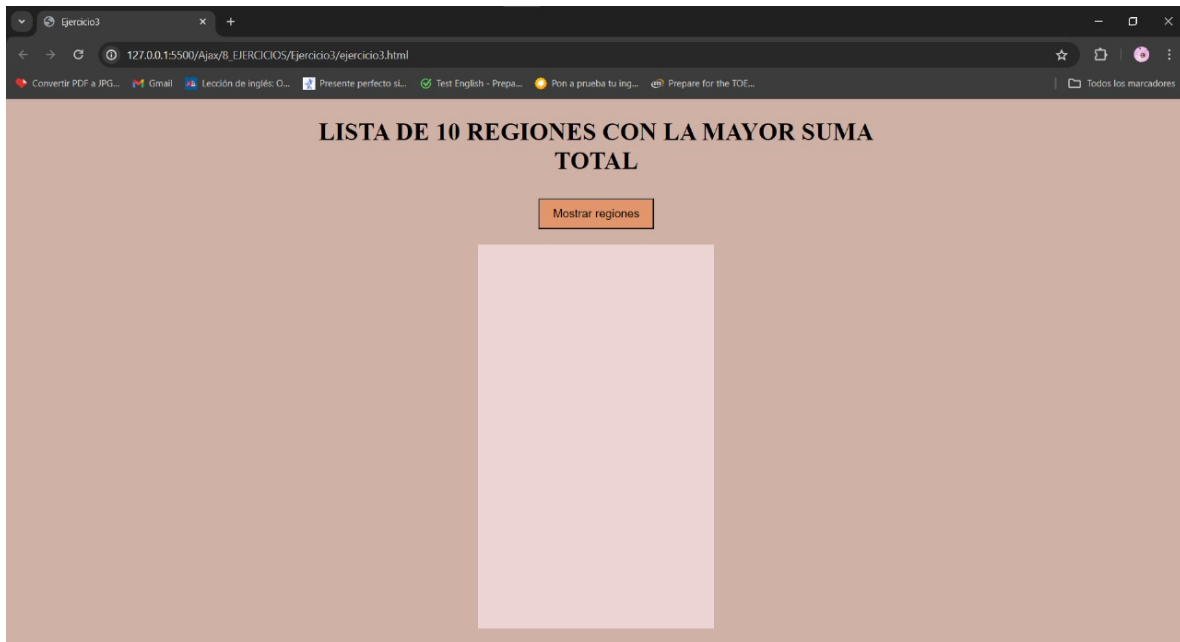
- Script

```

1 document.addEventListener("DOMContentLoaded", () => {
2     const btnMostrarRegiones = document.getElementById('btnMostrarRegiones');
3
4     btnMostrarRegiones.addEventListener('click', () => {
5         fetch('../data.json')
6             .then(response => {
7                 if (!response.ok) {
8                     throw new Error('Error en la red');
9                 }
10                return response.json();
11            })
12            .then(data => {
13                mostrarRegiones(data);
14                btnMostrarRegiones.style.display = 'none';
15            })
16            .catch(error => {
17                console.error('Error:', error);
18            });
19    });
20 });
21
22 function mostrarRegiones(data) {
23     const regionList = document.getElementById('regionList');
24     const casosConfirmadosPorRegion = {};
25
26     data.forEach(regionData => {
27         const region = regionData.region;
28         let totalCasosConfirmados = 0;
29         regionData.confirmed.forEach(caso => {
30             totalCasosConfirmados += parseInt(caso.value);
31         });
32         casosConfirmadosPorRegion[region] = totalCasosConfirmados;
33     });
34
35     while(Object.keys(casosConfirmadosPorRegion).length > 10){
36         let menorCantidad = Infinity;
37         let regionMenorCasos = null;
38         for (let region in casosConfirmadosPorRegion) {
39             const cantidadCasos = casosConfirmadosPorRegion[region];
40             if (cantidadCasos < menorCantidad) {
41                 menorCantidad = cantidadCasos;
42                 regionMenorCasos = region;
43             }
44         }
45         delete casosConfirmadosPorRegion[regionMenorCasos];
46     }
47
48     for (let region in casosConfirmadosPorRegion) {
49         const li = document.createElement('li');
50         li.textContent = `${region}: ${casosConfirmadosPorRegion[region]}`;
51         regionList.appendChild(li);
52     }
53
54     const totalLi = document.createElement('li');
55     totalLi.style.marginTop = '20px';
56     totalLi.textContent = `Total de casos: ${contarTotalCasos(casosConfirmadosPorRegion)}`;
57     regionList.appendChild(totalLi);
58 }
59
60 function contarTotalCasos(casosConfirmadosPorRegion) {
61     let totalCasos = 0;
62     for (let region in casosConfirmadosPorRegion) {
63         const casosRegion = casosConfirmadosPorRegion[region];
64         totalCasos += casosRegion;
65     }
66     return totalCasos;
67 }

```

■ Página



■ Resultado



5.4. Visualice un gráfico en el tiempo de los valores para la región de Arequipa

■ HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Confirmados en la región de Arequipa</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div id="datos">
    <h1>Gráfico Confirmados en Arequipa</h1>
    <button id="mostrarGrafico">Mostrar Grafico</button>
    <div class="contenedor">
      <canvas id="Grafico" width="400" height="200"></canvas>
    </div>

    <script src="ejercicio4.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
  </div>
</body>
</html>
```

- Script

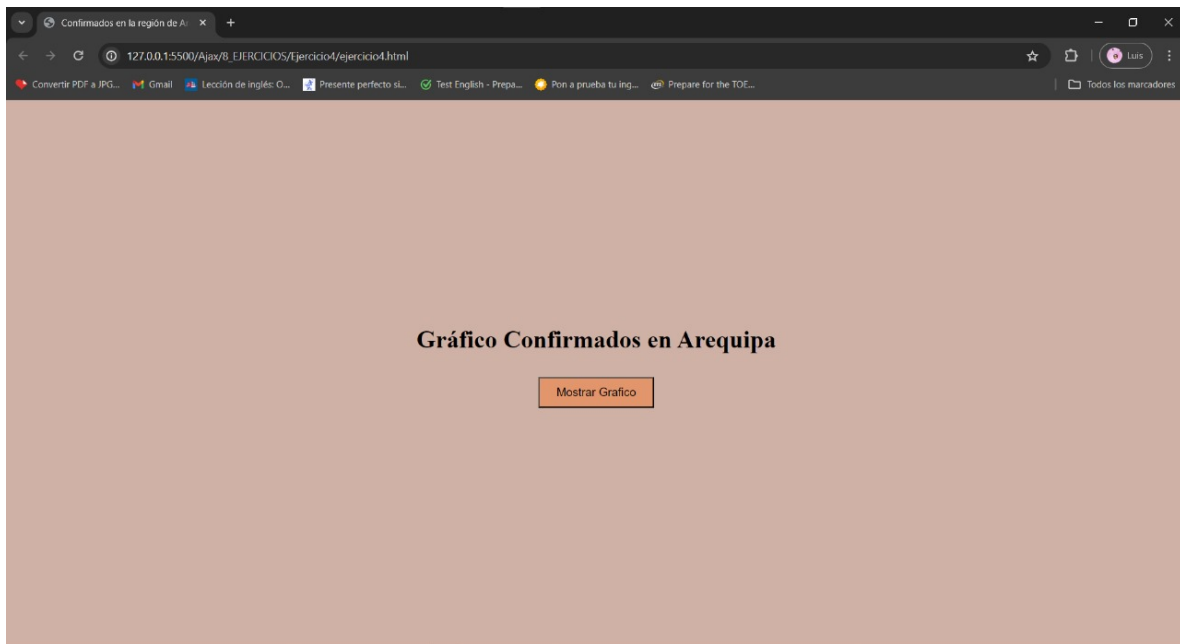
```
document.addEventListener("DOMContentLoaded", function(){
    const btnMostrarGrafico = document.getElementById('mostrarGrafico');
    const contenedor = document.querySelector('.contenedor');

    btnMostrarGrafico.addEventListener('click', () => {
        let xhr = new XMLHttpRequest();
        xhr.open("GET", "../data.json", true);
        xhr.onload = function() {
            if (xhr.status === 200) {
                let data = JSON.parse(xhr.responseText);
                let arequipaData = data.find(region => region.region === "Arequipa");
                if (arequipaData) {
                    mostrarGrafico(arequipaData.confirmed);
                    contenedor.style.display = 'flex';
                    btnMostrarGrafico.style.display = 'none';
                } else {
                    console.error("No se encontraron datos de Arequipa");
                }
            } else {
                console.error('Error en la red:', xhr.statusText);
            }
        };
        xhr.onerror = function() {
            console.error('Error en la red');
        };
        xhr.send();
    });
});

function mostrarGrafico(arequipaData) {
    let dates = arequipaData.map(entry => entry.date);
    let values = arequipaData.map(entry => parseInt(entry.value));

    let ctx = document.getElementById('Grafico').getContext('2d');
    let graphic = new Chart(ctx, {
        type: 'line',
        data: {
            labels: dates,
            datasets: [{
                label: 'Numero de Confirmados',
                data: values,
                borderColor: 'rgba(75, 192, 192, 1)',
                backgroundColor: 'rgba(75, 192, 192, 0.2)',
                fill: true,
            }]
        }
    });
}
```

■ Página



■ Resultado



5.5. Haga gráficos comparativos entre regiones usando líneas

■ HTML


```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ejercicio5</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div id="datos">
    <h1>Gráfico Confirmados en Regiones del Perú</h1>
    <button id="mostrarGrafico">Mostrar Graficos</button>
    <div class="contenedor"></div>
  </div>

  <script src="ejercicio5.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</body>
</html>
```

- Script

```

document.addEventListener("DOMContentLoaded", function() {
  const btnMostrarGrafico = document.getElementById('mostrarGrafico');
  const contenedor = document.querySelector('.contenedor');

  btnMostrarGrafico.addEventListener('click', () => {
    let xhr = new XMLHttpRequest();
    xhr.open("GET", "../data.json", true);
    xhr.onload = function() {
      if (xhr.status === 200) {
        let data = JSON.parse(xhr.responseText);
        if (data.length > 0) {
          mostrarGraficos(data);
          contenedor.style.display = 'flex';
          btnMostrarGrafico.style.display = 'none';
        } else {
          console.error("No se encontraron datos de regiones");
        }
      } else {
        console.error('Error en la red:', xhr.statusText);
      }
    };
    xhr.onerror = function() {
      console.error('Error en la red');
    };
    xhr.send();
  });
});

function mostrarGraficos(data) {
  const contenedor = document.querySelector('.contenedor');
  contenedor.innerHTML = ''; // Clear any existing content

  data.forEach(regionData => {
    const canvasContainer = document.createElement('div');
    canvasContainer.classList.add('canvas-container');
    const canvas = document.createElement('canvas');
    canvas.id = `Grafico-${regionData.region}`;
    canvasContainer.appendChild(canvas);
    contenedor.appendChild(canvasContainer);

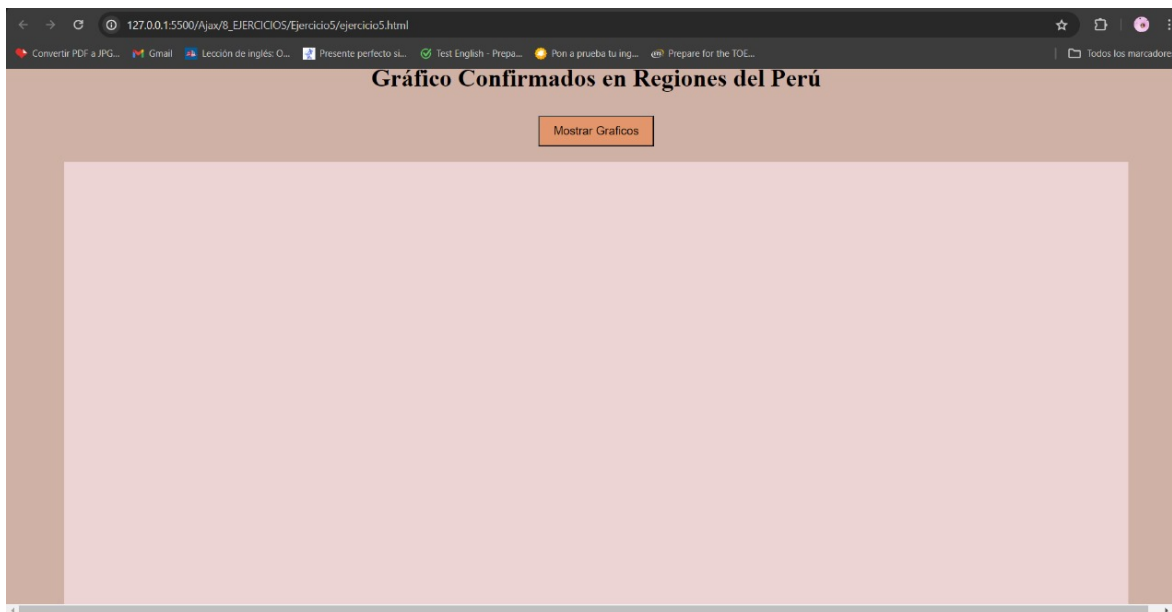
    let dates = regionData.confirmed.map(entry => entry.date);
    let values = regionData.confirmed.map(entry => parseInt(entry.value));

    let ctx = canvas.getContext('2d');
    new Chart(ctx, {
      type: 'line',
      data: {
        labels: dates,
        datasets: [{
          label: `Número de Confirmados en ${regionData.region}`,
          data: values,
          borderColor: 'rgba(75, 192, 192, 1)',
          backgroundColor: 'rgba(75, 192, 192, 0.2)',
          fill: true,
        }]
      }
    });
  });
}

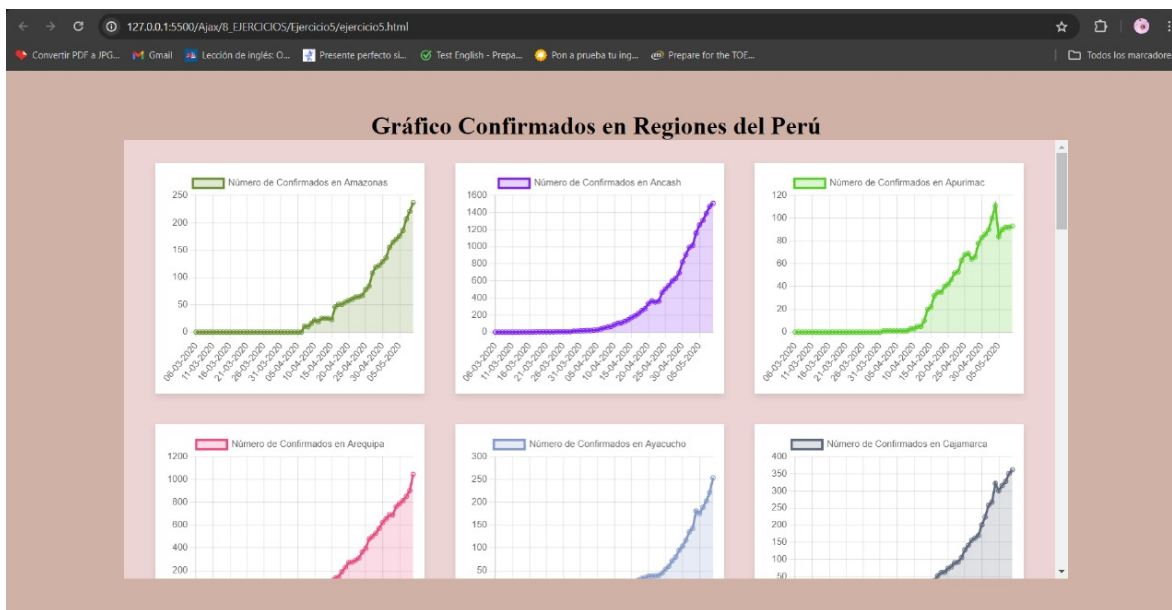
function getRandomColor(alpha = 1) {
  const r = Math.floor(Math.random() * 255);
  const g = Math.floor(Math.random() * 255);
  const b = Math.floor(Math.random() * 255);
  return `rgba(${r}, ${g}, ${b}, ${alpha})`;
}

```


■ Página



■ Resultado



5.6. Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao

■ HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Comparativa de Regiones</title>
  <link rel="stylesheet" href="style.css">
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
  <div id="datos">
    <h1>Grafico Comparativo de Regiones (exepcto Lima y Callao)</h1>
    <button id="mostrarGrafico">Mostrar Grafico</button>
    <div class="contenedor" style="display: none;">
      <canvas id="GraficoComparativo" width="800" height="400"></canvas>
    </div>
  </div>

  <script src="ejercicio6.js"></script>
</body>
</html>
```

- Script

```
document.addEventListener("DOMContentLoaded", function() {
  const btnMostrarGrafico = document.getElementById('mostrarGrafico');
  const contenedor = document.querySelector('.contenedor');

  btnMostrarGrafico.addEventListener('click', () => {
    let xhr = new XMLHttpRequest();
    xhr.open("GET", "../data.json", true);
    xhr.onload = function() {
      if (xhr.status === 200) {
        let data = JSON.parse(xhr.responseText);
        let regions = data.filter(region => region.region !== "Lima" && region.region !== "Callao");
        if (regions.length > 0) {
          mostrarGraficoComparativo(regions);
          contenedor.style.display = 'flex';
          btnMostrarGrafico.style.display = 'none';
        } else {
          console.error("No se encontraron datos de regiones");
        }
      } else {
        console.error("Error en la red:", xhr.statusText);
      }
    };
    xhr.onerror = function() {
      console.error("Error en la red");
    };
    xhr.send();
  });
});

function mostrarGraficoComparativo(regions) {
  let labels = [];
  let datasets = [];

  regions.forEach(region => {
    let dates = region.confirmed.map(entry => entry.date);
    let values = region.confirmed.map(entry => parseInt(entry.value));

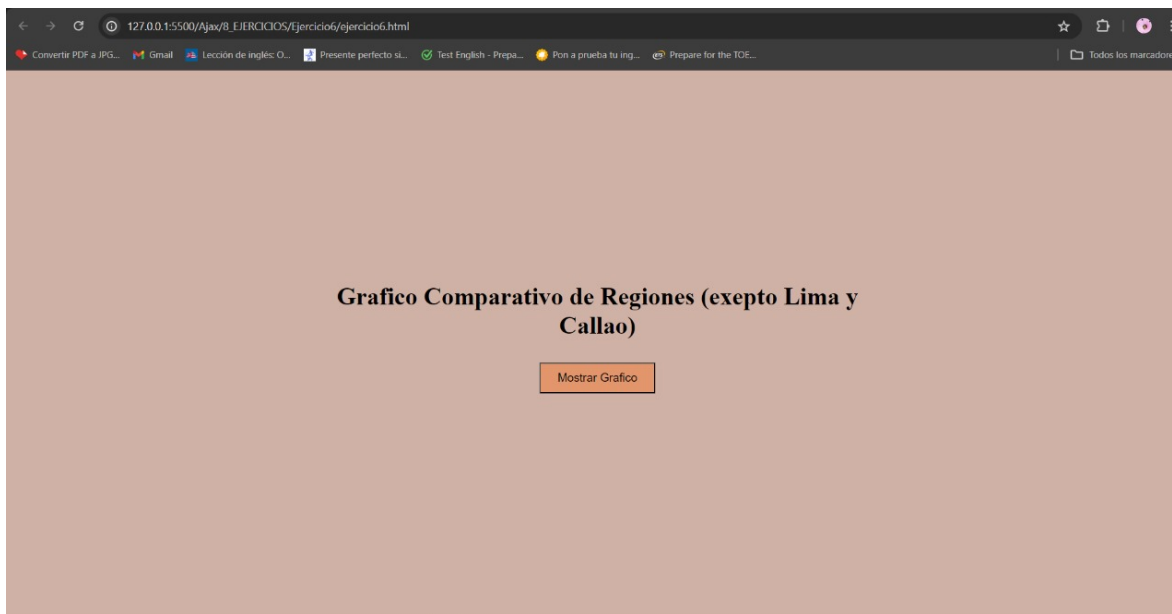
    if (labels.length === 0) {
      labels = dates;
    }

    datasets.push({
      label: region.region,
      data: values,
      borderColor: getRandomColor(),
      backgroundColor: getRandomColor(0.2),
      fill: false,
    });
  });

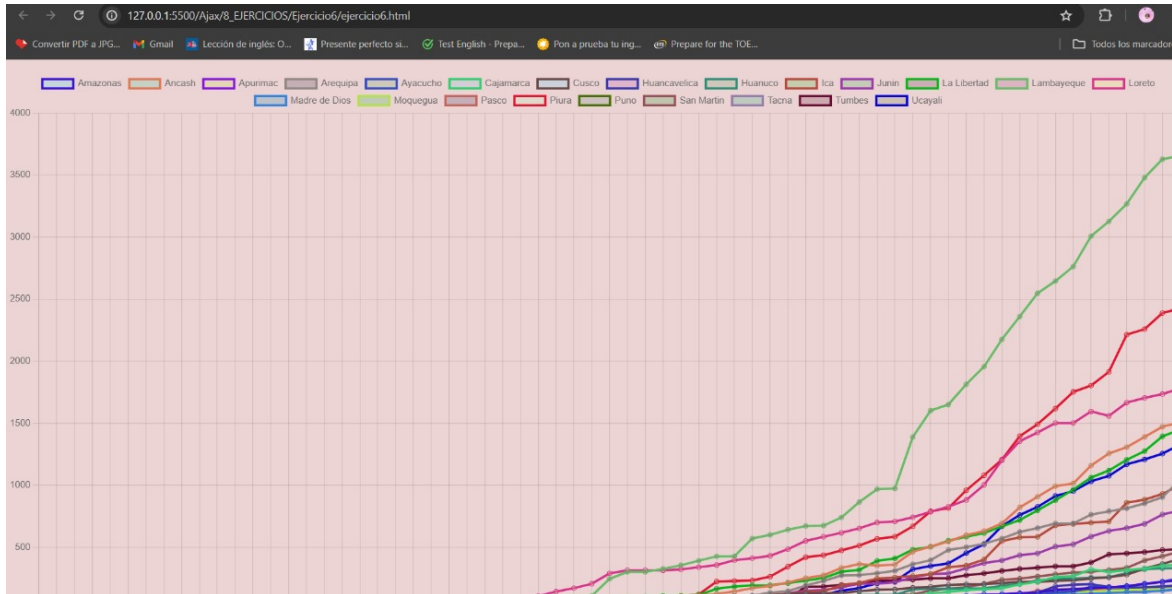
  let ctx = document.getElementById("GraficoComparativo").getContext('2d');
  new Chart(ctx, {
    type: 'line',
    data: {
      labels: labels,
      datasets: datasets
    }
  });
}

function getRandomColor(alpha = 1) {
  const r = Math.floor(Math.random() * 255);
  const g = Math.floor(Math.random() * 255);
  const b = Math.floor(Math.random() * 255);
  return `rgba(${r}, ${g}, ${b}, ${alpha})`;
}
```

■ Página



■ Resultado



5.7. Haga gráficos comparativos entre regiones elegidas por el usuario.

■ HTML

```

@ ajax > Ajax > 8. EJERCICIOS > Ejercicio7 > ejercicio7.html > html > body > div#datos > div.opciones > br
2  <html lang="en">
9  <body>
10  <div id="datos">
12    <div class="opciones">
16      <input type="checkbox" id="Ancash" name="opcion" value="Ancash">
17      <label for="Ancash">Ancash</label><br>
18
19      <input type="checkbox" id="Aurimac" name="opcion" value="Aurimac">
20      <label for="Aurimac">Aurimac</label><br>
21
22      <input type="checkbox" id="Arequipa" name="opcion" value="Arequipa">
23      <label for="Arequipa">Arequipa</label><br>
24
25      <input type="checkbox" id="Ayacucho" name="opcion" value="Ayacucho">
26      <label for="Ayacucho">Ayacucho</label><br>
27
28      <input type="checkbox" id="Cajamarca" name="opcion" value="Cajamarca">
29      <label for="Cajamarca">Cajamarca</label><br>
30
31      <input type="checkbox" id="Callao" name="opcion" value="Callao">
32      <label for="Callao">Callao</label><br>
33
34      <input type="checkbox" id="Huancavelica" name="opcion" value="Huancavelica">
35      <label for="Huancavelica">Huancavelica</label><br>
36
37      <input type="checkbox" id="Huanuco" name="opcion" value="Huanuco">
38      <label for="Huanuco">Huanuco</label><br>
39
40      <input type="checkbox" id="Ica" name="opcion" value="Ica">
41      <label for="Ica">Ica</label><br>
42
43      <input type="checkbox" id="Junin" name="opcion" value="Junin">
44      <label for="Junin">Junin</label><br>
45
46      <input type="checkbox" id="LaLibertad" name="opcion" value="LaLibertad">
47      <label for="LaLibertad">La Libertad</label><br>
48
49      <input type="checkbox" id="Lambayeque" name="opcion" value="Lambayeque">
50      <label for="Lambayeque">Lambayeque</label><br>
51
52      <input type="checkbox" id="Lima" name="opcion" value="Lima">
53      <label for="Lima">Lima</label><br>
54
55      <input type="checkbox" id="Loreto" name="opcion" value="Loreto">
56      <label for="Loreto">Loreto</label><br>
57
58      <input type="checkbox" id="MadreDeDios" name="opcion" value="MadreDeDios">
59      <label for="MadreDeDios">Madre de Dios</label><br>
60
61      <input type="checkbox" id="Moquegua" name="opcion" value="Moquegua">
62      <label for="Moquegua">Moquegua</label><br>
63
64      <input type="checkbox" id="Pasco" name="opcion" value="Pasco">
65      <label for="Pasco">Pasco</label><br>
66
67      <input type="checkbox" id="Piura" name="opcion" value="Piura">
68      <label for="Piura">Piura</label><br>
69
70      <input type="checkbox" id="Puno" name="opcion" value="Puno">
71      <label for="Puno">Puno</label><br>
72
73      <input type="checkbox" id="SanMartin" name="opcion" value="SanMartin">
74      <label for="SanMartin">San Martin</label><br>
75
76      <input type="checkbox" id="Tacna" name="opcion" value="Tacna">
77      <label for="Tacna">Tacna</label><br>
78
79      <input type="checkbox" id="Tumbes" name="opcion" value="Tumbes">
80      <label for="Tumbes">Tumbes</label><br>
81
82      <input type="checkbox" id="Ucayali" name="opcion" value="Ucayali">
83      <label for="Ucayali">Ucayali</label><br>
84    </div>
85
86    <button id="mostrarGrafico">Mostrar Graficos</button>
87    <div class="contenedor"></div>
88  </div>
89
90  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
91  <script src="ejercicio7.js"></script>
92 </body>
93 </html>

```

- Script


```

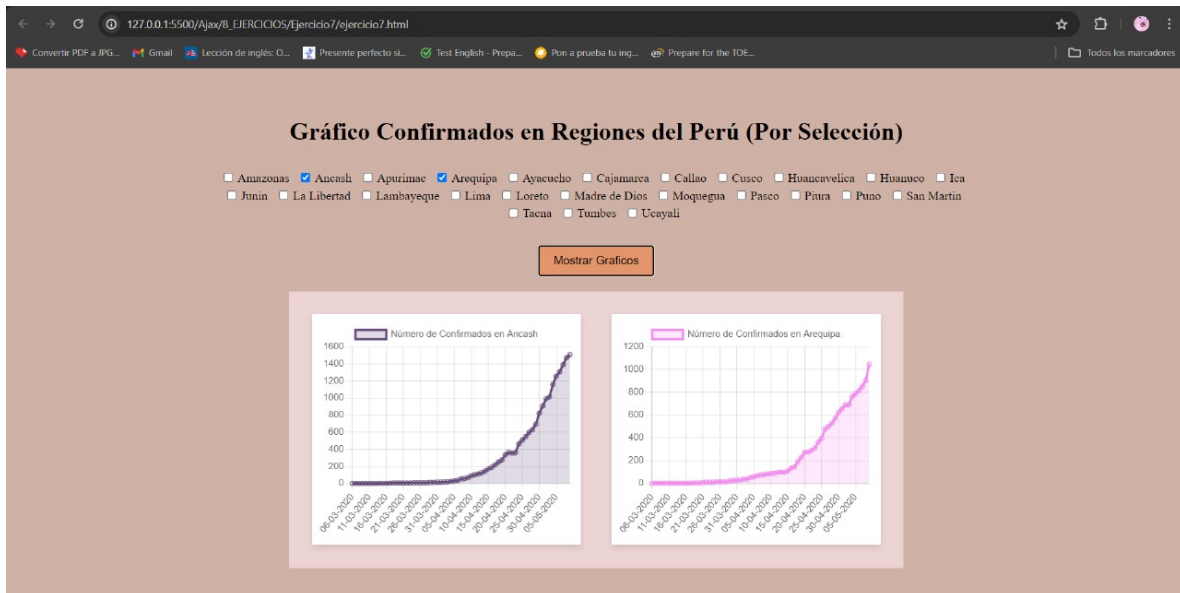
1 document.addEventListener("DOMContentLoaded", function() {
2     const btnMostrarGrafico = document.getElementById('mostrarGrafico');
3     const contenedor = document.querySelector('.contenedor');
4
5     btnMostrarGrafico.addEventListener('click', () => {
6         let xhr = new XMLHttpRequest();
7         xhr.open("GET", "../data.json", true);
8         xhr.onload = function() {
9             if (xhr.status === 200) {
10                 let data = JSON.parse(xhr.responseText);
11                 if (data.length > 0) {
12                     mostrarGraficos(data);
13                     contenedor.style.display = 'flex';
14                     btnMostrarGrafico.style.display = 'none';
15                 } else {
16                     console.error("No se encontraron datos de regiones");
17                 }
18             } else {
19                 console.error("Error en la red:", xhr.statusText);
20             }
21         };
22         xhr.onerror = function() {
23             console.error("Error en la red");
24         };
25         xhr.send();
26     });
27 });
28
29 function mostrarGraficos(data) {
30     const contenedor = document.querySelector('.contenedor');
31     contenedor.innerHTML = '';
32
33     data.forEach(regionData => {
34         const checkboxId = regionData.region.replace(/\s+/g, '');
35         const checkbox = document.getElementById(checkboxId);
36         if (checkbox.checked) {
37             const canvasContainer = document.createElement('div');
38             canvasContainer.classList.add('canvas-container');
39             const canvas = document.createElement('canvas');
40             canvas.id = `Grafico-${regionData.region}`;
41             canvasContainer.appendChild(canvas);
42             contenedor.appendChild(canvasContainer);
43
44             let dates = regionData.confirmed.map(entry => entry.date);
45             let values = regionData.confirmed.map(entry => parseInt(entry.value));
46             let colors = generarRandomColor();
47
48             let ctx = canvas.getContext('2d');
49             new Chart(ctx, {
50                 type: 'line',
51                 data: {
52                     labels: dates,
53                     datasets: [{
54                         label: `Número de Confirmados en ${regionData.region}`,
55                         data: values,
56                         borderColor: colors.solid,
57                         backgroundColor: colors.translucent,
58                         fill: true,
59                     }]
60                 },
61                 options: {
62                     responsive: true,
63                     maintainAspectRatio: false
64                 }
65             });
66         }
67     });
68 }
69
70 function generarRandomColor() {
71     let r = Math.floor(Math.random() * 256);
72     let g = Math.floor(Math.random() * 256);
73     let b = Math.floor(Math.random() * 256);
74
75     return {
76         solid: `rgb(${r},${g},${b},1)`,
77         translucent: `rgba(${r},${g},${b},0.2)`
78     };
79 }

```

■ Página



■ Resultado



5.8. Visualice un gráfico comparativo del crecimiento en regiones excepto Lima y Callao, mostrando el número de confirmados por cada día

■ 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>Comparativa de Regiones</title>
7   <link rel="stylesheet" href="style.css">
8   <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
9 </head>
10 <body>
11   <div id="datos">
12     <h1>Grafico Comparativo de Regiones (excepto Lima y Callao)</h1>
13     <button id="mostrarGrafico">Mostrar Grafico</button>
14     <div class="contenedor" style="display: none;">
15       <canvas id="GraficoComparativo" width="800" height="400"></canvas>
16     </div>
17   </div>
18
19   <script src="ejercicio8.js"></script>
20 </body>
21 </html>
```

- Script

```

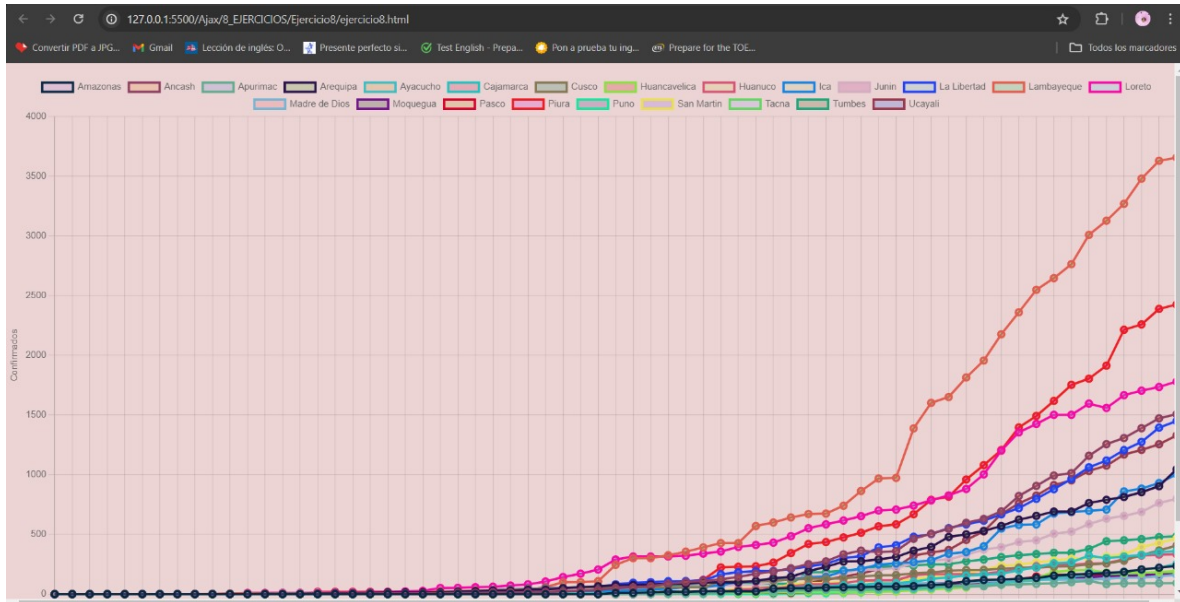
1 document.addEventListener('DOMContentLoaded', function() {
2     const btnMostrarGrafico = document.getElementById('mostrarGrafico');
3     const contenedor = document.querySelector('.contenedor');
4
5     btnMostrarGrafico.addEventListener('click', () => {
6         let xhr = new XMLHttpRequest();
7         xhr.open("GET", "../data.json", true);
8         xhr.onload = function() {
9             if (xhr.status === 200) {
10                 let data = JSON.parse(xhr.responseText);
11                 let regions = data.filter(region => region.region !== "Lima" && region.region !== "Callao");
12                 if (regions.length > 0) {
13                     mostrarGraficoComparativo(regions);
14                     contenedor.style.display = "flex";
15                     btnMostrarGrafico.style.display = "none";
16                 } else {
17                     console.error("No se encontraron datos de regiones");
18                 }
19             } else {
20                 console.error('Error en la red:', xhr.statusText);
21             }
22         };
23         xhr.onerror = function() {
24             console.error('Error en la red');
25         };
26         xhr.send();
27     });
28 })
29
30 function mostrarGraficoComparativo(regions) {
31     let labels = [];
32     let datasets = [];
33
34     regions.forEach((region, index) => {
35         let dates = region.confirmed.map(entry => entry.date);
36         let values = region.confirmed.map(entry => parseInt(entry.value));
37
38         if (labels.length === 0) {
39             labels = dates;
40         }
41
42         datasets.push({
43             label: region.region,
44             data: values,
45             borderColor: getRandomColor(),
46             backgroundColor: getRandomColor(0.2),
47             fill: false,
48             pointRadius: 4,
49             borderWidth: 2,
50         });
51     });
52
53     let ctx = document.getElementById('GraficoComparativo').getContext('2d');
54     new Chart(ctx, {
55         type: 'line',
56         data: {
57             labels: labels,
58             datasets: datasets
59         },
60         options: {
61             scales: {
62                 x: {
63                     display: true,
64                     title: {
65                         display: true,
66                         text: 'Fecha'
67                     },
68                 },
69                 y: {
70                     display: true,
71                     title: {
72                         display: true,
73                         text: 'Confirmados'
74                     },
75                     suggestedMin: 0,
76                 }
77             },
78             title: {
79                 display: true,
80                 text: 'Crecimiento de Confirmados en Regiones (excepto Lima y Callao)',
81                 fontSize: 18,
82                 fontColor: '#333'
83             },
84             legend: {
85                 display: true,
86                 position: 'bottom',
87                 labels: {
88                     fontColor: '#333'
89                 }
90             },
91             plugins: {
92                 legend: {
93                     display: true,
94                     position: 'top'
95                 },
96                 tooltip: {
97                     enabled: true,
98                     mode: 'index', // Se muestra todos los datos del día a la vez
99                 }
100             }
101         }
102     });
103 }

```

■ Página

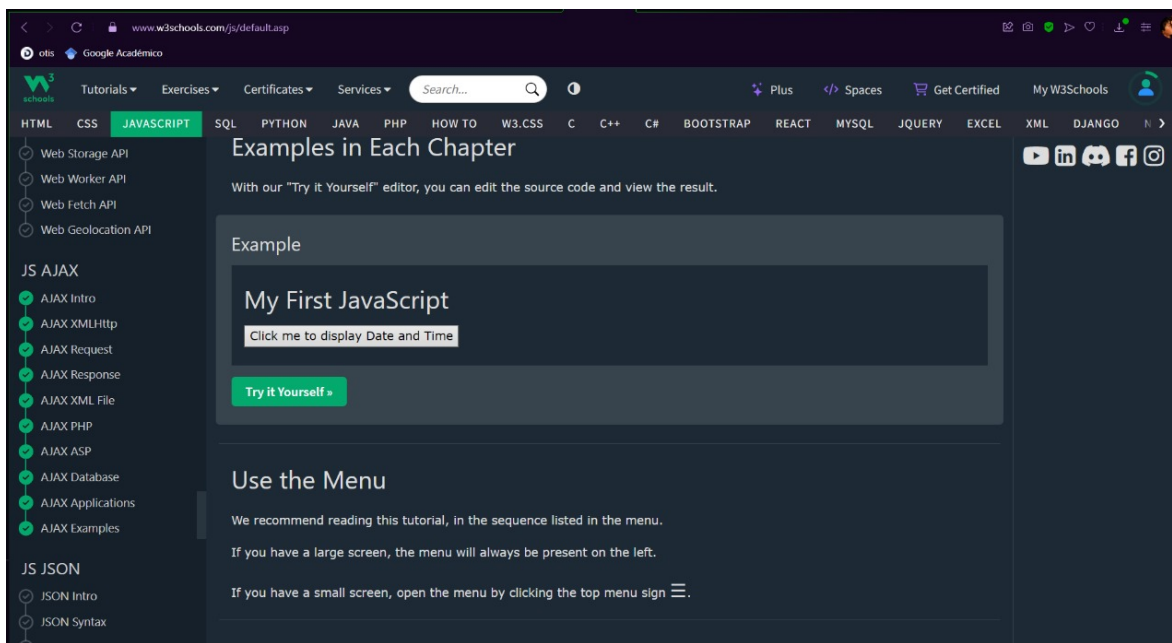


■ Resultado



6. w3schools

6.1. Luis Guillermo Luque Condori



6.2. Fernando Miguel Garambel Marín

6.3. William Herderson Choquehuanca Berma

6.4. Jeans Anthony Ajra Huacso

7. Rúbricas

7.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

7.2. 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	2	
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	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1	
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	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	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	1	
Total		20		13	

8. Referencias

- <https://www.w3schools.com/java/default.asp>
- <https://www.geeksforgeeks.org/insertion-sort/>