

**Universidad Nacional de San Agustín**  
**Facultad de Ingeniería**  
**Escuela Profesional de Ingeniería de**  
**Sistemas**



**Curso: Introducción al Desarrollo Web**  
**Laboratorio**

**Grupo: “D”**

**Docente: Carlo Jose Luis Corrales**  
**Delgado**

**Título de la Actividad: Laboratorio 13 –**  
**Javascript: Estructura de Datos**

**Estudiante: Alessandro Josue Justo**  
**Vilca**

**2025**

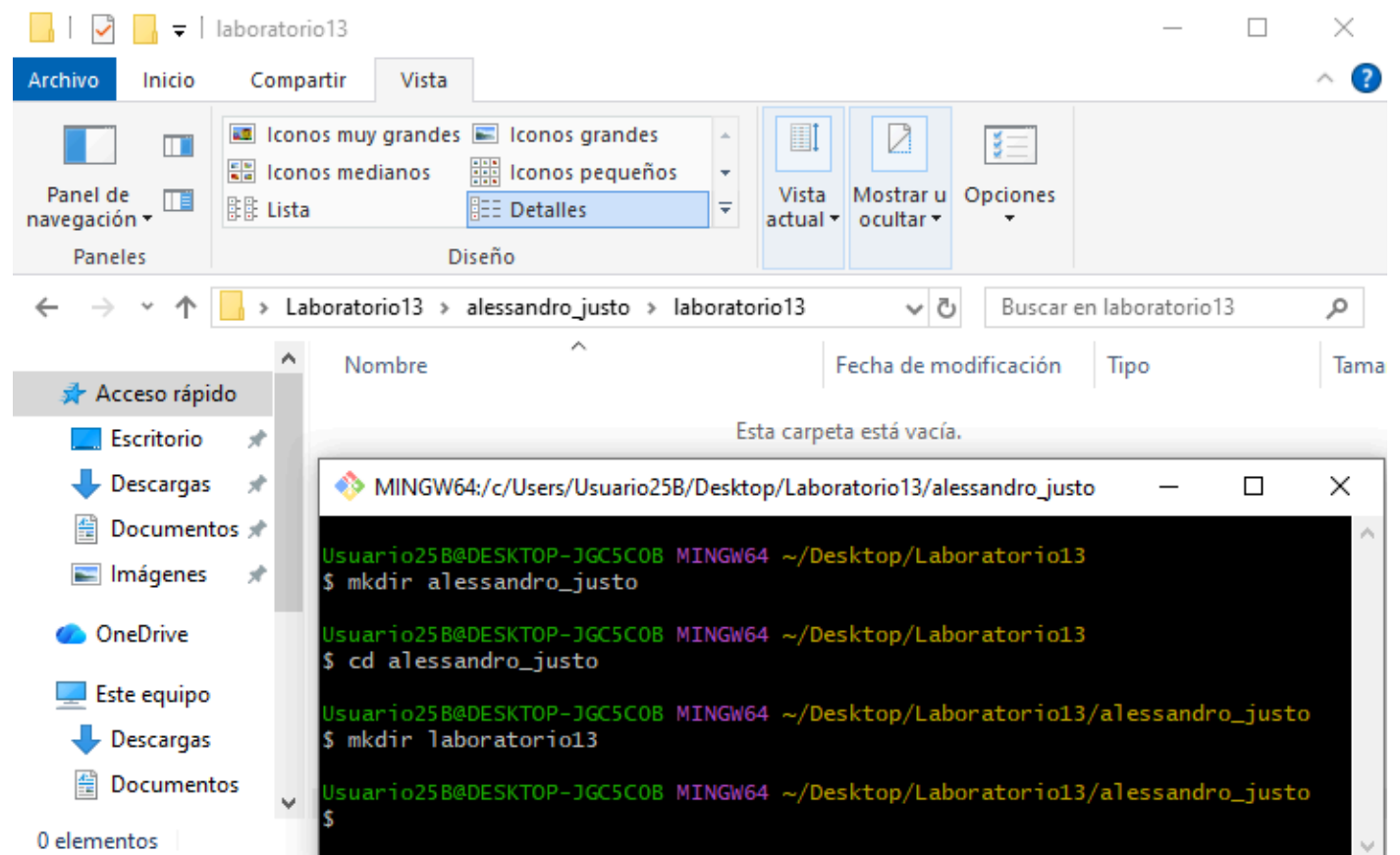
1. **Crear un directorio que tenga su nombre y un subdirectorio laboratorio13. Recomendación: usar minúsculas, sin espacios, sin tildes ni “ñ” y con guiones medios o bajos**

// Laboratorio Nro 13 - Ejercicio1

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 1min



2. **Utilizar los atajos de teclado o combinación de teclas para agilizar su trabajo**

// Laboratorio Nro 13 - Ejercicio2

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 1min

**Atajos de teclado que use y tuve en cuenta en VS Code:**

Ctrl + Ñ: abre la terminal integrada.

Ctrl + /: comentar línea seleccionada.

Alt + ↑: mover línea hacia arriba.

Shift + Alt + ↓: duplicar línea.

**Atajos de teclado útiles (recomendados)**

Ctrl + S → guardar

Ctrl + C / Ctrl + V → copiar / pegar

Ctrl + Z / Ctrl + Y → deshacer / rehacer

Ctrl + F → buscar texto en archivo

Ctrl + Shift + F → buscar en todo el proyecto

Ctrl + P → abrir archivo por nombre (VSCode)

Ctrl + ` → abrir terminal integrado (VSCode)

Ctrl + Shift + T → reabrir pestaña cerrada (navegador)

Alt + Tab → cambiar de aplicación

3. **Crea una función doblarNumeros que reciba un arreglo de números y devuelva uno nuevo con cada número duplicado**

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 17min

**HTML Ejecutor**

index.html &gt; html &gt; body &gt; main

```
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    <title>Laboratorio 13</title>
7  </head>
8  <body>
9    <main>
10     <h1>Laboratorio 13</h1>
11
12     <section id="ejercicio03">
13       <h2>3. Función que dobla números</h2>
14       <button id="run-ej3">Ejecutar ejercicio 3</button>
15       <pre id="out-ej3"></pre>
16       <script src="js/ejercicio03.js"></script>
17     </section>
18
19     <section id="ejercicio04">
20       <h2>4. Filtrar y transformar arreglo</h2>
21       <button id="run-ej4">Ejecutar ejercicio 4</button>
22       <pre id="out-ej4"></pre>
23       <script src="js/ejercicio04.js"></script>
24     </section>
25
26     <section id="ejercicio05">
27       <h2>5. Reordenar palabras alfabéticamente</h2>
28       <button id="run-ej5">Ejecutar ejercicio 5</button>
29       <pre id="out-ej5"></pre>
30       <script src="js/ejercicio05.js"></script>
31     </section>
32
33     <section id="ejercicio06">
34       <h2>6. Set con nombres (sin duplicados)</h2>
35       <button id="run-ej6">Ejecutar ejercicio 6</button>
36       <pre id="out-ej6"></pre>
37       <script src="js/ejercicio06.js"></script>
38     </section>
39
40     <section id="ejercicio07">
41       <h2>7. Verificar si hay duplicados</h2>
42       <button id="run-ej7">Ejecutar ejercicio 7</button>
43       <pre id="out-ej7"></pre>
44       <script src="js/ejercicio07.js"></script>
45     </section>
46
47     <section id="ejercicio08">
48       <h2>8. Calcular precio total con Map</h2>
49       <button id="run-ej8">Ejecutar ejercicio 8</button>
50       <pre id="out-ej8"></pre>
51       <script src="js/ejercicio08.js"></script>
52     </section>
53
```

```
53
54 <section id="ejercicio09">
55   <h2>9. Contar palabras con Map</h2>
56   <button id="run-ej9">Ejecutar ejercicio 9</button>
57   <pre id="out-ej9"></pre>
58   <script src="js/ejercicio09.js"></script>
59 </section>
60
61 <section id="ejercicio10">
62   <h2>10. Invertir claves y valores de un Map</h2>
63   <button id="run-ej10">Ejecutar ejercicio 10</button>
64   <pre id="out-ej10"></pre>
65   <script src="js/ejercicio10.js"></script>
66 </section>
67
68 <section id="ejercicio11">
69   <h2>11. Objeto auto con método detalles</h2>
70   <button id="run-ej11">Ejecutar ejercicio 11</button>
71   <pre id="out-ej11"></pre>
72   <script src="js/ejercicio11.js"></script>
73 </section>
74
75 <section id="ejercicio12">
76   <h2>12. Contar letras en texto</h2>
77   <button id="run-ej12">Ejecutar ejercicio 12</button>
78   <pre id="out-ej12"></pre>
79   <script src="js/ejercicio12.js"></script>
80 </section>
81
82 <section id="ejercicio13">
83   <h2>13. Combinar catálogos de tiendas</h2>
84   <button id="run-ej13">Ejecutar ejercicio 13</button>
85   <pre id="out-ej13"></pre>
86   <script src="js/ejercicio13.js"></script>
87 </section>
88
89 <section id="ejercicio14">
90   <h2>14. Gestionar empleados por área</h2>
91   <button id="run-ej14">Ejecutar ejercicio 14</button>
92   <pre id="out-ej14"></pre>
93   <script src="js/ejercicio14.js"></script>
94 </section>
95
96 </main>
97 </body>
98 </html>
```

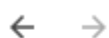
<> index.html

JS ejercicio03.js X

js > JS ejercicio03.js > ...

```
1 document.addEventListener('DOMContentLoaded', () => {
2   const btn = document.getElementById('run-ej3');
3   const out = document.getElementById('out-ej3');
4
5   function doblarNumeros(arr) {
6     return arr.map(num => num * 2);
7   }
8
9   btn.addEventListener('click', () => {
10    const entrada = prompt("Ingrese números separados por comas (ejemplo: 2,4,6):");
11    if (!entrada) {
12      out.textContent = "Debe ingresar al menos un número.";
13      return;
14    }
15
16    const numeros = entrada.split(',').map(Number);
17    if (numeros.some(isNaN)) {
18      out.textContent = "Ingrese solo números válidos.";
19      return;
20    }
21
22    const resultado = doblarNumeros(numeros);
23    out.textContent = `Números originales: [${numeros}] \nNúmeros doblados: [${resultado}]`;
24  });
25 });
```

▼ Laboratorio 13 X + - □ X



📁 Archivo

C:/Users/aless/OneDrive/Desktop/La...



Lab

Esta página dice

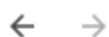
Ingrese números separados por comas (ejemplo: 2,4,6):

3,4,5|

Aceptar

Cancelar

▼ Laboratorio 13 X + - □ X



📁 Archivo

C:/Users/aless/OneDrive/Desktop/La...



# Laboratorio 13

## 3. Función que dobla números

Ejecutar ejercicio 3

Números originales: [3,5,7]

Números doblados: [6,10,14]

4. Crea una función `filtrarYTransformar(arr)` que reciba un arreglo de números y:

- Elimine los valores negativos
- Eleve los restantes al cuadrado
- Devuelva la suma total de esos cuadrados. `filtrarYTransformar([2, -3, 4, -1, 5]);` //  $2^2 + 4^2 + 5^2 = 45$

// Laboratorio Nro 13 - Ejercicio4

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 16min

```
index.html JS ejercicio04.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2     const btn = document.getElementById('run-ej4');
3     const out = document.getElementById('out-ej4');
4
5     function filtrarYTransformar(arr) {
6         const positivos = arr.filter(num => num >= 0);
7         const cuadrados = positivos.map(num => num * num);
8         const suma = cuadrados.reduce((acc, val) => acc + val, 0);
9         return suma;
10    }
11
12    btn.addEventListener('click', () => {
13        const ejemplo = [2, -3, 4, -1, 5];
14        const resultado = filtrarYTransformar(ejemplo);
15        out.textContent = `Arreglo original: [${ejemplo}]\nResultado: ${resultado}`;
16    });
17 });
```

## 4. Filtrar y transformar arreglo

Ejecutar ejercicio 4

Arreglo original: [2,-3,4,-1,5]

Resultado: 45

5. Crea una función `reordenarPalabras(oracion)` que:

- Reciba una cadena con palabras separadas por espacios
  - Devuelva un arreglo ordenado alfabéticamente, pero con las palabras en mayúsculas
- `reordenarPalabras("sol luna estrella planeta");` // `["ESTRELLA", "LUNA", "PLANETA", "SOL"]`

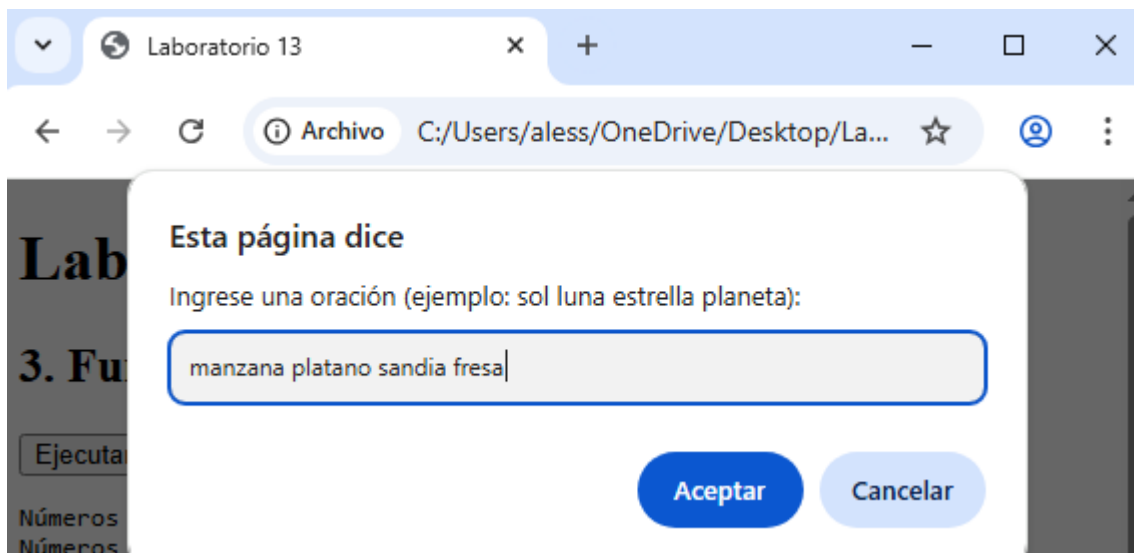
// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 15min

```
index.html x JS ejercicio05.js x JS ejercicio04.js JS ejercicio03.js
js > JS ejercicio05.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2   const btn = document.getElementById('run-ej5');
3   const out = document.getElementById('out-ej5');
4
5   function reordenarPalabras(oracion) {
6     return oracion.toUpperCase().split(' ').sort();
7   }
8
9   btn.addEventListener('click', () => {
10    const texto = prompt("Ingrese una oración (ejemplo: sol luna estrella planeta:");
11    if (!texto) {
12      out.textContent = "Debe ingresar una oración.";
13      return;
14    }
15
16    const resultado = reordenarPalabras(texto);
17    out.textContent = `Palabras ordenadas: [${resultado.join(', ')}]`;
18  });
19 });
```



## 5. Reordenar palabras alfabéticamente

Ejecutar ejercicio 5

Palabras ordenadas: [FRESA, MANZANA, PLATANO, SANDIA]

### 6. Crea un Set con nombres y elimina los repetidos automáticamente

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 13min



```
<> index.html JS ejercicio06.js JS ejercicio04.js JS ejercicio03.js
js > JS ejercicio06.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2     const btn = document.getElementById('run-ej6');
3     const out = document.getElementById('out-ej6');
4
5     btn.addEventListener('click', () => {
6         const nombres = ["Ana", "Luis", "Pedro", "Ana", "María", "Pedro"];
7         const setNombres = new Set(nombres);
8         out.textContent = `Original: [${nombres.join(', ')}]\nSin duplicados: [${[...setNombres].join(', ')}]`;
9     });
10 });
```

## 6. Set con nombres (sin duplicados)

Ejecutar ejercicio 6

Original: [Ana, Luis, Pedro, Ana, María, Pedro]  
Sin duplicados: [Ana, Luis, Pedro, María]

### 7. Crea una función tieneDuplicados(arr) que use un Set para determinar si un arreglo contiene elementos repetidos. Debe devolver true si hay duplicados

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 16min

```
<> index.html JS ejercicio07.js JS ejercicio05.js JS ejercicio04.js JS ejercicio03.js
js > JS ejercicio07.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2     const btn = document.getElementById('run-ej7');
3     const out = document.getElementById('out-ej7');
4
5     function tieneDuplicados(arr) {
6         return new Set(arr).size !== arr.length;
7     }
8
9     btn.addEventListener('click', () => {
10         const ejemplo = [1, 2, 3, 2, 4];
11         out.textContent = `Arreglo: [${ejemplo}] \n¿Tiene duplicados? ${tieneDuplicados(ejemplo)}`;
12     });
13 });
```

## 7. Verificar si hay duplicados

Ejecutar ejercicio 7

Arreglo: [1,2,3,2,4]  
¿Tiene duplicados? true

### 8. Crea un Map con productos y precios, y calcula el precio total de una lista de compras

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 15min

js &gt; JS ejercicio08.js &gt; ...

```
1 document.addEventListener('DOMContentLoaded', () => {
2     const btn = document.getElementById('run-ej8');
3     const out = document.getElementById('out-ej8');
4
5     btn.addEventListener('click', () => {
6         //Map con productos y precios
7         const productos = new Map([
8             ["pan", 2.5],
9             ["leche", 4.2],
10            ["queso", 10.5],
11            ["jugo", 6.0]
12        ]);
13
14        //Simulación una lista de compras
15        const lista = ["pan", "queso", "leche"];
16
17        let total = 0;
18        lista.forEach(item => {
19            if (productos.has(item)) {
20                total += productos.get(item);
21            }
22        });
23
24        out.textContent = `Productos disponibles: ${Array.from(productos.entries())}
25        Lista de compras: [${lista.join(', ')}]
26        Precio total: S/. ${total.toFixed(2)}`;
27    });
28 });
```

## 8. Calcular precio total con Map

[Ejecutar ejercicio 8](#)

Productos disponibles: pan,2.5,leche,4.2,queso,10.5,jugo,6  
Lista de compras: [pan, queso, leche]  
Precio total: S/. 17.20

9. Crea una función `contarPalabras(texto)` que use un Map para contar cuántas veces aparece cada palabra en una cadena (sin distinguir mayúsculas).

```
contarPalabras("sol luna sol sol estrella luna");
```

```
// Map { "sol" => 3, "luna" => 2, "estrella" => 1 }
```

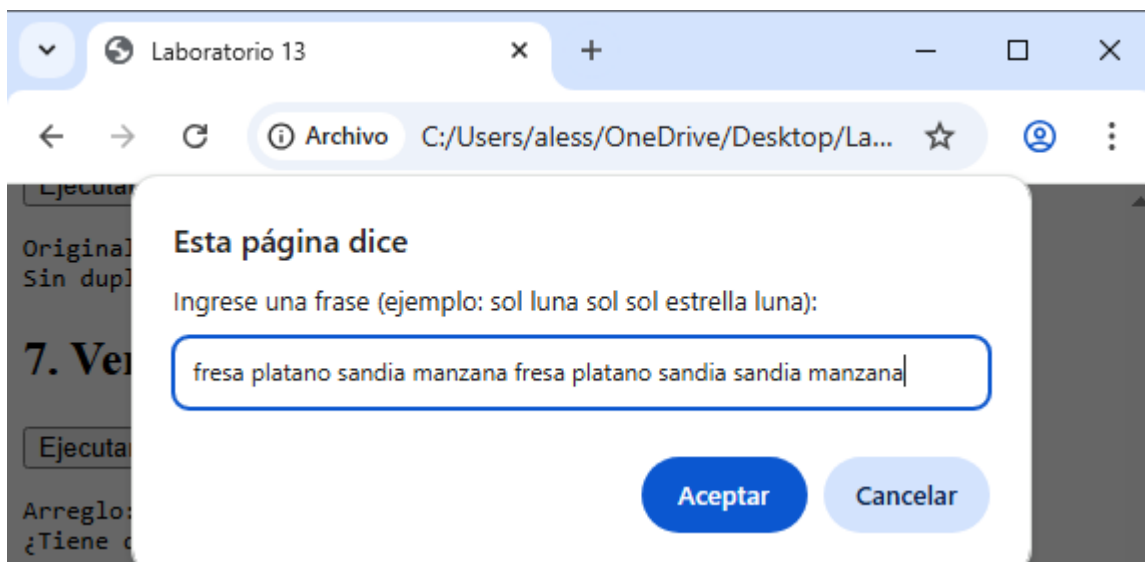
```
// Laboratorio Nro 13 - Ejercicio3
```

```
// Autor: Alessandro Josue Justo Vilca
```

```
// Colaboró : sin colaboradores
```

```
// Tiempo : 14min
```

```
index.html JS ejercicio JS ejercicio07.js JS ejercicio06.js JS ejercicio05.js
js > JS ejercicio09.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2   const btn = document.getElementById('run-ej9');
3   const out = document.getElementById('out-ej9');
4
5   function contarPalabras(texto) {
6     const palabras = texto.toLowerCase().split(' ');
7     const conteo = new Map();
8
9     for (let palabra of palabras) {
10      conteo.set(palabra, (conteo.get(palabra) || 0) + 1);
11    }
12
13    return conteo;
14  }
15
16  btn.addEventListener('click', () => {
17    const texto = prompt("Ingrese una frase (ejemplo: sol luna sol sol estrella luna:");
18    if (!texto) {
19      out.textContent = "Debe ingresar texto.";
20      return;
21    }
22
23    const resultado = contarPalabras(texto);
24    let salida = "";
25    resultado.forEach((valor, clave) => {
26      salida += `${clave}: ${valor}\n`;
27    });
28
29    out.textContent = salida.trim();
30  });
31 });
```



## 9. Contar palabras con Map

Ejecutar ejercicio 9

```
fresa: 2
platano: 2
sandia: 3
manzana: 2
```

10. Crea una función `invertirMap(map)` que reciba un `Map` y devuelva uno nuevo donde las claves sean los valores y los valores sean las claves.

```
let capitales = new Map([
  ["Perú", "Lima"],
  ["Chile", "Santiago"]
]);
invertirMap(capitales); // Map { "Lima" => "Perú", "Santiago" => "Chile" }
// Laboratorio Nro 13 - Ejercicio3
// Autor: Alessandro Josue Justo Vilca
// Colaboró : sin colaboradores
// Tiempo : 16min
```



```
js > JS ejercicio10.js > ...
1  document.addEventListener('DOMContentLoaded', () => {
2      const btn = document.getElementById('run-ej10');
3      const out = document.getElementById('out-ej10');
4
5      function invertirMap(map) {
6          const nuevo = new Map();
7          for (let [clave, valor] of map) {
8              nuevo.set(valor, clave);
9          }
10         return nuevo;
11     }
12
13     btn.addEventListener('click', () => {
14         const capitales = new Map([
15             ["Perú", "Lima"],
16             ["Chile", "Santiago"]
17         ]);
18
19         const invertido = invertirMap(capitales);
20         let texto = "";
21         invertido.forEach((v, k) => {
22             texto += `${k} => ${v}\n`;
23         });
24
25         out.textContent = texto.trim();
26     });
27 });
```

## 10. Invertir claves y valores de un Map

Ejecutar ejercicio 10

Lima => Perú  
Santiago => Chile

11. Crea un objeto `auto` con propiedades `marca`, `modelo`, `año`, y un método `detalles()` que muestre sus datos

```
// Laboratorio Nro 13 - Ejercicio3
// Autor: Alessandro Josue Justo Vilca
// Colaboró : sin colaboradores
```

// Tiempo : 15min

```
<> index.html JS ejercic JS ejercicio09.js JS ejercicio08.js JS ejerci
js > JS ejercicio11.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2     const btn = document.getElementById('run-ej11');
3     const out = document.getElementById('out-ej11');
4
5     const auto = {
6         marca: "Toyota",
7         modelo: "Corolla",
8         año: 2022,
9         detalles: function() {
10             return `Marca: ${this.marca}\nModelo: ${this.modelo}\nAño: ${this.año}`;
11         }
12     };
13
14     btn.addEventListener('click', () => {
15         out.textContent = auto.detalles();
16     });
17 });
```

## 11. Objeto auto con método detalles()

Ejecutar ejercicio 11

Marca: Toyota  
Modelo: Corolla  
Año: 2022

12. Crea una función contarLetras(texto) que cuente cuántas veces aparece cada letra en una cadena (ignora mayúsculas/minúsculas).

Devuelve un objeto con el formato { letra: cantidad }.

contarLetras("banana"); // { b:1, a:3, n:2 }

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 18min

js &gt; JS ejercicio12.js &gt; ...

```

1  document.addEventListener('DOMContentLoaded', () => {
2      const btn = document.getElementById('run-ej12');
3      const out = document.getElementById('out-ej12');
4
5      function contarLetras(texto) {
6          const letras = texto.toLowerCase().split('');
7          const conteo = {};
8
9          for (let letra of letras) {
10             if (letra !== ' ') {
11                 conteo[letra] = (conteo[letra] || 0) + 1;
12             }
13         }
14         return conteo;
15     }
16
17     btn.addEventListener('click', () => {
18         const texto = prompt("Ingrese una palabra o frase (ejemplo: banana:");
19         if (!texto) {
20             out.textContent = "Debe ingresar un texto válido.";
21             return;
22         }
23
24         const resultado = contarLetras(texto);
25         out.textContent = JSON.stringify(resultado, null, 2);
26     });
27 });

```

## 12. Contar letras en texto

Ejecutar ejercicio 12

```

{
  "d": 1,
  "e": 2,
  "x": 2,
  "o": 3,
  "i": 3,
  "r": 2,
  "b": 1,
  "n": 1,
  "u": 1,
  "c": 2,
  "l": 1
}

```

13. Crea una función llamada `combinarCatalogos(tiendaA, tiendaB)` que reciba dos objetos donde: Cada propiedad representa el nombre de un producto y su valor es el precio

La función debe devolver un nuevo objeto que:

- Combine ambos catálogos de las tiendas
- Si un producto aparece en ambos, se debe mantener el precio más bajo
- Y todos los precios finales deben convertirse a dos decimales
- TiendaA: laptop: 3500.5, mouse: 100.35, teclado: 250.9
- TiendaB: mouse: 95.2, monitor: 720.457, teclado: 260.1

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 18min

```
index.html JS ejercicio11.js JS ejercicio10.js JS ejercicio13.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2   const btn = document.getElementById('run-ej13');
3   const out = document.getElementById('out-ej13');
4
5   function combinarCatalogos(tiendaA, tiendaB) {
6     const combinado = { ...tiendaA };
7
8     for (let producto in tiendaB) {
9       if (combinado[producto]) {
10        combinado[producto] = Math.min(combinado[producto], tiendaB[producto]);
11      } else {
12        combinado[producto] = tiendaB[producto];
13      }
14    }
15
16    //Redondear a 2 decimales
17    for (let prod in combinado) {
18      combinado[prod] = combinado[prod].toFixed(2);
19    }
20
21    return combinado;
22  }
23
24  btn.addEventListener('click', () => {
25    const tiendaA = { laptop: 3500.5, mouse: 100.35, teclado: 250.9 };
26    const tiendaB = { mouse: 95.2, monitor: 720.457, teclado: 260.1 };
27
28    const resultado = combinarCatalogos(tiendaA, tiendaB);
29    out.textContent = JSON.stringify(resultado, null, 2);
30  });
31 });
```

## 13. Combinar catálogos de tiendas

Ejecutar ejercicio 13

```
{
  "laptop": "3500.50",
  "mouse": "95.20",
  "teclado": "250.90",
  "monitor": "720.46"
}
```

14. Crea una función llamada `gestionarEmpleados(empleados)` que reciba un array de objetos, donde cada objeto representa un empleado con las siguientes propiedades:

- `id`: número,
- `nombre`: string,
- `area`: string,
- `salario`: número

La función debe:

Crear un Map donde la clave sea el área (por ejemplo "Ventas", "TI", "Recursos Humanos"),

y el valor sea un array con los nombres de los empleados que trabajan en esa área.

Ejemplo de entrada:

```
const listaEmpleados = [
  { id: 1, nombre: "Juan", area: "Ventas", salario: 2400 },
  { id: 2, nombre: "Marta", area: "Ventas", salario: 2600 },
  { id: 3, nombre: "Luis", area: "TI", salario: 4000 },
  { id: 4, nombre: "Ana", area: "Recursos Humanos", salario: 3000 }
];
console.log(gestionarEmpleados(listaEmpleados));
```

Salida esperada:

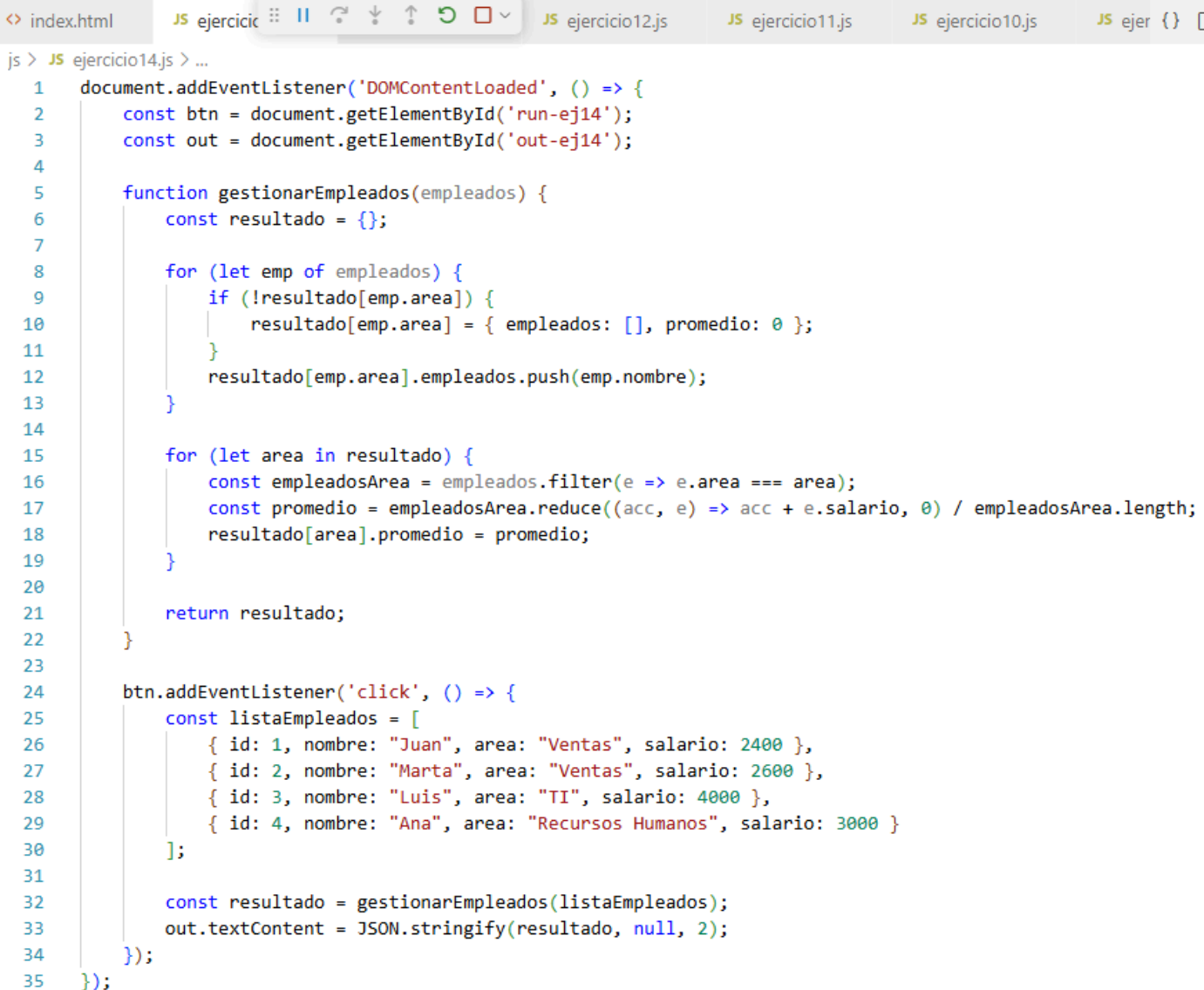
```
{
  Ventas: { empleados: [ 'Juan', 'Marta' ], promedio: 2500 },
  TI: { empleados: [ 'Luis' ], promedio: 4000 },
  Recursos Humanos: { empleados: [ 'Ana' ], promedio: 3000 }
}
```

// Laboratorio Nro 13 - Ejercicio3

// Autor: Alessandro Josue Justo Vilca

// Colaboró : sin colaboradores

// Tiempo : 18min



```
<> index.html JS ejercic JS ejercicio12.js JS ejercicio11.js JS ejercicio10.js JS ejer {} [
js > JS ejercicio14.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2   const btn = document.getElementById('run-ej14');
3   const out = document.getElementById('out-ej14');
4
5   function gestionarEmpleados(empleados) {
6     const resultado = {};
7
8     for (let emp of empleados) {
9       if (!resultado[emp.area]) {
10         resultado[emp.area] = { empleados: [], promedio: 0 };
11       }
12       resultado[emp.area].empleados.push(emp.nombre);
13     }
14
15     for (let area in resultado) {
16       const empleadosArea = empleados.filter(e => e.area === area);
17       const promedio = empleadosArea.reduce((acc, e) => acc + e.salario, 0) / empleadosArea.length;
18       resultado[area].promedio = promedio;
19     }
20
21     return resultado;
22   }
23
24   btn.addEventListener('click', () => {
25     const listaEmpleados = [
26       { id: 1, nombre: "Juan", area: "Ventas", salario: 2400 },
27       { id: 2, nombre: "Marta", area: "Ventas", salario: 2600 },
28       { id: 3, nombre: "Luis", area: "TI", salario: 4000 },
29       { id: 4, nombre: "Ana", area: "Recursos Humanos", salario: 3000 }
30     ];
31
32     const resultado = gestionarEmpleados(listaEmpleados);
33     out.textContent = JSON.stringify(resultado, null, 2);
34   });
35 });
```



## 14. Gestionar empleados por área

Ejecutar ejercicio 14

```
{
  "Ventas": {
    "empleados": [
      "Juan",
      "Marta"
    ],
    "promedio": 2500
  },
  "TI": {
    "empleados": [
      "Luis"
    ],
    "promedio": 4000
  },
  "Recursos Humanos": {
    "empleados": [
      "Ana"
    ],
    "promedio": 3000
  }
}
```

15. Crear un repositorio remoto en GitHub y subir tu repositorio local. Compartir URL y pdf con captura de pantalla del código de los archivos js y de la ejecución

The screenshot shows a GitHub repository named "laboratorio13-IDWeb" by user "Ajustov". The repository is public and has 1 branch (master) and 0 tags. The file list shows three files: "js", "LABORATORIO 13 - JAVASCRIPT\_ESTRUCTUR...", and "index.html", all from the "Primer commit - laboratorio13 - ejercicio15".

Below the repository page, a terminal window is open, showing the following commands and output:

```
MINGW64/c:/Users/aless/OneDrive/Desktop/Laboratorio13/alessandro_justo/laboratorio13
aless@DESKTOP-T6G2CKT MINGW64 ~/OneDrive/Desktop/Laboratorio13/alessandro_justo/laboratorio13
$ git init
Initialized empty Git repository in C:/Users/aless/OneDrive/Desktop/Laboratorio13/alessandro_justo/Laboratorio13/.git/
aless@DESKTOP-T6G2CKT MINGW64 ~/OneDrive/Desktop/Laboratorio13/alessandro_justo/laboratorio13 (master)
$ git remote add origin https://github.com/Ajustov/laboratorio13-IDWeb.git
aless@DESKTOP-T6G2CKT MINGW64 ~/OneDrive/Desktop/Laboratorio13/alessandro_justo/laboratorio13 (master)
$ git add .
aless@DESKTOP-T6G2CKT MINGW64 ~/OneDrive/Desktop/Laboratorio13/alessandro_justo/laboratorio13 (master)
$ git commit -m "Primer commit - laboratorio13 - ejercicio15"
[master (root-commit) 57f10fb] Primer commit - laboratorio13 - ejercicio15
14 files changed, 378 insertions(+)
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 8 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (17/17), 763.46 KiB | 18.18 MiB/s, done.
Total 17 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Ajustov/laboratorio13-IDWeb.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
aless@DESKTOP-T6G2CKT MINGW64 ~/OneDrive/Desktop/Laboratorio13/alessandro_justo/laboratorio13 (master)
$ |
```

URL: <https://github.com/Ajustov/laboratorio13-IDWeb>