3 way partition problem

Grupo 6
Sección 1

Juan Pablo Contreras
Rodrigo Heiremans
Matias Perez

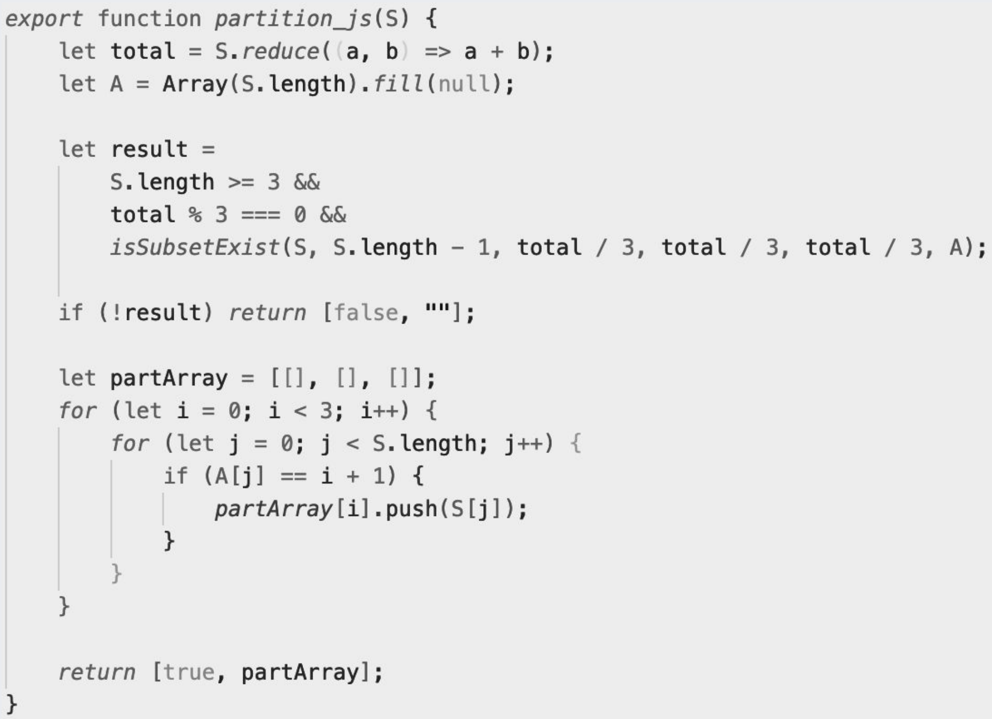
Proceso

- Estudio del algoritmo
- Código en C y en JS
- Uso de emscripten
- Creación de HTML

Demo

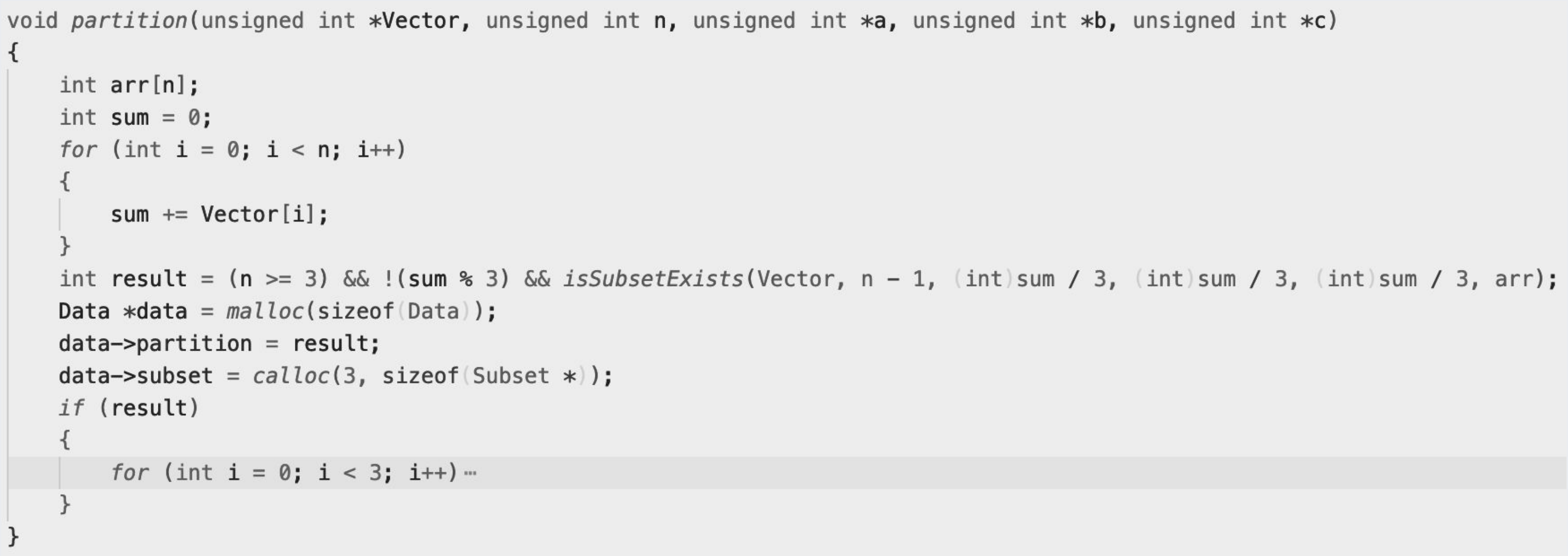
Código

Partition JavaScript



```
export function partition_js(S) {
    let total = S.reduce((a, b) => a + b);
    let A = Array(S.length).fill(null);

    let result =
        S.length >= 3 &&
        total % 3 === 0 &&
        isSubsetExist(S, S.length - 1, total / 3, total / 3, total / 3, A);

    if (!result) return [false, ""];

    let partArray = [[], [], []];
    for (let i = 0; i < 3; i++) {
        for (let j = 0; j < S.length; j++) {
            if (A[j] == i + 1) {
                partArray[i].push(S[j]);
            }
        }
    }

    return [true, partArray];
}
```

Partition C

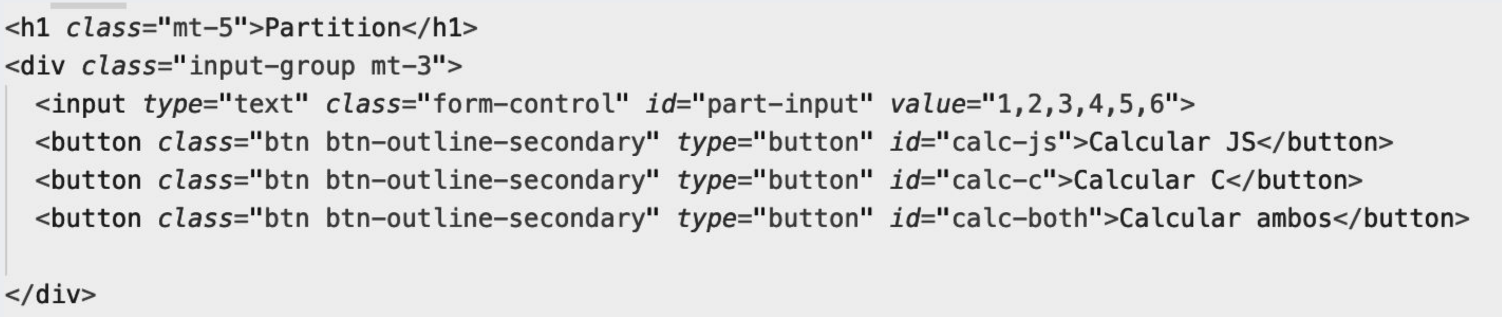


```c
void partition(unsigned int *Vector, unsigned int n, unsigned int *a, unsigned int *b, unsigned int *c)
{
    int arr[n];
    int sum = 0;
    for (int i = 0; i < n; i++)
    {
        sum += Vector[i];
    }
    int result = (n >= 3) && !(sum % 3) && isSubsetExists(Vector, n - 1, (int)sum / 3, (int)sum / 3, (int)sum / 3, arr);
    Data *data = malloc(sizeof(Data));
    data->partition = result;
    data->subset = calloc(3, sizeof(Subset *));
    if (result)
    {
        for (int i = 0; i < 3; i++)…
    }
}
```

Compilador

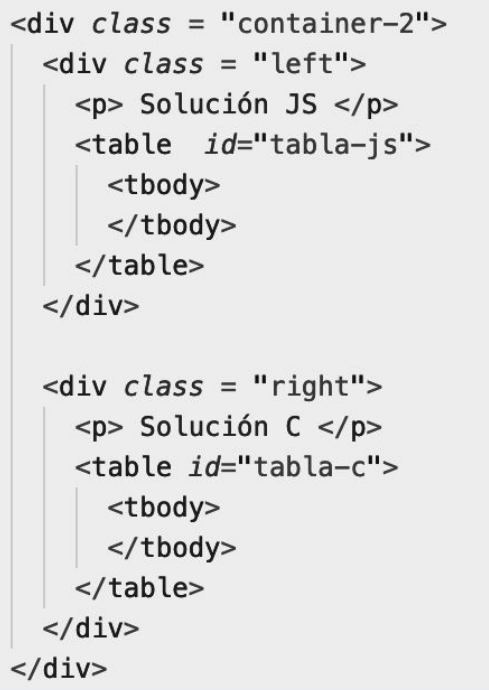
```
emcc -s ASSERTIONS=1 -s WASM=1 -s EXPORT_ES6=1 -s EXPORTED_RUNTIME_METHODS='["cwrap", "getValue", "setValue"]'

-s EXPORTED_FUNCTIONS="['_partition', '_malloc']" -o Cpartition.js src/partition.c
```

HTML



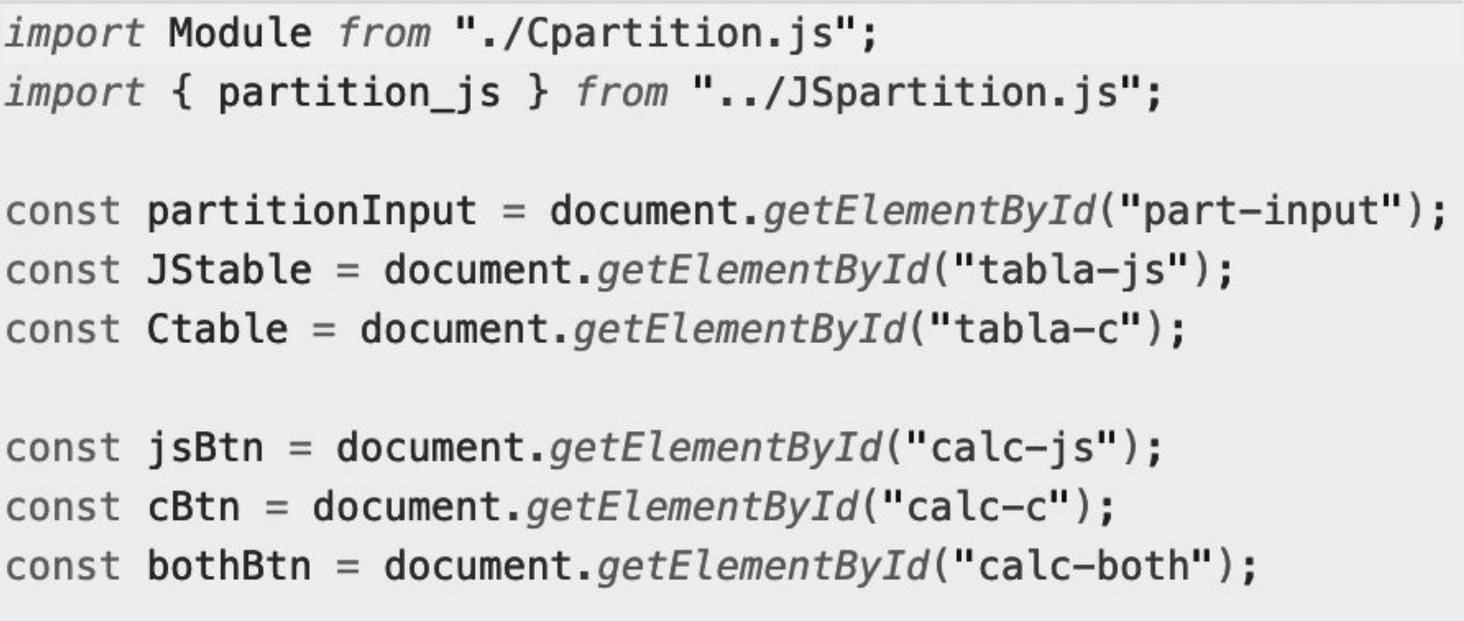
```html
<h1 class="mt-5">Partition</h1>
<div class="input-group mt-3">
  <input type="text" class="form-control" id="part-input" value="1,2,3,4,5,6">
  <button class="btn btn-outline-secondary" type="button" id="calc-js">Calcular JS</button>
  <button class="btn btn-outline-secondary" type="button" id="calc-c">Calcular C</button>
  <button class="btn btn-outline-secondary" type="button" id="calc-both">Calcular ambos</button>

</div>
```

HTML



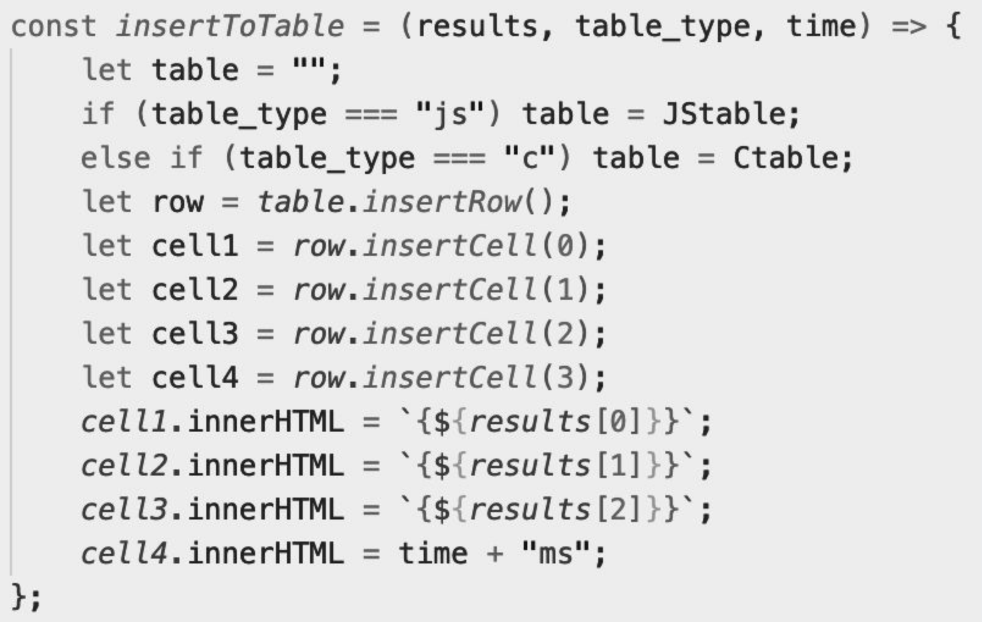
```
<div class = "container-2">
  <div class = "left">
    <p> Solución JS </p>
    <table  id="tabla-js">
      <tbody>
      </tbody>
    </table>
  </div>

  <div class = "right">
    <p> Solución C </p>
    <table id="tabla-c">
      <tbody>
      </tbody>
    </table>
  </div>
</div>
```

JavaScript



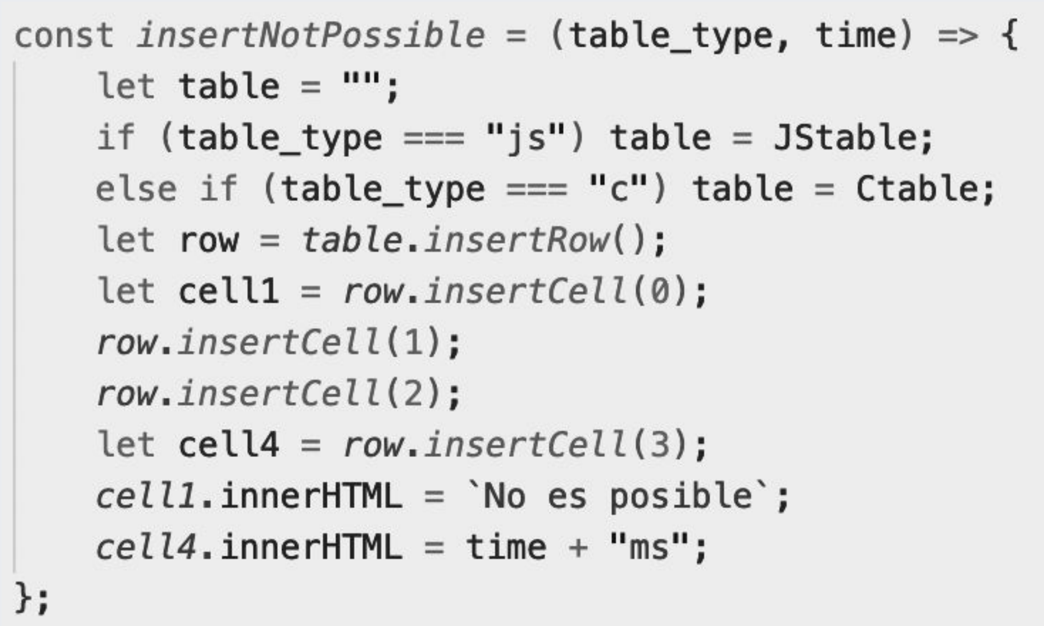
```javascript
import Module from "./Cpartition.js";
import { partition_js } from "../JSpartition.js";

const partitionInput = document.getElementById("part-input");
const JStable = document.getElementById("tabla-js");
const Ctable = document.getElementById("tabla-c");

const jsBtn = document.getElementById("calc-js");
const cBtn = document.getElementById("calc-c");
const bothBtn = document.getElementById("calc-both");
```

JavaScript



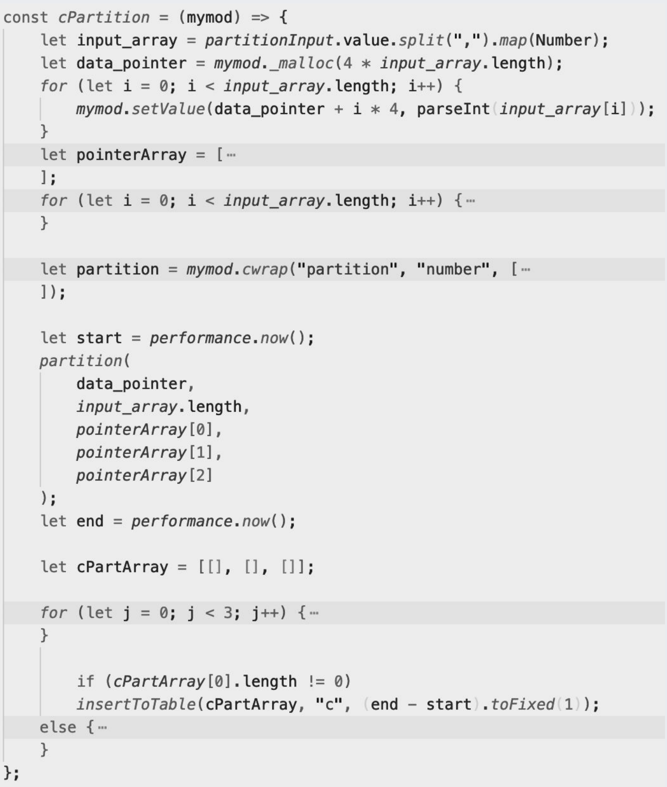
```
const insertToTable = (results, table_type, time) => {
    let table = "";
    if (table_type === "js") table = JStable;
    else if (table_type === "c") table = Ctable;
    let row = table.insertRow();
    let cell1 = row.insertCell(0);
    let cell2 = row.insertCell(1);
    let cell3 = row.insertCell(2);
    let cell4 = row.insertCell(3);
    cell1.innerHTML = `{${results[0]}}`;
    cell2.innerHTML = `{${results[1]}}`;
    cell3.innerHTML = `{${results[2]}}`;
    cell4.innerHTML = time + "ms";
};
```

JavaScript



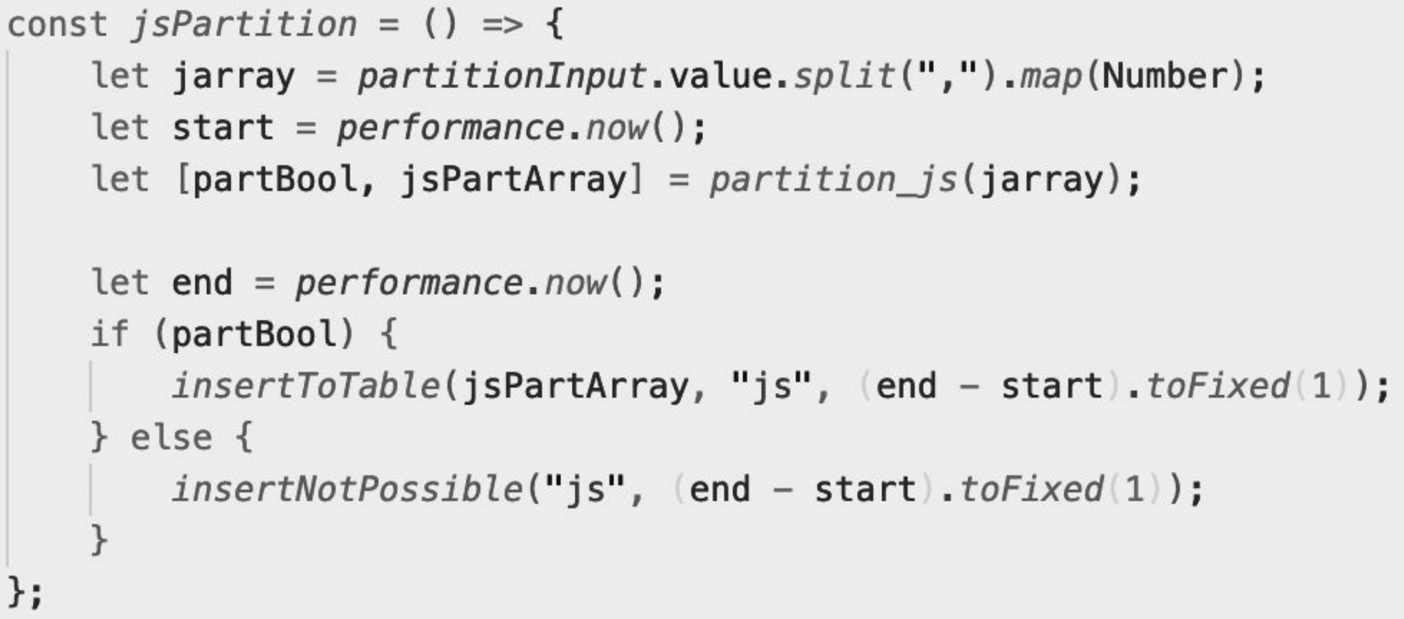
```
const insertNotPossible = (table_type, time) => {
    let table = "";
    if (table_type === "js") table = JStable;
    else if (table_type === "c") table = Ctable;
    let row = table.insertRow();
    let cell1 = row.insertCell(0);
    row.insertCell(1);
    row.insertCell(2);
    let cell4 = row.insertCell(3);
    cell1.innerHTML = `No es posible`;
    cell4.innerHTML = time + "ms";
};
```

JavaScript



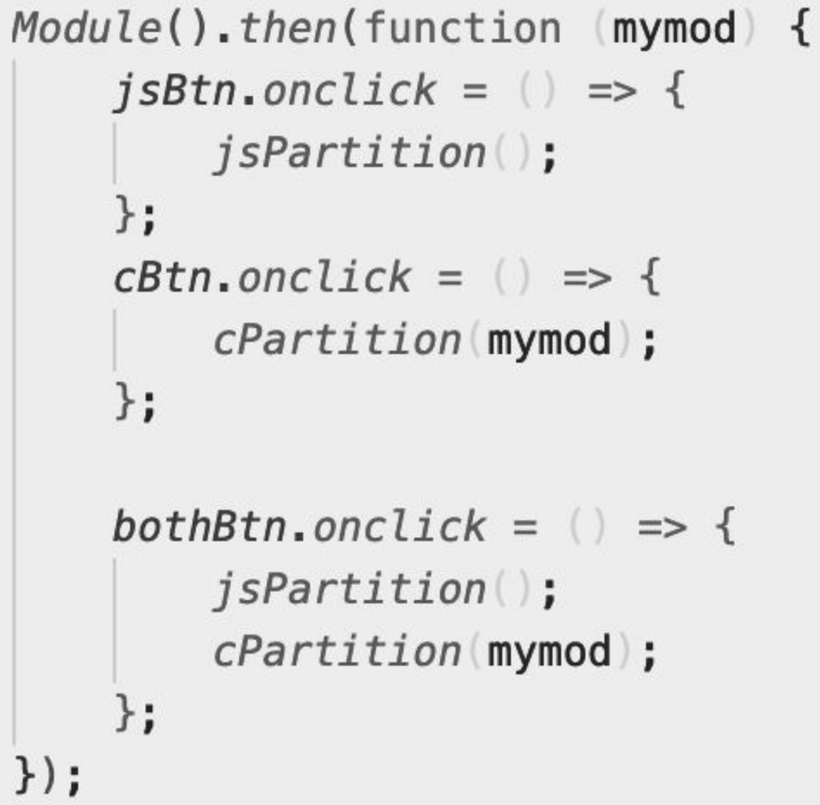
```
const cPartition = (mymod) => {
    let input_array = partitionInput.value.split(",").map(Number);
    let data_pointer = mymod._malloc(4 * input_array.length);
    for (let i = 0; i < input_array.length; i++) {
        mymod.setValue(data_pointer + i * 4, parseInt(input_array[i]));
    }
    let pointerArray = [⋯
    ];
    for (let i = 0; i < input_array.length; i++) {⋯
    }

    let partition = mymod.cwrap("partition", "number", [⋯
    ]);

    let start = performance.now();
    partition(
        data_pointer,
        input_array.length,
        pointerArray[0],
        pointerArray[1],
        pointerArray[2]
    );
    let end = performance.now();

    let cPartArray = [[], [], []];

    for (let j = 0; j < 3; j++) {⋯
    }

        if (cPartArray[0].length != 0)
        insertToTable(cPartArray, "c", (end - start).toFixed(1));
    else {⋯
    }
};
```

JavaScript



```
const jsPartition = () => {
    let jarray = partitionInput.value.split(",").map(Number);
    let start = performance.now();
    let [partBool, jsPartArray] = partition_js(jarray);

    let end = performance.now();
    if (partBool) {
        insertToTable(jsPartArray, "js", (end - start).toFixed(1));
    } else {
        insertNotPossible("js", (end - start).toFixed(1));
    }
};
```

JavaScript



```
Module().then(function (mymod) {
    jsBtn.onclick = () => {
        jsPartition();
    };
    cBtn.onclick = () => {
        cPartition(mymod);
    };

    bothBtn.onclick = () => {
        jsPartition();
        cPartition(mymod);
    };
});
```

Conclusiones

Web
Assembly

Grupo 6
Sección 1

Juan Pablo Contreras
Rodrigo Heiremans
Matias Perez

Codigo basado de

https://www.techiedelight.com/3-partition-problem-extended-print-all-partitions/