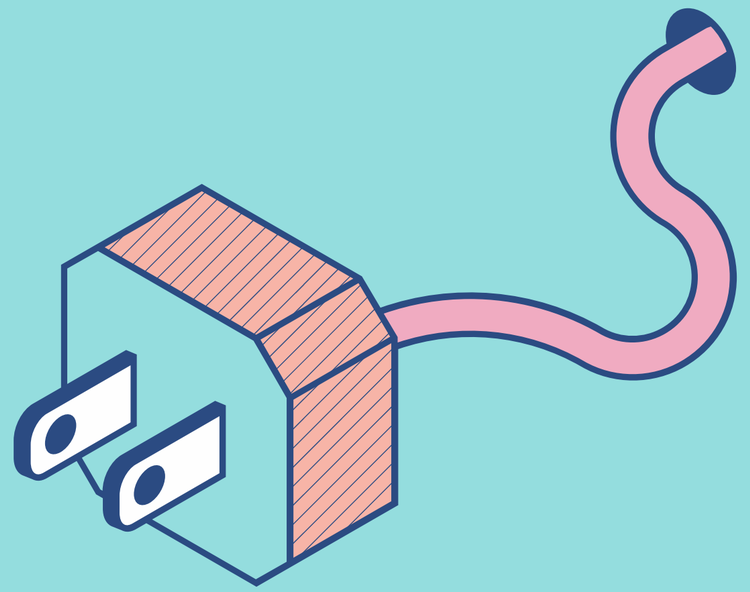
IIIC3548 - DAAW

Tarea 3 - WebAssembly

Grupo 7: Matías Cadile, Matías Soto y José Luco

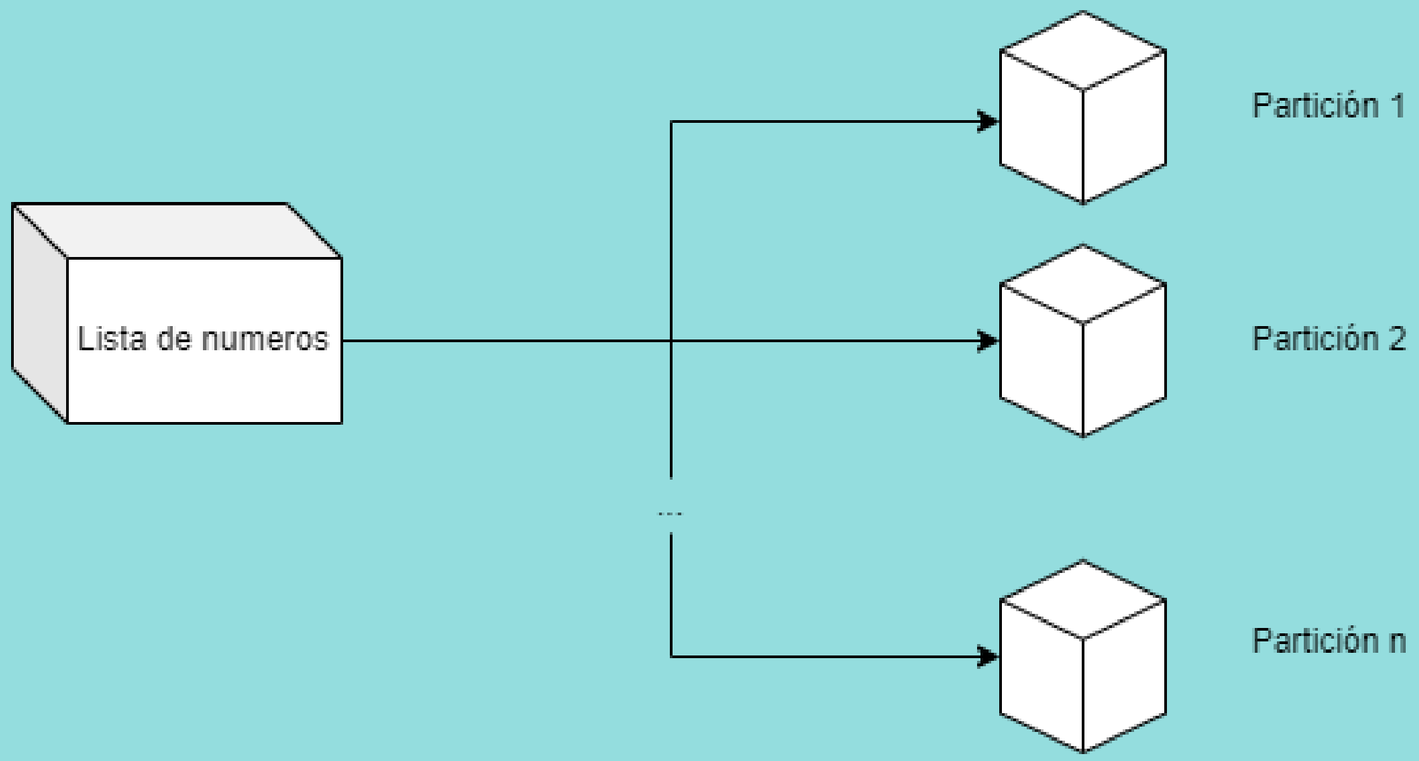
WebAssembly

Incorporar código C/C++ en un aplicación Web



Problema sugerido

Encontar particiones en listas de modo que ellas sumen lo mismo



2 Soluciones

1 — C++ / WASM

2 — JAVASCRIPT

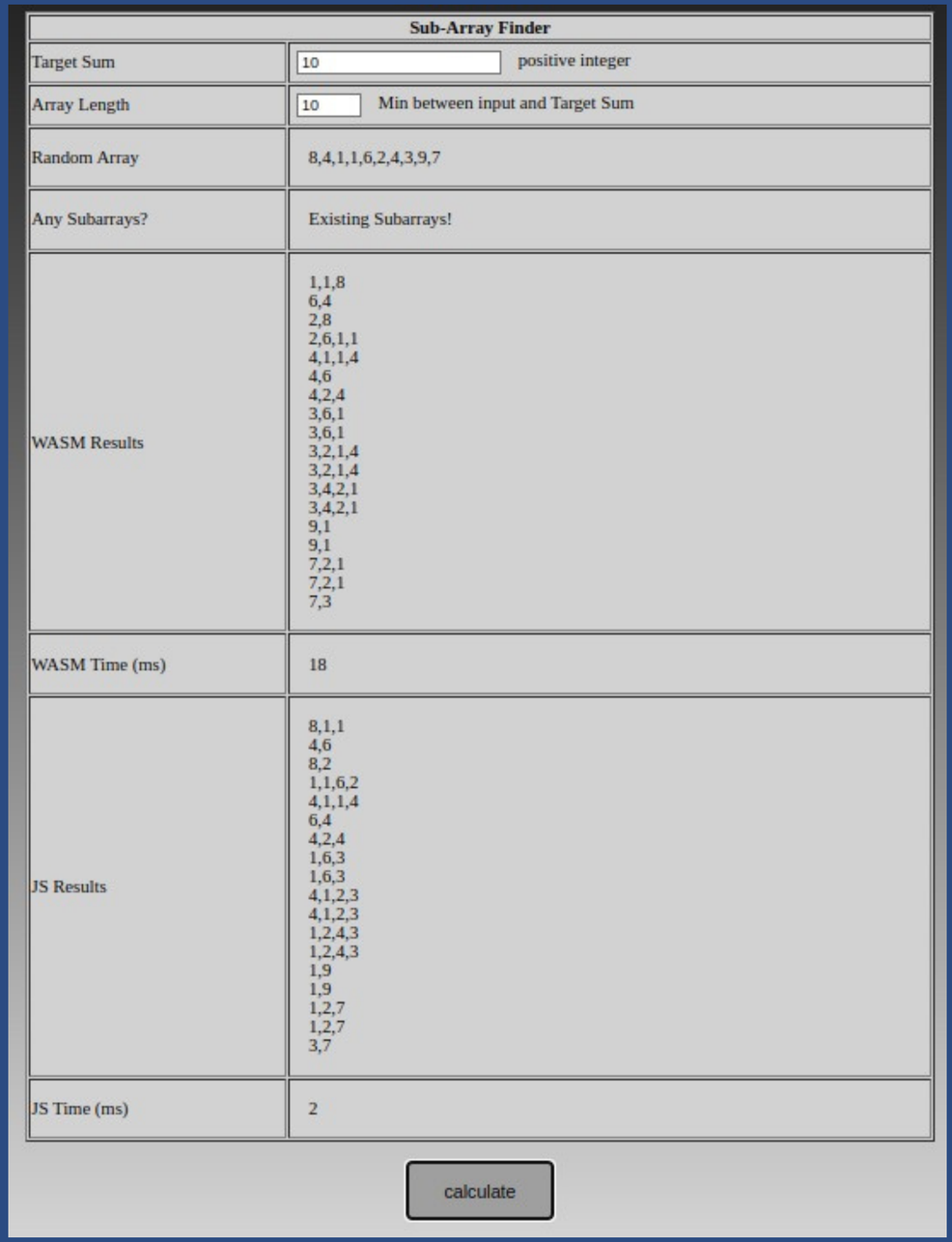
Solucion 1
C++ / WASM

- Solucion recursiva.
- Funciones getValue y SetValue para manejar punteros desde JS.
- Emscripten para ocupar y liberar Calloc

```cpp
#include <vector>
#include <iostream>
#include <stdbool.h>
#include <emscripten/bind.h>
#include <emscripten/emscripten.h>
using namespace std;

// dp[i][j] is going to store true if sum j is
// possible with array elements from 0 to i.
bool ** dp;

int32_t** results;
int lastRow;
int lastCol;

void fillArray(int32_t* a, int32_t len);

extern "C"{
    void display(const vector<int>& v)
    {
        for (int i = 0; i < v.size(); ++i){
            printf("%d ", v[i]);
            results[lastRow][lastCol] = v[i];
            lastCol ++;
        }
        printf("\n");
        lastRow ++;
        lastCol = 0;

    }
}

// A recursive function to print all subsets with the
// help of dp[][]. Vector p[] stores current subset.
extern "C"{
    void printSubsetsRec(int32_t* arr, int i, int sum, vector<int>& p)
    {
```

Dificultades

- Liberación/Asignación de memoria.
- Manejo de Vectores 2D para guardar las soluciones --> Emscripten Bindings.
- Cantidad de memora asignada para soluciones muy grandes.



Sub-Array Finder	
Target Sum	10 positive integer
Array Length	10 Min between input and Target Sum
Random Array	8,4,1,1,6,2,4,3,9,7
Any Subarrays?	Existing Subarrays!
WASM Results	1,1,8 6,4 2,8 2,6,1,1 4,1,1,4 4,6 4,2,4 3,6,1 3,6,1 3,2,1,4 3,2,1,4 3,4,2,1 3,4,2,1 9,1 9,1 7,2,1 7,2,1 7,3
WASM Time (ms)	18
JS Results	8,1,1 4,6 8,2 1,1,6,2 4,1,1,4 6,4 4,2,4 1,6,3 1,6,3 4,1,2,3 4,1,2,3 1,2,4,3 1,2,4,3 1,9 1,9 1,2,7 1,2,7 3,7
JS Time (ms)	2

calculate

Solucion 2 JavaScript

- Funciones asincronas para manejo de tiempo.
- Combination.filter del arreglo de numeros.

```
function combinations(array) {
  return new Array(1 << array.length)
    .fill()
    .map((e1, i) => array.filter((e2, j) => i & (1 << j)));
}

function add(a, b) {
  return a + b;
}

function pariticionsAddN(array, n) {
  return combinations(array).filter((subarray) => subarray.reduce(add, 0) == n);
}

async function timeParticion(array, n) {
  var startTime = performance.now();
  //var output = await resolveAfter3Seconds();
  var output = await pariticionsAddN(array, n);
  var endTime = performance.now();
  var time = endTime - startTime;
  //output.push(time);
  return [output, time];
}

async function resolveAfter3Seconds() {
  return new Promise((resolve) => {
    setTimeout(() => {
      resolve("resolved");
    }, 3000);
  });
}

async function Example() {
  const a = await timeParticion([2, 4, 45, 6, 0, 19], 51);
  console.log(a);
}

export default timeParticion;
```

Aplicación Web Solución

DEMO

Conclusiones

- **Bindings** son útiles para **reutilizar código OOP** en lenguajes como Rust, C y C++
- **Embind.h** alternativa directa para hacer **bindings**.
  - **WebIDL Binder** como vía más sencilla
- WASM para computación intensiva
  - Graph/arrays
  - dApps & Blockchain