Reporte de Diagnóstico Inicial / Initial Diagnosis Report.

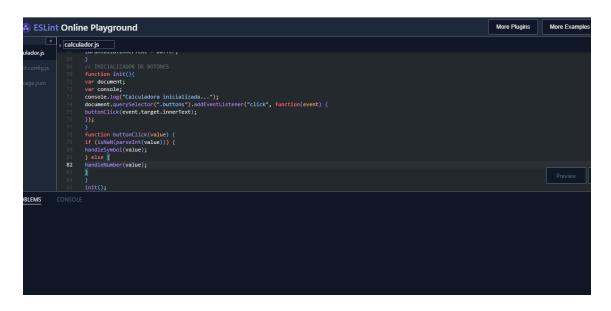
ANTES

```
dpoatestreen();
} function andlesymbol(symbol) {
    switch (symbol) {
    case 'C':
    buffer = "0";
    wemoria = 0;
    ultimo_operador = null;
    break;
    case '-':
    if (ultimo_operador === null) {
        return;
        return;
    }
}
 }
flushOperation(parseInt(buffer));
ultimo_operador = null;
buffer = "" + memoria;
memoria = 0;
break;
case '+':
case '*':
case '*':
  function handleMath(symbol) {
  if (buffer === '0' && memoria === 0) {
 }
var intBuffer = parseInt(buffer);
if (memoria === 0) {
   memoria = intBuffer;
} else {
   flushOperation(intBuffer);
}
burler = 0,
}
function flushOperation(intSuffer) {
if (ultimo_operador === '+') {
memoria += intBuffer;
} else if (ultimo_operador === '-') {
memoria -= intBuffer;
} else if (ultimo_operador === '*') {
memoria '= intBuffer;
} else if (ultimo_operador === '/') {
memoria '= intBuffer;
}
  function updateScreen(){
var laPantalla = document.getElementById("display");
laPantalla.innerText = buffer;
```

DESPUES

```
flushoperation(intmuffer);
)
ultimo_operador = symbol;
buffer = "0";

Junction flushoperation(intBuffer) {
if (ultimo_operador == """) {
    menoria -= IntBuffer;
    less if (ultimo_operador == """) {
    memoria -= IntBuffer;
    less if (ultimo_operador == """) {
    memoria -= IntBuffer;
    less if (ultimo_operador == """) {
    memoria -= IntBuffer;
    less if (ultimo_operador == """) {
    memoria /= IntBuffer;
    }
```



ESLint

ESLint es una herramienta de análisis estático de código (conocida como linter) para JavaScript y TypeScript. Su principal objetivo es identificar y reportar problemas en tu código antes de que se ejecute.

Esto incluye:

Errores potenciales: Detecta patrones de código que probablemente sean errores (por ejemplo, usar una variable antes de definirla).

Buenas prácticas: Ayuda a seguir un conjunto de reglas y buenas prácticas de programación.

Problemas de estilo: Puede señalar inconsistencias en el formato (aunque para corregir el formato automáticamente, Prettier es más específico).

Prettier

Prettier es un formateador de código (o code formatter). Su único propósito es reescribir tu código para que cumpla con un conjunto de reglas de estilo consistentes y predefinidas.

A diferencia de ESLint (que busca errores), Prettier solo se preocupa por la apariencia del código:

- Ajusta la indentación (espacios o tabulaciones).
- Maneja los saltos de línea para respetar un ancho máximo.
- Estandariza el uso de comillas (simples o dobles).
- Añade o quita espacios, puntos y comas, etc., según sus reglas.