Tarea clase 3

1. SumEvens

Implementa una función *sumEvens* que tome por parámetro una matriz de números y retorna un template literal con suma de todos sus números pares (ver ejemplo). Si la matriz está vacía, retorna un template literal con el texto "la matriz está vacía".

```
sumEvens([[1,2],[3,4]]); // → El resultado es 6
sumEvens([]); // → La matriz está vacía

let matrix = [[1,2,3],[1,2,3],[1,2,3]]
sumEvens(matrix) // → El resultado es 6
```

> tip: vas a tener que usar un `for` anidado. Usa `%` para chequear si un número es par.

2. InvertMatrix

Implementar una función *invertMatrix* que tome por parámetro una matriz y retorne una versión invertida de la original. Vea el ejemplo:

```
invertMatrix([[1,2],[3,4]]) // → [[3,4],[1,2]]
```

3. SumMatrix

Implementar una función *sumMatrix* que reciba por parámetro dos matrices de números enteros con la misma dimensión. Antes de realizar los cálculos la función debe comprobar que las dimensiones de las matrices recibidas por parámetros tengan la misma dimensión, en caso contrario debe retornar -1. En caso de pasar la validación de las dimensiones, se debe calcular la suma de ambas matrices y retornar la

matriz resultante. Finalmente se solicita invocar a la función y mostrar el resultado por la consola del sistema.

Ejemplos:

```
SumMatrix ([[1,2],[3,4]],[[5,6],[7,8]]) ⇒ [[6,8],[7,12]]
SumMartix ([[10,23,2],[3,56,11]],[[12,3],[4,5]]) ⇒ -1
```