Hell Triangle - Challenge

Given a triangle of numbers, find the maximum total from top to bottom

Example:

6

3 5

9 7 1

4 6 8 4

In this triangle the maximum total is 6 + 5 + 7 + 8 = 26

An element can only be summed with one of the two nearest elements in the next row So the element 3 in row 2 can be summed with 9 and 7, but not with 1

Choose the programming language you want... let us know about why is that your choice Besides the solution itself, write an automated test for your code (using a known framework or just another function/method)

Your code will receive an (multidimensional) array as input.

The triangle from above would be:

example = [[6],[3,5],[9,7,1],[4,6,8,4]]

O programa foi implementado na linguagem TypeScript baseado na tecnologia NodeJS e foi usado a biblioteca mocha para realizar os testes, a escolha desta tecnologia foi devido à ampla variedade de bibliotecas prontas e ao alto nível da linguagem que facilita o desenvolvimento.

1. Correctness:

A solução desenvolvida permite determinar o máximo total de uma matriz triangular independente da altura, demostrando que a implementação foi realizada corretamente, o algoritmo é recursivo o que permitiu uma solução simples do projeto

Na figura 1 é apresentado o comando “node .”, que é necessário para executar o programa

```
john@ASUS:/www/HellTriangle$ node .
HellTriangle

Input: [[6],[3,5],[9,7,1],[4,6,8,4]]
Maximum Total: 26
```

Figura 1: resultado do método aplicado a um caso de exemplo

2. Readability;

O código foi organizado nos seguintes três arquivos principais:

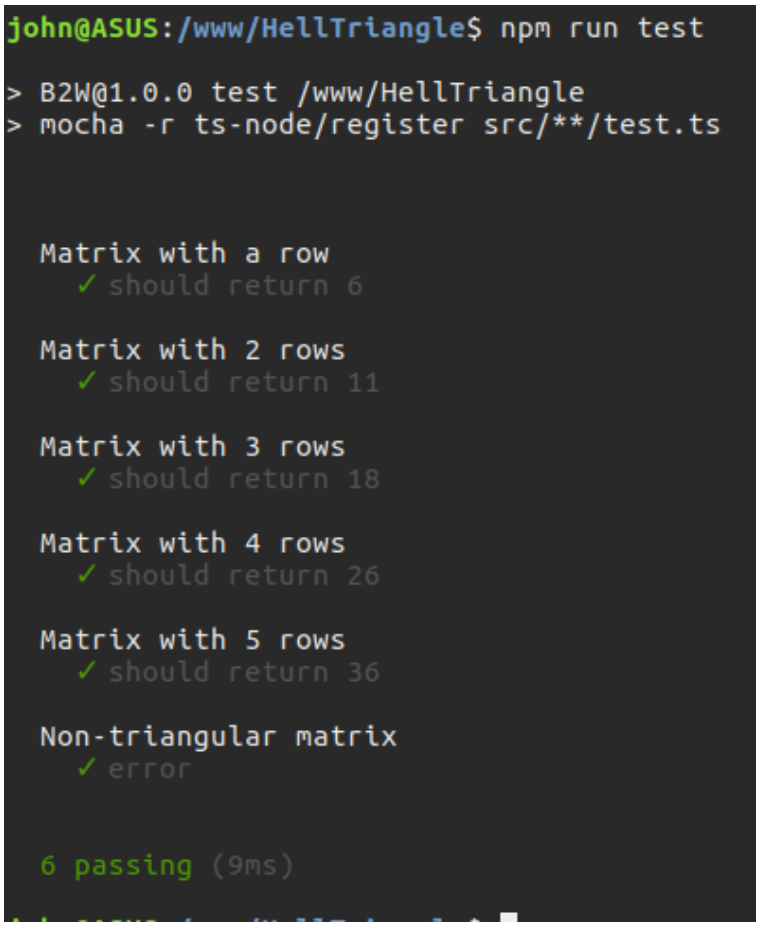
Triangle.ts: contem uma classe que permite modelar os atributos e métodos relacionados ao HellTriangle, a principal função é calculateMaxTotal.
Test.ts: definição dos 6 test variando as características dos valores de entrada do programa.

Main.ts: Logica principal do programa onde é definido a matriz do exemplo e excutado o código da classe HellTriangle

O código está documentado e escrito seguindo padrões para facilitar a legibilidade dele.

3. The automated test;

Foram implementados 6 unit test que permitiram avaliar diferentes tipos de matrizes de entrada do método, entre eles foi validado se input errado gerava uma excepção, permitindo validar a correta implementação do programa.



4. Execution time;

O tempo necessário para procurar o máximo total no caso de exemplo do desafio foi de 1 milissegundo em media.