Prezado candidato,

Para o desenvolvimento dos algoritmos propostos abaixo, não serão permitidos o uso de bibliotecas de terceiros, exceto para testes unitários.

Deve-se desenvolver os devidos códigos em Java, com a bibliotecas nativas do JEE.

Ao finalizar o desafio, os códigos devem ser comitados no GitHub/BitBucket.

Favor enviar o endereço público dos mesmos para [redacted]

Um equipamento de inspeção industrial é responsável por identificar defeitos em uma estrutura metálica, para tanto, o mesmo identifica as regiões em sua superfície cuja a rugosidade ultrapassa determinado limite. As peças tem um tamanho arbitrário de NxN, sendo sempre quadradas. O equipamento então divide a área escaneada em quadrados de 1mm2, cada ponto então compõe uma matriz bidimensional, marcando com “1” as regiões falhas.

Seu trabalho é receber estes dados e processá-los para fornecer as seguintes informações:

1. Área total que tem sua rugosidade marcada como alta (em pixels);
2. Identificar e contar quantas ‘manchas’ existem na superfície. É dito que dois pontos adjacentes pertencem a mesma mancha se ambos compartilham arestas, não bastando que compartilhem vértices. Isso é, apenas o pixel imediatamente acima, abaixo, direita e esquerda, desconsiderando as diagonais;
3. Calcular a média da área das manchas apresentados na imagem (em pixels);
4. Retornar a área da maior mancha (em pixels);

O equipamento enviará os dados em JSON e espera a resposta no mesmo formato, sendo os dados transmitidos compostos por uma array de tamanho N, contendo por sua vez arrays de mesmo tamanho. Cada elemento desta segunda array será sempre um inteiro, “1” para quando a região é de alta rugosidade e representa uma mancha, “0” para quando a região está OK. As respostas são um objeto, contendo os seguintes campos: { “total_area”: 10, “number_of_spots”: 1, “spots_average_area”: 10.0, “biggest_spot_area”: 10 } O equipamento manda as requisições via HTTP Post e

nenhum estado precisa ser mantido entre cada chamada. Exemplos de chamadas e respostas:

curl -X POST \ -d '[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]' http://localhost:8080/spot_check
R: {"total_area": 0, "number_of_spots": 0, "spots_average_area": 0.0, "biggest_spot_area": 0}

curl -X POST \ -d '[[1, 1, 0, 0], [1, 1, 0, 0], [0, 0, 1, 1], [0, 0, 1, 1]]' http://localhost:8080/spot_check
R: {"total_area": 8, "number_of_spots": 2, "spots_average_area": 4.0, "biggest_spot_area": 4}

curl -X POST \ -d '[[1, 0, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1], [0, 0, 1, 1]]' http://localhost:8080/spot_check
R: {"total_area": 5, "number_of_spots": 3, "spots_average_area": 1.66666, "biggest_spot_area": 3}

curl -X GET \ -d '[[1, 0, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1], [0, 0, 1, 1]]' http://localhost:8080/spot_check
R: Invalid Method

curl -X POST \ -d '[[1, 0, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1], [0, 0, 1, 1, 1, 0, 1, 1, 0]]' http://localhost:8080/spot_check Invalid Matrix Format curl -X POST \ -d '[["0", 0.0, false, 0], ["1", 1.0, true, 0], [0, 0, 0, 0], [0, 0, 0, 0]]' http://localhost:8080/spot_check **R: Invalid Matrix Format**