João Marcos Couto
Matemática Discreta (2017.2)
Documentação: Soma Máxima; Quadrado Magico;

Soma Máxima

No intuito de criar um algoritmo capaz de resolver a proposta, optei pela busca da solução através da comparação de todas as somas possíveis dentro do vetor recebido. O programa resultante armazena as maiores somas para diferentes tamanhos de sub-vetores e posteriormente compara estas somas a fim de determinar qual delas é numericamente maior, apresentando também os índices de começo e fim do sub-vetor cuja soma dos elementos resulta nesta maior soma.

Funcionamento:

O programa recebe inicialmente o valor "n" do tamanho do vetor. Em seguida, recebe também os elementos que compõem este vetor de tamanho n. Uma vez tendo feito isso, a estrutura principal do programa passa a escolher índices de começo no vetor (i) e compara as somas obtidas com a maior soma até então obtida para sub-vetores de tamanho (j – i + 1), onde j é o índice de termino do sub-vetor em análise. Caso esta comparação retorne valor positivo, então a soma deste vetor é armazenada como a maior soma obtida para sub-vetores de seu tamanho. Ademais, seus índices i e j de começo e fim são também armazenados.

Uma vez tendo feito isso, o programa passa então a comparar todas as somas obtidas para diferentes tamanhos de sub-vetores entre si, imprimindo então o valor da maior delas assim como seus indicies de começo e fim.

Quadrado magico

A fim de criar um algoritmo capaz de gerar quadrados mágicos de lado 3 a 5, pesquisei por metodologias de resolução conhecidas. Logo descobri a impossibilidade de se computar as soluções eficientemente por um método único que resolvesse quadrados de qualquer tamanho. Em minhas pesquisas, encontrei dois papers que foram a base para o programa. O primeiro, "On the Construction of Doubly Even Order Magic Squares" (https://arxiv.org/ftp/arxiv/papers/1402/1402.3273.pdf) possibilitou a criação de um algoritmo capaz de gerar quadrados mágicos não só de lado 4, mas qualquer quadrado com lado múltiplo de 4. A função que aplica o método descrito nele está nomeada como "JacobMurugan" em homenagem aos autores do paper. O segundo paper, "Matrix Properties of Magics Squares" (http://faculty.etsu.edu/stephen/matrix-magic-squares.pdf), possibilitou a criação de uma função capaz de gerar um quadrado magico ímpar, e não apenas dos impares 3 ou 5, utilizando um método denominado pelo autor como "Hindu". Este é também o nome da função que aplica ele no programa.

A parte comum da minha implementação, para todos os casos, inicialmente cria uma matriz n por n e preenche todas as células com 0.

O primeiro método utilizado, "JacobMurugan", para casos de lados múltiplos de 4, recebe a matriz n por n, e navega a matriz de tal forma a visitar cada um de seus blocos 4 por 4. Para cada bloco, ele preenche as diagonais com seus respectivos números de célula. A numeração utilizada refere-se à matriz como toda, e não apenas o bloco 4x4. Isto é, a celular C(1,1) recebe 1, C(1,2) recebe 2 e assim por diante até a C(n,n) que recebe n^2. Em seguida, tendo preenchido todas as diagonais, a função passa então a preencher o resto das células com o valor da expressão (n*n – c) onde c é o número da célula no padrão previamente descrito.

O segundo método utilizado, "Hindu", para casos de lados impares, recebe a matriz n por n e atribui à celula C((n+1/2),1 = a celular central da primeira linha) o valor 1. Posteriormente, atribui 2 à célula da ultima linha na coluna à direita da central. Tendo feito isso, a função navega em direção "nordeste" da matriz atribuindo os números naturais 3 em diante, incrementando a cada célula visitada

até atingir a coluna mais à direita. Uma vez lá, ela salta para a linha superior, porém na coluna mais à esquerda da matriz, continuando a atribuição até atingir uma célula já preenchida. Uma vez lá, o próximo numero da sequencia de atribuição deve ser colocada na célula abaixo à célula mais recentemente preenchida. A partir dai, continua-se a navegação em direção à "nordeste" recursivamente conforme definido acima, até estarem preenchidas todas as células da matriz.

A fim de enriquecer a implementação e possibilitar a geração de quadrados mágicos de lado impares diferentes para cada caso, utilizei a função rand() para determinar o local de inicio e a direção da navegação de forma a possibilitar a execução do algoritmo acima de forma simétrica. Ademais, para qualquer método aplicado, o programa por fim imprime a matriz gerada.