**Integrantes:** Gabriel Dias Barros, Matheus Santos Silva Pereira e Matheus Henrique Medeiros

- Existem diferentes formas de alocação de memória e acesso aos vetores em C/C++? Liste quais são.

R: Ponteiro(alocação dinâmica) e pelo índice de alocação semi estática.

- Python e Matlab realizam a verificação de acesso aos índices de um vetor? Isso tem alguma implicação de desempenho?

R: Sim, torna mais lento devido a uma utilização um pouco maior de recursos computacionais, mas que em alguns casos vale mais a pena ter esta verificação.

- O que seria *slicing*? C/C++ apresentam operação de *slicing* sobre vetores e matrizes?

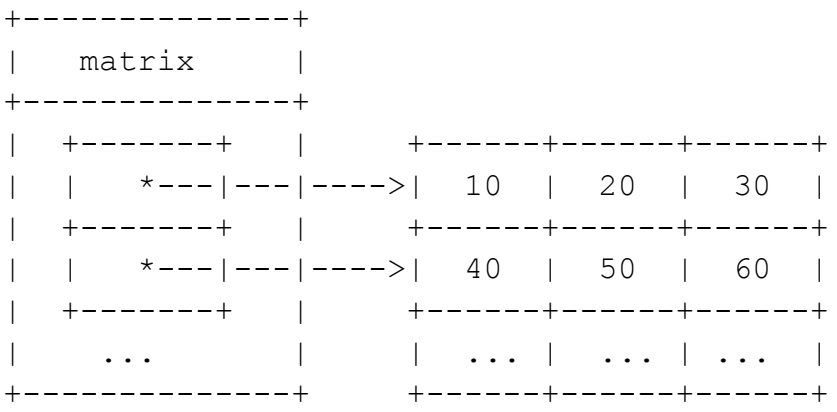
R: Slicing é uma funcionalidade poderosa da linguagem de programação Python e Matlab que permite extrair partes específicas de uma lista, string ou qualquer outra sequência de elementos. Com o Slice, podemos selecionar um intervalo de elementos de forma simples e eficiente, facilitando a manipulação e o processamento de dados. Em C e C++ não possui essa função nativamente, porém podem-se usar outros recursos para se fazer algo semelhante.

- No código abaixo, existe alguma diferença entre os procedimentos que visam levar a informação de um vetor *source* para um vetor *destination*?

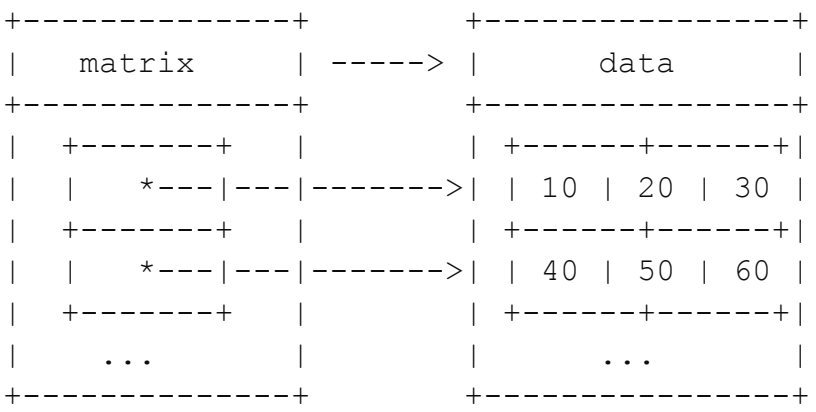
R: Sim, existe diferença. É executado a função memcpy para copiar parâmetro por parâmetro do source para o destination, já na outra parte referencia-se o vetor trocando o destination no índice [2] pelo número 3, sem copiar todos os índices no outro vetor, muito parecido com um strcpy.

- Desenhe como as matrizes dos dois códigos-fonte abaixo estão organizadas na memória. Diga qual é a organização utilizada pela linguagem C.

R: Primeiro Código:




Segundo Código:




- Explique porque os tempos de acesso pelas formas row-major e column major são diferentes no código em Matlab abaixo. Teste para matrizes com diferentes dimensões (100x100, 1000x1000 e etc...). Os tempos de acesso são alterados ao se modificar as dimensões das matrizes?

R: O texto explica que em MATLAB, os tempos de acesso diferem entre as formas "row-major" (linha por linha) e "column-major" (coluna por coluna) devido à maneira como os elementos da matriz são armazenados na memória e indexados. Na abordagem "row-major", os elementos de cada linha são armazenados sequencialmente na memória, resultando em acesso mais rápido devido à localidade espacial. Por outro lado, na abordagem "column-major", os elementos de cada coluna são armazenados sequencialmente, o que pode levar a acessos menos eficientes devido a mais saltos de memória. A escolha da abordagem mais eficiente depende do padrão de acesso aos dados na aplicação específica. Em muitos casos, a ordem "row-major" é preferida em MATLAB devido à eficiência em operações de linha comuns. Sim.