**Trabalho de Projeto e Análise de Algoritmos**

**(Selection-Sort e Merge-Sort)**

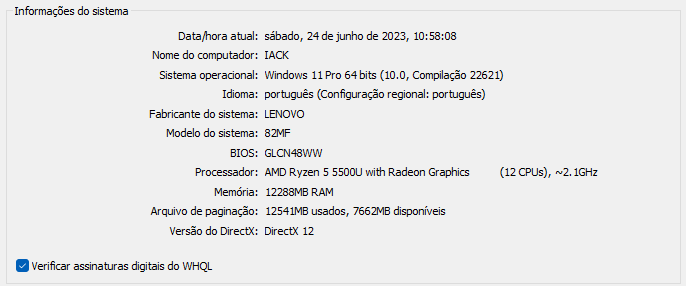
* **O que é um Algoritmo de ordenação?**

Um algoritmo de ordenação é um conjunto de instruções ou regras lógicas utilizadas para organizar um conjunto de elementos em uma determinada ordem. Ele permite que os dados sejam colocados em uma sequência específica, como em ordem crescente ou decrescente, facilitando a busca, a comparação e a manipulação desses dados. Existem vários algoritmos de ordenação, cada um com suas próprias características e eficiência. Mas no resumo serão abordados dois que são:

* Selection Sort (Ordenação por Seleção): Nesse algoritmo, o menor elemento é selecionado e colocado na primeira posição, em seguida o segundo menor elemento é selecionado e colocado na segunda posição, e assim por diante. O processo continua até que a lista esteja completamente ordenada.
* Merge Sort (Ordenação por Junção): Este algoritmo divide a lista em duas metades, ordena cada metade separadamente e, em seguida, combina as duas metades ordenadas para obter a lista final ordenada. Esse processo de divisão e combinação é repetido até que a lista esteja completamente ordenada.
* **Linguagem Utilizada:**

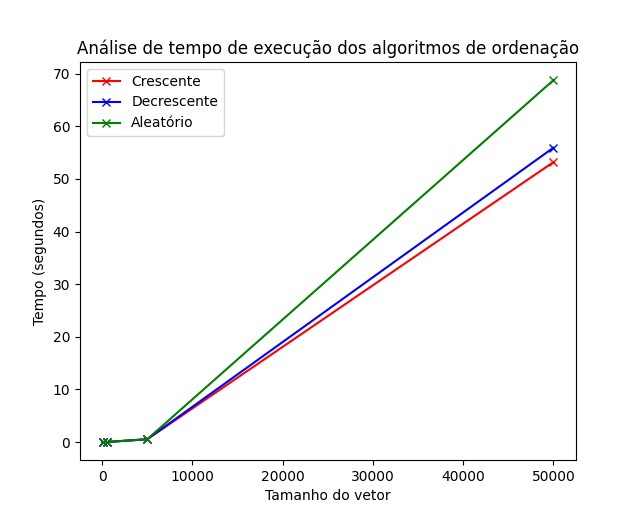
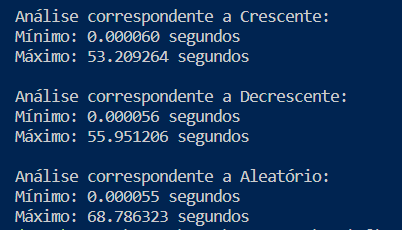
Nos programas, foram utilizados a linguagem computacional python, pois ele tem uma grande variedade de bibliotecas e a sintaxe é clara (simples e legível).

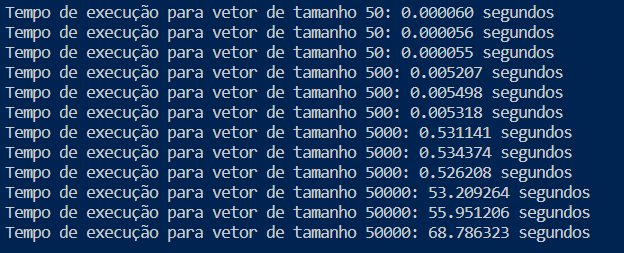
* **Configuração do Computador:**

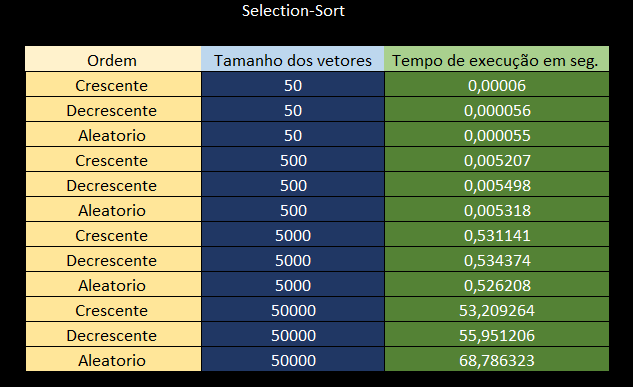


**Resultados do Selection-Sort**

* **Gráficos do Programa Selection-Sort:**

****





Observa-se que o tempo de execução para vetores de tamanho 50, nas seguintes ordens: crescente, decrescente e aleatória, apresentam os seguintes resultados: 0.000060 segundos, 0.000056 segundos e 0.000055 segundos, respectivamente. Pode-se inferir que as variações de tempo de execução entre essas ordens são mínimas, entretanto, é notável que a ordem crescente demanda um tempo ligeiramente maior em comparação com as demais.

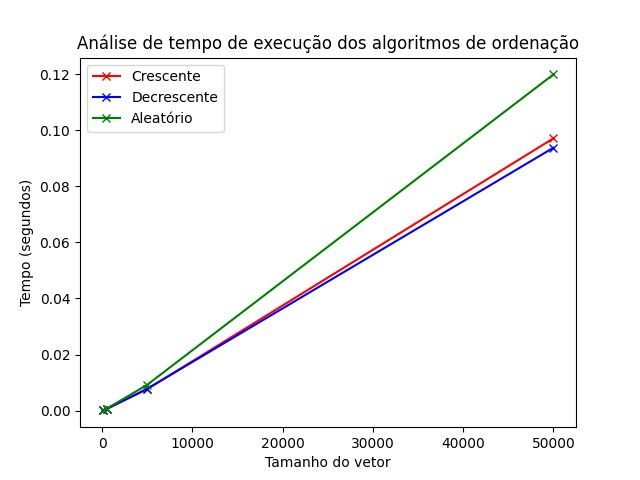
Ao analisarmos vetores de tamanho 500, observa-se um tempo de execução de 0.005207 segundos para a ordem crescente, 0.005498 segundos para a ordem decrescente e 0.005318 segundos para a ordem aleatória. Conclui-se que as diferenças de tempo de execução ainda são insignificantes para esse tamanho de vetor, sendo a ordem decrescente a que demanda mais tempo em comparação com as outras ordens.

Considerando vetores de tamanho 5000, o tempo de execução é de 0.531141 segundos para a ordem crescente, 0.534374 segundos para a ordem decrescente e 0.526208 segundos para a ordem aleatória. Observa-se que as diferenças de tempo de execução entre as ordens ainda são relativamente pequenas, sendo a ordem decrescente a que apresenta maior tempo de execução.

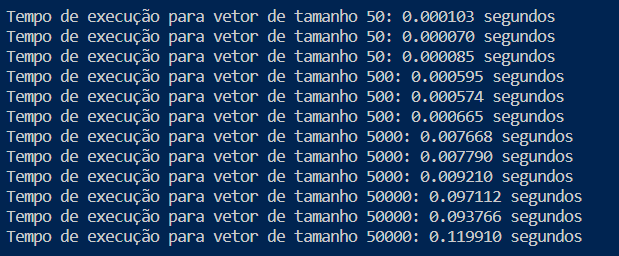
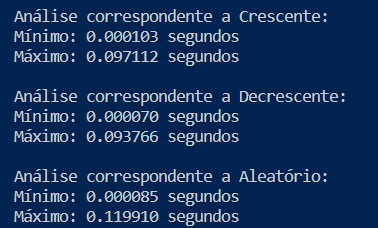
Na última análise, com vetores de tamanho 50000, o tempo de execução é de 53.209264 segundos para a ordem crescente, 55.951206 segundos para a ordem decrescente e 68.786323 segundos para a ordem aleatória. Nesta análise, nota-se que as diferenças de tempo de execução aumentaram um pouco, sendo a ordem aleatória a que demanda mais tempo em comparação com as outras ordens.

Após considerar todas as análises de ordem, tamanho e tempo, conclui-se que as diferenças de tempo de execução entre vetores em ordem crescente, decrescente e aleatória tendem a ser relativamente pequenas. No entanto, é importante ressaltar que tais diferenças podem se tornar mais perceptíveis em vetores de tamanhos muito grandes.

**Resultados do Merge-Sort**



* **Gráficos do Programa Merge-Sort:**





Observa-se que o tempo de execução para vetores de tamanho 50, nas seguintes ordens: crescente, decrescente e aleatória, apresentam os seguintes resultados: 0.000103 segundos na ordem crescente, 0.000070 segundos na ordem decrescente e 0.000085 segundos na ordem aleatória, respectivamente. Verifica-se que a diferença de tempo de execução entre as ordens não é insignificante, sendo a ordem crescente a que demanda mais tempo.

Ao analisar vetores de tamanho 500, observa-se um tempo de execução de 0.000595 segundos para a ordem crescente, 0.000574 segundos para a ordem decrescente e 0.000665 segundos para a ordem aleatória. Conclui-se que as diferenças de tempo de execução diminuíram um pouco em relação aos vetores de tamanho 50, sendo a ordem aleatória a que apresenta maior tempo de execução.

Considerando vetores de tamanho 5000, o tempo de execução é de 0.007668 segundos para a ordem crescente, 0.007790 segundos para a ordem decrescente e 0.009210 segundos para a ordem aleatória. Observa-se que as diferenças de tempo de execução entre as ordens crescente e decrescente são relativamente pequenas, porém a ordem aleatória apresenta uma diferença mais significativa em relação ao tempo de execução, sendo a que demanda mais tempo.

Na última análise, com vetores de tamanho 50000, o tempo de execução é de 0.097112 segundos para a ordem crescente, 0.093866 segundos para a ordem decrescente e 0.119910 segundos para a ordem aleatória. Nesta análise, nota-se que as diferenças de tempo de execução entre as ordens crescente e decrescente são pequenas, enquanto a ordem aleatória continua apresentando uma diferença relativamente maior, sendo a que demanda mais tempo em comparação com as outras ordens.

Após considerar todas as análises de ordem, tamanho e tempo, podemos concluir que as diferenças de tempo de execução entre vetores em ordem crescente, decrescente e aleatória podem variar dependendo do tamanho do vetor. Em tamanhos pequenos, as diferenças podem ser mínimas, mas à medida que o tamanho do vetor aumenta, o desempenho do Merge Sort começa a se destacar em relação a algoritmos de ordenação menos eficientes.