Relatório Projeto 4.2 AED 2020/2021

Nome: Nº Estudante:

TP (inscrição): *Login* no *Mooshak:*

Nº de horas de trabalho: *H* *Aulas Práticas de Laboratório:* *H* *Fora de Sala de Aula:* *H*

(A Preencher pelo Docente) CLASSIFICAÇÃO:

Comentários:

Registar os tempos computacionais do QS e das 4 variantes selecionadas do QS+IS para os diferentes tipos de sequências. O tamanho das sequências (N) deve ser crescente e terminar em 10,000,000. Só deve ser contabilizado o tempo de ordenamento. Exclui-se o tempo de leitura do input e de impressão dos resultados. Devem apresentar e discutir as regressões para a melhor variante em cada tipo de sequência.

Gráfico para SEQ_ALEATORIA

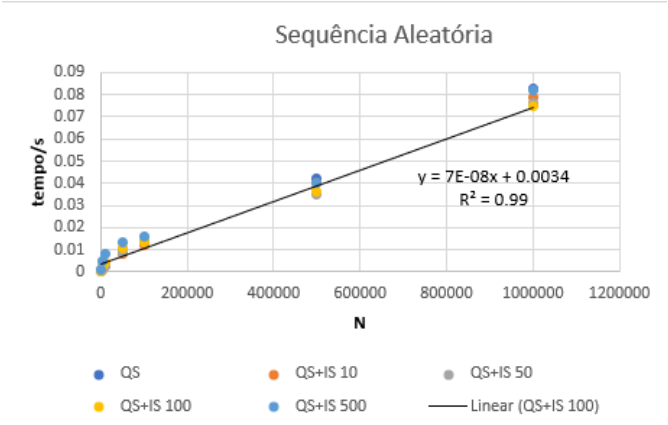


Gráfico para SEQ_ORDENADA_DECRESCENTE

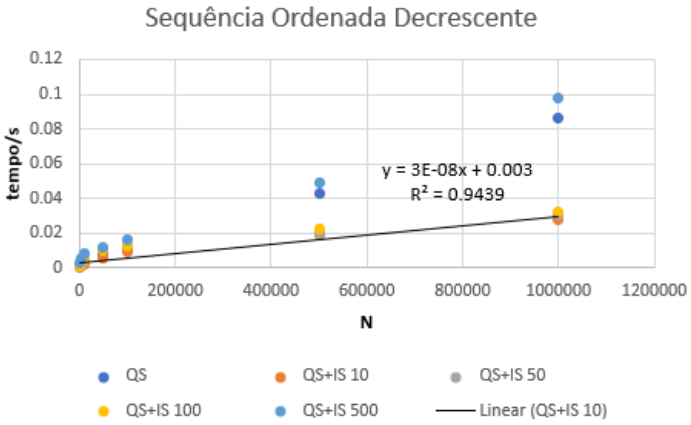


Gráfico para SEQ_QUASE_ORDENADA_1%

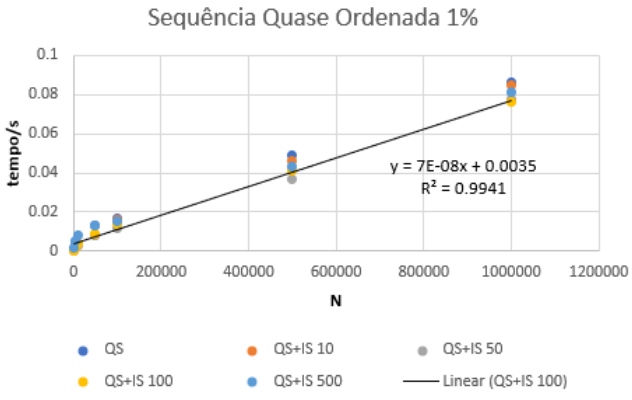
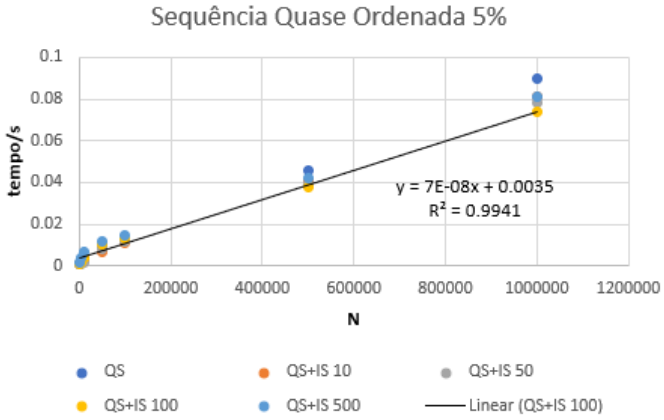


Gráfico para SEQ_QUASE_ORDENADA_5%



Análise dos resultados:

Analisando os 4 gráficos verifica-se claramente uma tendência linear para qualquer um dos algoritmos, em qualquer uma das situações. Isto deve-se à escolha da localização do *pivot,* que no código produzido se encontra no centro do array, levando assim ao seu melhor caso com complexidade O(N*log(N)) que apresenta um gráfico quase linear.

Reparamos então que na grande maioria a versão QS+IS 100 (k=100) será melhor perdendo apenas ligeiramente para o QS+IS 10 (k=10) no caso de termos uma sequência ordenada decrescente. Ao longo deste trabalho algo que se vai apercebendo na versão do QS+IS (qualquer uma delas), dependerá tanto do k escolhido como do tamanho do array a ser ordenado, pois se tivéssemos um k>N , ou seja, um k maior que o tamanho do array, só iria ser aplicado um insertion sort e nenhuma vez um quicksort. Ora visto que o insertion sort tem uma complexidade O(N^2) naturalmente iria demorar mais.

Assim sendo um dos motivos para o QS+IS 100 será o melhor entre os apresentados será por ter um k digamos mediano em relação ao tamanho dos array testados.