# Relatório Projeto 4.4 AED 2020/2021

Nome: Miguel Pedroso Nº Estudante: 2019218176

TP (inscrição): 5 *Login* no *Mooshak: AED2019218176*

Nº de horas de trabalho: 3 *H* *Aulas Práticas de Laboratório:* 2 *H* *Fora de Sala de Aula:* 1 *H*

**(A Preencher pelo Docente) CLASSIFICAÇÃO:**

**Comentários:**

Registar os tempos computacionais das variantes em consideração 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.

**Gráfico para SEQ_ALEATORIA**

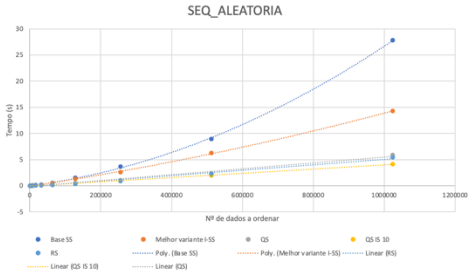



**Gráfico para SEQ_ORDENADA_DECRESCENTE**

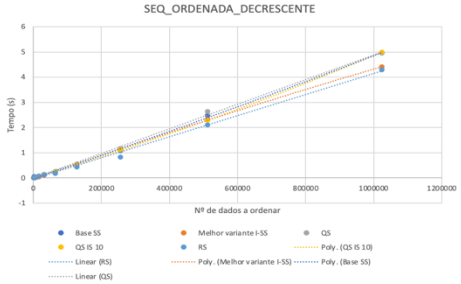



**Gráfico para SEQ_QUASE_ORDENADA_1%**

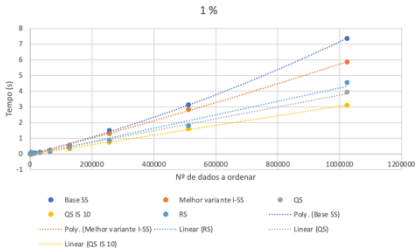
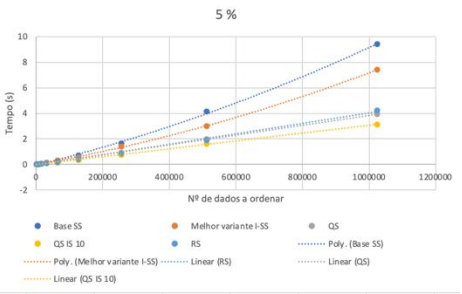



**Gráfico para SEQ_QUASE_ORDENADA_5%**




**Sequência de incremento ou regra de incremento do I-SS para cada tipo de sequência:**

Sequência de incremento de Papernov & Stasevich: 2^k + 1 (1, 3, 5, 9, 17...)

**Análise dos resultados considerando também a complexidade espacial dos algoritmos:**

Tal como seria espectável, o BSS e a sua melhor variante tiveram os piores desempenhos em quase todos os cenários, uma vez que têm complexidade O(n^5/3) e O(n^3/2), respetivamente. Os outros três algoritmos tiveram desempenhos muito semelhantes entre si, sendo que a variante QS IS 10 do Quicksort foi o melhor algoritmo na maioria dos cenários. Ambas as variantes do QS possuem complexidade O(n log n), sendo que a variante que utiliza Insertion Sort para valores baixos foi a mais eficiente das duas. O Insertion Sort, apesar de ter complexidade O(n^2) consegue ser mais rápido que o QS, com complexidade O(k*n log n), devido à constante k, que faz com que para valores baixos o suficiente, o Insertion Sort tenha menor complexidade que o QS. Já o Radix Sort, apesar de ter complexidade O(k*n), não conseguiu ser mais rápido que a melhor variante do QS.