**ICMC** - Instituto de Ciências Matemáticas e da Computação

**BCC** - Bacharelado em Ciências de Computação

**Aluno**: Bernardo Marques Costa

**Número** USP: 11795551

**Docente**: Leonardo Pereira

**Disciplina:** Laboratório de Introdução a Ciência da Computação II

**RELATÓRIO 4 - ORDENAÇÃO DE PALAVRAS**

**Introdução**

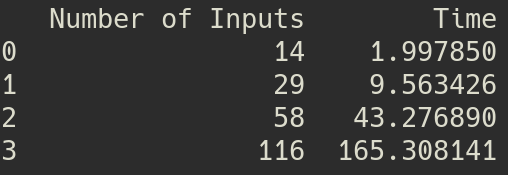
Neste trabalho é apresentado a comparação entre os algoritmos de ordenação: **bubblesort**, **insertion sort** e **merge sort** no que se refere a ordenação de matrizes (vetores de palavras).

É fornecido ao programa 1 dos 4 arquivos: 14k.txt, 29k.txt, 58k.txt e 116k.txt. Cada um contém um número total de palavras próximo ao valor descrito no título.

Para a parte do relatório, é feito a análise da média de 10 execução de cada um dos algoritmos para cada um das quantidades possíveis de inputs, comparando o valor do tempo e a complexidade dos algoritmos estudados.

**Bubblesort**

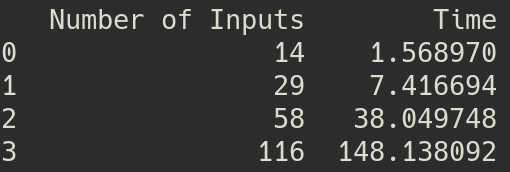
Após executar o programa, obtemos um arquivo CSV que contém a seguinte tabela:

Como podemos ver, o número de inputs cresce, aproximadamente, de 2 em dois.

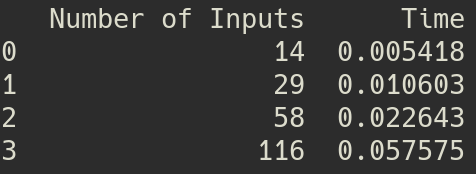
Considerando a complexidade do algoritmo bubblesort de , podemos observar que o tempo também cresce proporcionalmente com

**Insertion sort**

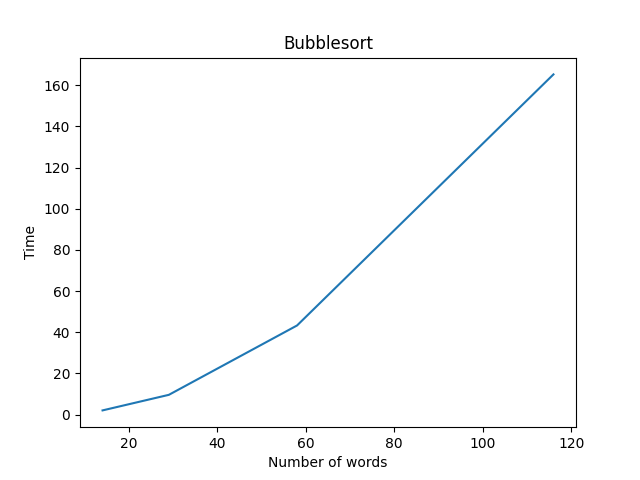
Após executar o programa, obtemos a seguinte tabela de dados:

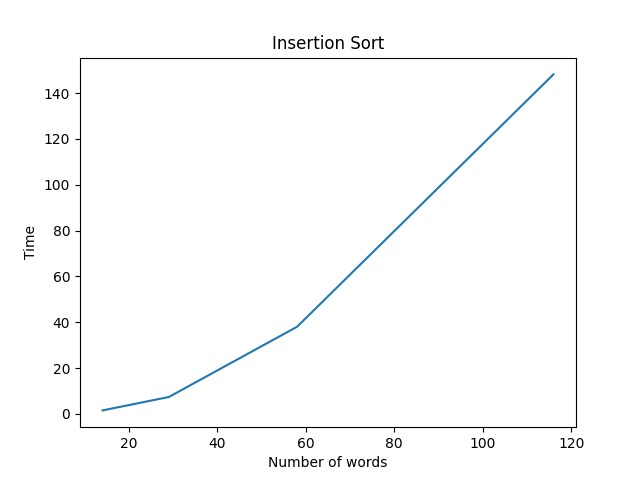


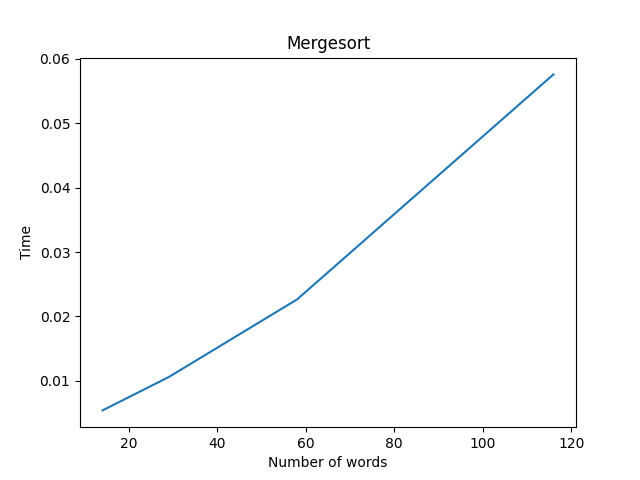
O algoritmo insertion sort possui uma complexidade e tempos de execução muito semelhantes ao bubblesort, tendo uma complexidade , sendo sutilmente mais otimizado

**Mergesort**

Ao contrário dos algoritmos anteriores, podemos observar que o tempo de execução é muito menor, e seu crescimento não é exponencial. De fato, como podemos observar pela complexidade do mergesort, temos como a função big O do algoritmo de merge sort

**Gráficos**





**Conclusão**

Como podemos observar a partir do tempo e da construção dos gráficos, temos uma leve otimização do bubble sort para o insertion sort e uma gigantesca diferença de tempo entre os dois primeiros algoritmos e o merge sort, em que o valor correspondente ao tempo de execução cai de vários segundos para menos de 1 segundo.

Assim, o método de divisão e conquista do mergesort aumenta consideravelmente a eficácia de ordenação, em comparação aos métodos sequenciais estudados agora.