1. **Bubble Sort:**
    - O Bubble Sort apresentou os piores tempos de execução, com crescimento exponencial do tempo à medida que o tamanho do vetor aumentou.
    - Para 100.000 elementos, o tempo foi de aproximadamente 29 segundos, e para 400.000 elementos, ultrapassou 574 segundos.
    - Este resultado era esperado, dado o custo computacional do algoritmo.
2. **Insertion Sort:**
    - O Insertion Sort teve desempenho ligeiramente melhor que o Bubble Sort para vetores maiores.
    - Mesmo assim, o tempo de execução aumentou significativamente com o tamanho do vetor. Para 400.000 elementos, o tempo foi de cerca de 97 segundos.
    - Ele é mais eficiente para vetores parcialmente ordenados, o que não era o caso neste teste.
3. **Selection Sort:**
    - O Selection Sort teve desempenho melhor que o Bubble Sort e o Insertion Sort para vetores maiores, mas ainda foi ineficiente para tamanhos grandes.
    - Para 400.000 elementos, o tempo foi de cerca de 139 segundos, maior que o Insertion Sort.
    - Apesar de possuir o mesmo custo computacional, o Selection Sort realiza menos trocas, o que explica a leve vantagem em alguns casos.
4. **Quick Sort:**
    - O Quicksort apresentou um desempenho excelente, com tempos insignificantes mesmo para vetores grandes.
    - Para 400.000 elementos, o tempo foi de apenas 56 ms, graças ao seu custo médio.
    - Este resultado confirma que o Quicksort é um dos algoritmos mais eficientes para conjuntos de dados grandes e desordenados.
5. **Merge Sort:**
    - O Mergesort também apresentou desempenho excelente, próximo ao Quicksort, mas levemente mais lento.
    - Para 400.000 elementos, o tempo foi de 113 ms.
    - Assim como o Quick Sort, e Merge Sort tem custo, mas o seu tempo extra pode ser atribuído à necessidade de alocação de memória adicional.

Conclusões Gerais

1. **Algoritmos Ineficientes:**
    - Bubble Sort, Insertion Sort e Selection Sort são inadequados para vetores maiores que 10.000 elementos devido ao crescimento exponencial do tempo de execução.
    - Eles podem ser úteis para vetores muito pequenos (menores que 1.000 elementos) ou casos específicos, como o Insertion Sort em vetores parcialmente ordenados.
2. **Algoritmos Eficientes:**
    - Quick Sort e Merge Sort foram claramente superiores, com tempos de execução muito baixos para todos os tamanhos de vetor.

- Entre os dois, o Quicksort foi o mais rápido, enquanto o Merge Sort teve um desempenho consistente e previsível.