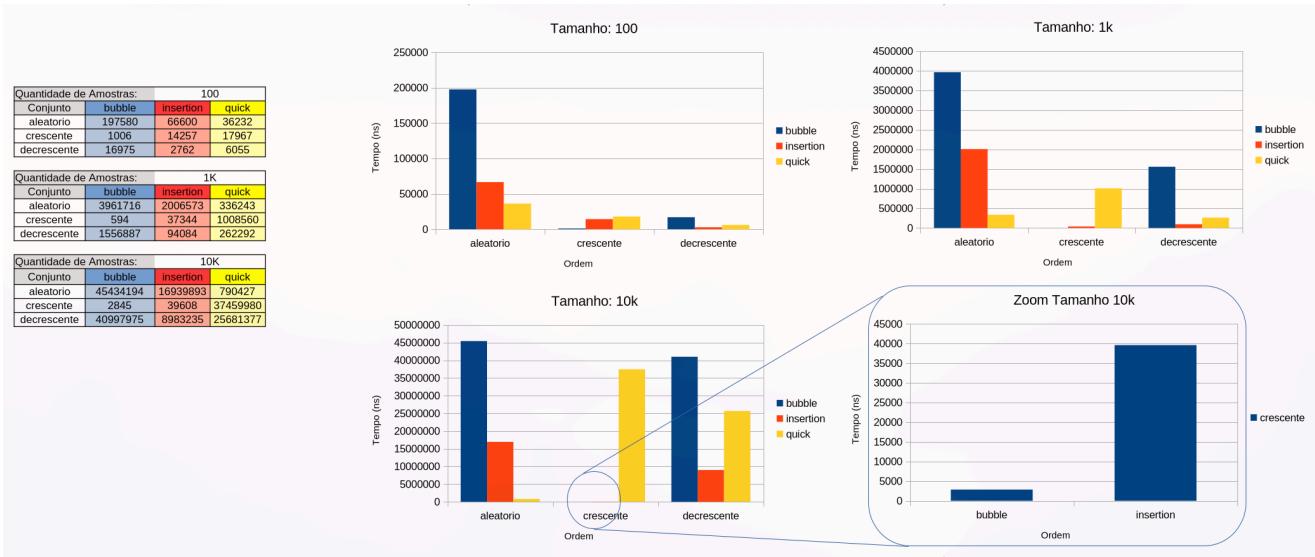


Análise dos Resultados e Explicação



A análise foi feita com base na média de três execuções diferentes, com intervalos de cinco minutos entre cada uma, para minimizar interferências como o cache do computador.

Cenário 1 — Dados aleatórios

Quando os dados estavam em **ordem aleatória**, o algoritmo QuickSort apresentou o melhor desempenho. Justificativa: nesse caso, QuickSort consegue dividir o problema de maneira bem equilibrada, já que não há padrão no conjunto.

Cenário 2 — Dados já em ordem crescente

Quando os dados estavam pré-ordenados em **ordem crescente**, o algoritmo Bubble Sort teve o menor tempo de execução. Justificativa: como os elementos já estavam ordenados, Bubble Sort praticamente não precisa fazer nada e poucas ou nenhuma troca são necessárias, o que o torna muito rápido neste caso. Já QuickSort “sofre” porque, se seu pivô for mal escolhido e ele acaba fazendo particionamentos muito desequilibrados, o que o torna lento.

Cenário 3 — Dados em ordem decrescente

Quando os dados estavam em **ordem decrescente**, o algoritmo Insertion Sort apresentou o melhor desempenho entre os três. Justificativa: Insertion Sort consegue lidar de forma bastante direta com esse tipo de entrada porque vai “inserindo” cada elemento na posição correta num conjunto que já tem uma estrutura previsível.