**An analysis of the complexity of the implementation refers to an examination of the computational performance of the GaussSolve algorithm in terms of time and space requirements. This analysis helps to understand how the algorithm's efficiency scales with the size of the input (the matrix size and the number of equations). The two primary aspects to consider are time complexity and space complexity.**

**1. \*\*Time Complexity:\*\* Time complexity describes how the running time of the algorithm grows as the size of the input increases. In the case of GaussSolve, we typically focus on the worst-case time complexity, which is often related to the number of operations performed.**

**- Partial Pivoting: O(n^2), where n is the number of rows or columns in the matrix.**

**- Forward Elimination: O(n^3), where n is the number of rows or columns in the matrix. This is usually the most time-consuming part.**

**- Backward Substitution: O(n^2), where n is the number of rows or columns in the matrix.**

**The overall time complexity of the GaussSolve algorithm is typically dominated by the forward elimination step, resulting in an O(n^3) time complexity.**

**2. \*\*Space Complexity:\*\* Space complexity refers to the amount of memory required by the algorithm as a function of the input size. This includes both the input data and any additional data structures used during the computation.**

**- The original matrix A and vector b each require O(n^2) space (assuming an n x n matrix and a vector of length n).**

**- Additional space is used for intermediate calculations, but they do not dominate the space complexity. The total space complexity is still O(n^2).**

**It's important to note that the space complexity is usually less of a concern in practice compared to the time complexity. The GaussSolve algorithm is known to have a cubic time complexity, which means that its running time grows rapidly with the size of the system of equations. For large systems, it might become computationally expensive.**

**In summary, when analyzing the complexity of the implementation, we find that GaussSolve has a cubic time complexity and quadratic space complexity. Understanding these complexities helps assess the algorithm's efficiency and make informed decisions when applying it to solve systems of linear equations, especially for large or computationally intensive problems.**So, in Gaussian elimination, the forward elimination step is usually the more time-consuming part of the process, as it involves more complex operations. Once the matrix is in row-echelon form, the backward substitution step is relatively straightforward and faster.