Report for Programming Assignment 6: Linear Equation System Solver

Introduction

This report details the experience and process of developing a C++ program to solve a system of linear equations using matrices and vectors with class inheritance. The project includes implementing the `Matrix` class and its derived classes `SMatrix` and `Vector`, as well as a program that utilizes these classes to solve and verify linear equations using Cramer's rule.

Project Structure

The project consists of the following files:

- `matrix.h` and `matrix.cpp`: Define and implement the `Matrix` class.

- `smatrix.h` and `smatrix.cpp`: Define and implement the `SMatrix` class, derived from `Matrix`.

- `vector.h` and `vector.cpp`: Define and implement the `Vector` class, derived from `Matrix`.

- `linear_equation_system_solver.cpp`: The main application that generates random matrices and vectors, solves the linear equation system, and verifies the solution.

Development Experience

Initial Setup

The project started with defining the base class `Matrix`, which includes common matrix operations and member functions. The class was designed with constructors, destructors, and operator overloading to facilitate various matrix operations.

Implementation of `Matrix` Class

The `Matrix` class was implemented to handle basic matrix operations such as addition, subtraction, multiplication, and assignment. Key considerations included dynamic memory allocation for the matrix elements and ensuring proper resource management through constructors, destructors, and copy constructors.

Derived Classes: `SMatrix` and `Vector`

The `SMatrix` class was derived from `Matrix` to represent square matrices and included a function to calculate the determinant using Gaussian elimination. This required careful handling of matrix elements during the elimination process to ensure accuracy.

The `Vector` class, representing column vectors, was also derived from `Matrix`. It included a function to replace a specific column of a matrix with the vector, which was crucial for implementing Cramer's rule.

Main Application

The main application, `linear_equation_system_solver.cpp`, was developed to:

1. Generate random coefficient matrices and constant vectors.
2. Print the system of linear equations in a readable format.
3. Solve the linear equation system using Cramer's rule.
4. Verify the solution by substituting it back into the original equations.

Challenges and Solutions

Handling Determinants

One of the main challenges was accurately calculating the determinant of matrices, especially when dealing with floating-point precision. This was addressed by carefully implementing Gaussian elimination and ensuring row swaps were correctly handled.

Memory Management

Ensuring proper memory management was crucial. Dynamic allocation and deallocation of matrix elements were handled using custom functions `allocateMatrix` and `deallocateMatrix` to avoid memory leaks.

Formatting Output

Another challenge was ensuring the output format matched the provided example exactly. This required precise control over the output stream, using manipulators like `setw` and `setprecision` to align numbers correctly and maintain consistent formatting.

Debugging and Testing

Extensive debugging and testing were conducted to ensure the program's correctness. Intermediate values, such as determinants and matrix replacements, were printed to verify each step of the algorithm. This helped identify and resolve issues with the determinant calculation and matrix operations.

Conclusion

The development of the linear equation system solver was a comprehensive exercise in object-oriented programming, matrix operations, and numerical methods. The project reinforced the importance of careful memory management, precise floating-point operations, and thorough testing. The final program successfully generates, solves, and verifies systems of linear equations, providing a valuable tool for mathematical computations.