# Development Report: Quadratic Equation Solver with Complex Numbers

D1262028 李皓鈞

## Introduction:

The objective of this report is to detail the development process of the assignment solution for solving quadratic equations with complex numbers in the C programming language. The assignment required the implementation of complex number operations and the development of a program to solve quadratic equations and verify the roots.

## Understanding the Requirements:

The first step in the development process was to thoroughly understand the requirements

specified in the assignment. This included:

Defining a complex number data type and implementing operations for addition, subtraction, multiplication, division, and absolute value.

Developing a program to solve quadratic equations using the complex number data type and operations.

Verifying the roots of the quadratic equation by substituting them back into the equation and checking within a certain precision.

Designing the Solution:

Based on the requirements, I designed the solution architecture, which included:

Defining the complex number data type using a

struct in C.

Implementing functions for complex number operations according to the specified arithmetic rules.

Designing the main program to prompt the user for coefficients, solve the quadratic equation, and verify the roots.

Implementation:

The implementation phase involved translating the design into actual code. This included:

Creating the header file complex_D1262028.h to declare the complex number data type and function prototypes.

Implementing the complex number operations in the source file complex_D1262028.c.

Writing the main program

quadratic_equation_verifier.c to solve quadratic equations using complex numbers and verify the roots.

Testing:

After implementing the solution, extensive testing was conducted to ensure correctness and reliability. This involved:

Testing individual complex number operations to verify their accuracy.

Testing the main program with various input coefficients to ensure correct computation and verification of roots.

Handling edge cases such as when the discriminant is zero, positive, or negative to ensure robustness.

Documentation:

Finally, documentation was prepared to provide clear instructions on how to compile, run, and use the program. This included:

Adding comments to the source code for clarity and maintainability.

Writing a development report (like this one) to explain the development process and rationale behind design decisions.

Conclusion:

The development of the assignment solution involved understanding the requirements, designing the solution architecture, implementing the code, testing for correctness, and documenting the process. By following a systematic approach, the solution was successfully developed, meeting all the specified requirements.