Enhancement Two: Algorithms and Data Structure

Takhmina Rahmatova

Southern New Hampshire University

CS-499-11109-M01

Professor Ramsey Kraya

07/27/2025

**Artifact Description**

This artifact is a C++ program made in my CS-405 class that performs a basic encryption and decryption operation using the XOR bitwise operator. The program reads text data from a file, encrypts it using a provided key, and writes the encrypted output to another file. Then it decrypts the encrypted text back to its original form and saves it to a third file.

The core logic is located in a function that loops over the characters of the source string, applying an XOR transformation using the encryption key. The original design used a procedural approach with multiple global functions and a hardcoded key. The program was initially functional but lacked structure, user interaction, and error handling. The main function contained most of the logic, which limited modularity and reuse.

**Justification for Inclusion**

I selected this artifact because it demonstrates foundational algorithmic thinking and the use of standard data structures. The main algorithm shows how to apply transformations over a string using a loop and modular arithmetic, which is directly related to algorithmic principles. The artifact also incorporates file input/output, string parsing, and basic validation — all of which involve managing data structures like std::string, std::ifstream, and std::ofstream.

The enhancements made to this project reflect my growth in designing reusable and secure code. I replaced the procedural approach with an object-oriented design using a Cipher class, added exception handling for file operations, and replaced the hardcoded key with user input. These changes showcase my ability to write flexible, user-friendly programs that follow modern software design practices.

**Planned and Performed Enhancements**

Originally, the program had several limitations:

* The encryption key was hardcoded.
* There was no user input validation.
* The structure was procedural, with little reuse potential.
* File I/O lacked error detection and handling.

To address these issues, I developed a plan to:

* Encapsulate the encryption/decryption logic in a class called Cipher.
* Prompt the user for the encryption key at runtime.
* Add error handling for missing or invalid files using exception handling.
* Refactor logic to make the code easier to test and maintain.

These enhancements improve the program’s modularity, security, and usability while maintaining its core logic.

**Alignment With Course Outcomes**

This enhancement demonstrates progress toward the following computer science program outcomes:

* Design and evaluate computing solutions using algorithmic principles: The XOR algorithm was applied efficiently, and its implementation was refactored into a reusable method.
* Use well-founded and innovative techniques to implement computing solutions: Object-oriented design and exception handling replaced procedural coding.
* Develop a security mindset: Avoid hardcoded keys, validated user input, and ensured reliable file operations.

**Reflection on the Enhancement Process**

Enhancing this artifact helped me better understand how to take a working algorithm and wrap it in a more robust, scalable design. I encountered challenges when restructuring the logic into a class while preserving its behavior. Debugging file read/write paths and validating input required attention to detail. However, the process also reinforced the importance of defensive programming and good coding practices. These enhancements reflect how I now approach problems with clarity and foresight, aiming not just for a working solution but one that’s maintainable, modular, and secure.

This enhanced artifact represents my skills in algorithms and data structures by demonstrating my ability to implement, refactor, and improve an encryption algorithm. It shows my growth from procedural programming toward object-oriented, user-centered, and error-resilient design, aligning directly with the learning outcomes of the computer science program.