**Item Tracking Program – Corner Grocer**

**Project Summary**

This project is a C++ program designed to help Corner Grocer track item purchases. The program reads a text file containing a list of purchased items, counts the frequency of each item, and provides an interactive menu for users to query and visualize the data. Additionally, the program saves the frequency data into a backup file, ensuring data integrity and future access.

**Strengths in the Project**

One of the strongest aspects of this project is the modular design, which enhances readability and maintainability. Functions such as getValidatedChoice() ensure robust input validation, preventing incorrect inputs from disrupting the program’s flow. Additionally, the use of the STL map container efficiently manages item frequencies, reducing complexity in data handling.

**Areas for Improvement**

While the program is well-structured, one potential enhancement could be implementing an exception handling for file operations. Although the program checks for file existence, adding try-catch blocks would provide more graceful error handling. Furthermore, optimizing file reading by using unordered\_map instead of map might slightly improve performance due to its faster average lookup time.

**Challenges and Solutions**

One of the more challenging aspects was implementing robust input validation. Ensuring that users entered only valid integers required careful handling of cin errors and clearing input streams. Using loops to continuously prompt users until they provided valid input helped overcome this issue.

To support development and debugging, I relied on:

* **C++ documentation** for STL functions
* **Online programming communities** such as Stack Overflow
* **IDE debugging tools** to step through code execution

**Transferable Skills**

The skills gained from this project—such as file handling, input validation, and data structure are highly transferable. These concepts are essential in cybersecurity, where logging, tracking, and secure data handling play a critical role. Additionally, the emphasis on modular programming and clear documentation aligns with industry’s best practices for software development.

**Code Readability and Maintainability**

To ensure the program remains readable and adaptable, I followed these practices:

* **Consistent indentation and naming conventions** to improve clarity
* **Commenting** on describing the purpose of each function and complex logic
* **Modular functions** to separate concerns and make future modifications easier

By implementing these practices, this project serves as a strong portfolio piece, showcasing proficiency in C++ programming, problem-solving, and software design.