**Corner Grocer** **Project**

A logo of a grocery bag

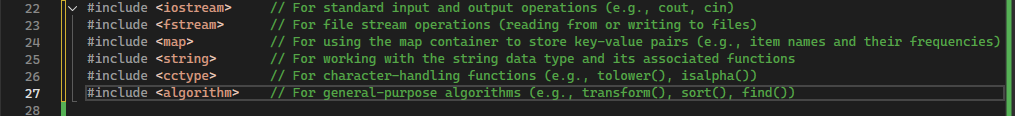
AI-generated content may be incorrect.

**Documentation: Corner Grocer Program**

The **Corner Grocer** program is designed to track the frequency of grocery item purchases based on data read from a text file. The program allows users to interact with a simple menu to:

1. Search for the frequency of a specific item (case-insensitive),
2. Display the frequency of all items,
3. Display a histogram (a visual representation) of item purchase counts,
4. Save the frequency data to a file,
5. Exit the program.

**Libraries and Includes**

**Class Overview: GroceryTracker and Its Internal LogicA screen shot of a computer program

AI-generated content may be incorrect.**

The GroceryTracker class handles reading item data from a file and tracking how often each grocery item is purchased. It uses a std::map called frequencyMap to count item occurrences efficiently. To support case-insensitive searches while keeping original formatting, it also uses a second map, caseMap, which links lowercase item names to their original versions.

* The class reads from a file named "CS210\_Project\_Three\_Input\_File.txt" and includes two helper functions: ToLower, which converts strings to lowercase for matching, and CapitalizeFirstLetter, which ensures proper display formatting. All data and functionality are kept private to maintain clean encapsulation, making this class the core of the program's item-tracking logic.

**Loading Item Data from File**

**A computer screen shot of a program

AI-generated content may be incorrect.**

*The LoadFromFile() function reads grocery items from the input file and populates both the frequencyMap and caseMap. It first attempts to open the file using an ifstream. If the file can't be opened, an error message is printed, and the function returns false.*

For each item read from the file:

* It converts the item to lowercase using ToLower() for consistent, case-insensitive handling.
* If the lowercase version isn’t already in caseMap, it adds the original item to preserve formatting.
* It then increments the item’s count in frequencyMap using the original-case version from caseMap.

The function returns true once all data is successfully read and processed.

**Input Validation Functions**

**A computer screen shot of a program code

AI-generated content may be incorrect.**

*These two helper functions ensure that user input is valid before the program proceeds:*

IsValidItemName(std::string item)

* Ensures the given item name is made up of only alphabetic characters.
* First, it trims any leading or trailing spaces.
* Then, it checks if the string is empty or contains any non-letter characters.
* Returns true only if the item name is valid.

IsValidMenuChoice(const std::string& choice)

* Checks if the user entered a valid menu option (choices 1 to 4).
* Confirms the input is a single digit and falls within the accepted range.
* Returns true for valid choices, otherwise false.

**Saving Frequency Data to File**

**A screen shot of a computer program

AI-generated content may be incorrect.**

*The SaveToFile() function writes the item frequency data to a file named frequency.dat.*

How It Works:

* Opens frequency.dat using an ofstream.
* If the file fails to open, an error message is displayed and the function exits.
* Otherwise, it loops through the frequencyMap and writes each item and its frequency on a new line.
* After writing, it closes the file and confirms the save to the user.

**Public Interface & User Interaction Methods**

A screenshot of a computer program

AI-generated content may be incorrect.

*This section defines the* ***public methods*** *of the GroceryTracker class, handling data loading and all user-facing functionality.*

**Key Methods:**

* **Constructor (GroceryTracker())  
  Automatically loads data from the file when an object is created. If loading fails, the program exits with an error message.**
* **LookUpFrequency()  
  Prompts the user to enter an item name, validates it, and displays how many times that item was purchased (case-insensitive match using caseMap).**
* **DisplayAllItemFrequencies()  
  Prints each item in the frequencyMap along with its purchase count.**
* **DisplayHistogram()  
  Outputs a simple text-based histogram where each asterisk (\*) represents one purchase of the item.**

**Main User Interaction Loop & Histogram Display**

**A computer screen shot of a program code

AI-generated content may be incorrect.**

This section includes the method DisplayHistogram() to show a text-based histogram and Run() to handle the main user interface, allowing the user to interact with the program.

Key Methods:

* DisplayHistogram()  
  Displays a histogram of item frequencies where each \* represents one purchase of a specific item. It loops through the frequencyMap and prints the item followed by a number of asterisks corresponding to how many times it was purchased.
* Run()  
  This is the main loop of the program. It presents a menu to the user, letting them choose actions such as:
  1. Look up the frequency of a specific item
  2. Display all item frequencies
  3. Display a histogram of item purchases
  4. Save item frequency data to a file
  5. Exit the program

The loop continues until the user chooses to exit (option "5"). It validates input and calls appropriate functions based on user choice.

**Main Program Execution & User Interaction**A computer screen shot of a program code

AI-generated content may be incorrect.

*The code snippet represents the entry point for the program and the user interface interaction. The Run() method handles the core logic for interacting with the user, while the main() function initializes the program.*

**Key Methods:**

* **Run()**  
  This method displays a menu to the user with multiple options (lookup, display frequencies, display histogram, save data, and exit). The loop continues until the user chooses to exit the program. It validates the input for menu choices, calls the appropriate functions (e.g., LookUpFrequency(), DisplayAllItemFrequencies(), DisplayHistogram(), SaveToFile()), and prints the corresponding output.
* **main()**  
  The main() function initializes an instance of the GroceryTracker class and starts the user interaction by calling tracker.Run(). Once the program finishes execution, the method returns 0 and exits.