James Lee

CS-210 23EW3

**Project Three - Documentation**

**Introduction**

The Corner Grocer needed to rearrange their produce section in the most effective layout for the customer, and they wanted to achieve this by analyzing the data of produce items sold on a given day to determine how often each item were purchased. This program was designed to analyze the text records of grocery items sold on a given day at the Corner Grocer.

**Design**

The program consists of the following classes, as well as other non-class member functions:

Classes

* InputOutput
  + Opens, reads, and closes input file
  + Opens, writes, and closes output file
* ItemVector
  + Creates vector list that stores unique produce items read from the input file
  + Sets and gets vector values
  + Sorts the vector alphabetically
  + Returns the vector size

Functions

* mainMenu()
  + Calls InputOutput class
  + Calls printMenu function in a loop
  + Loops main menu, and executes user-selected menu functions
* nCharString()
  + Returns a string of character c repeated n number of times
* printMenu()
  + Prints main menu and returns user menu selections

The UML class diagrams for classes implemented in this program are shown below.

|  |
| --- |
| InputOutput |
| - m\_inFS: ifstream  - m\_outFS: ofstream  - m\_fileItem: string  - m\_itemMap: map<string, int>  - i: int |
| + openInputFile(): int  + readInputFile(): void  + setMapValue(): void  + getMapValue(string item): int  + closeInputFile(): void  + openOutputFile(): void  + writeOutputFile(): void  + closeOutputFile(): void  + printList(): void  + printHist(): void |

|  |
| --- |
| ItemVector |
| - m\_itemList: vector<string> |
| + setVectorValue(int i, string fileItem): void  + getVectorValue(int i): string  + sortVector(): void  + vectorSize(): int |

**Functionality**

* The program’s Main() calls the mainMenu() function.
* The mainMenu() function creates an InputOutput class object that opens, reads, and closes the input file, then opens, writes (to output file *frequency.dat*, the vector list that holds the name of each unique produce item, and the map with names of produce items and sales count as the key-value pairs), then closes the output file.
  + Note: The program sorts the data (i.e. unique produce items) read from the input file in an alphabetical order. This results in an output of produce item list that allows users to more easily search for a specific item. See Fig. 3 and Fig. 4.
  + The purpose of the output file *frequency.dat* is for backing up the accumulated data read from the input file. This output file is created in the beginning of the program without user intervention.
* The mainMenu() function then calls the printMenu() function to print the main menu (see Fig. 1) and get the user menu choice, then executes one of the 4 menu function selected by the user:

**Menu option 1**: Look up and print the sales count for an item

* Gets produce item name from user then returns the item count from the itemMap (see Fig. 2)

**Menu option 2:** Print Summary of Item Sales Count

* Outputs every unique produce item sold on a given day along with associated item count for each item (see Fig. 3)

**Menu option 3:** Print Histogram of Item Sales Count

* Outputs a horizontal histogram of every unique produce item sold on a given day vs. item count (see Fig. 4)

**Menu option 4:** Exit the program

* Exits the program (see Fig. 5)
* This program validates user input for menu selection by catching runtime errors. See Fig. 6 for a screenshot of an example of input validation implementation.

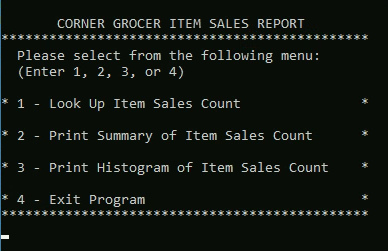


Fig. 1: The program’s main menu

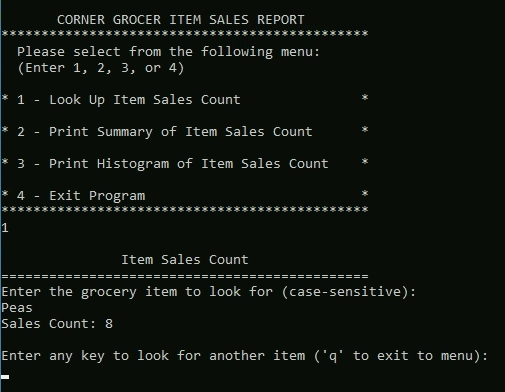


Fig. 2: Example of Menu Option 1

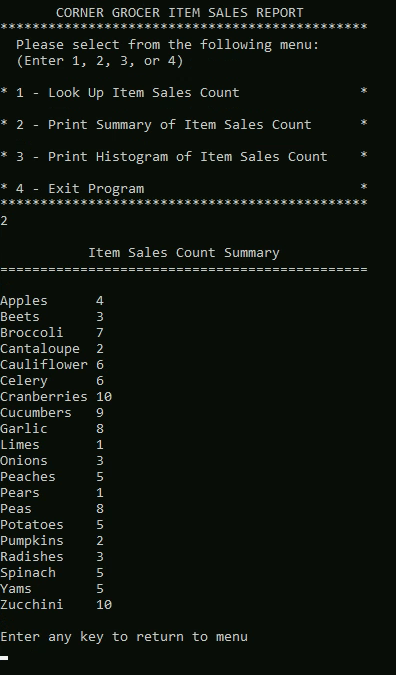


Fig. 3: Example of Menu Option 2



Fig. 4: Example of Menu Option 3

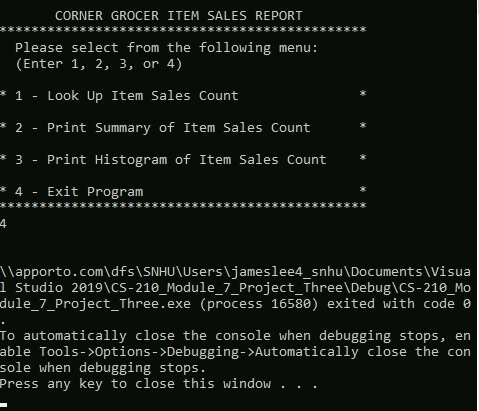


Fig. 5: Example of Menu Option 4

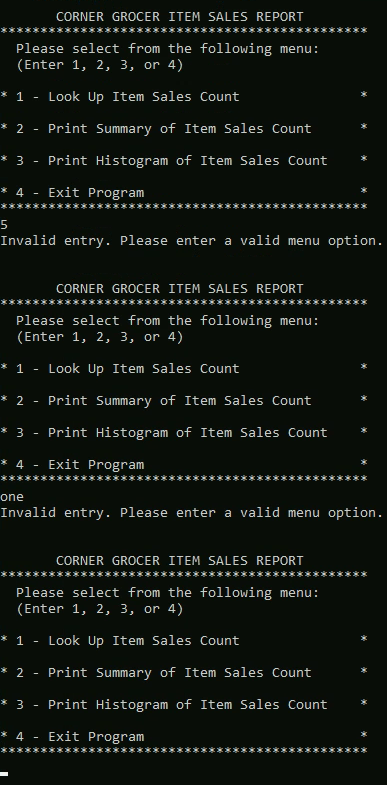


Fig. 6: Examples of User Input Validation