**Project 3 – Item Tracking for Corner Grocer**

4/11/2023 2:45 AM

—-----------------------------------------------------------------------------------------------------

**Program Pseudocode: Version 2**

**START PROGRAM:**

INPUT FILE: CS210\_Project\_Three\_Input\_File.txt

* Contains a list of items that have been bought throughout the day.

OUTPUT FILE: frequency.dat

* Receives output containing items and their sale frequency.

—------------------------------------------------------------------------------------------------------------------

Program checks to see if files are available.

If files are not available, the choice variable is set to 4 and the program terminates itself.

Graphical user interface, text, application

Description automatically generated

If files are available, the program runs.

OUTPUT Greeting message function



OUTPUT Menu option function.

Text

Description automatically generated

INPUT a menu choice (1-4) from the user.



If choice variable is equal to 4: (Missing file program termination)

1. If true, terminate the program because of missing files.
2. If false, continue to while loop.

While user input doesn’t equal 4, we will continue to loop the program. {

1. If user input is #1
   * 1. Input requested item.
     2. Function to find item and its frequency.
     3. Outputs item name and frequency value

Text

Description automatically generated

1. If user input is #2
   * 1. Function to find items and their frequency’s.
     2. Add spaces in front of items to lineup numbers and stars.
        1. Pick a constant base number and minus the current items length from it, then add that many spaces to the front of items name. example:
           1. 15 – Apples(6) = 9 pre spaces
     3. Output number aligned item names and frequency values.

Text

Description automatically generated

3. If user input is #3

1. Function to find items and their frequency’s.
2. Add spaces in front of items to lineup numbers and stars.
   1. Pick a constant base number and minus the current items length from it, then add that many spaces to the front of items name. example:
      1. 15 – Apples(6) = 9 pre spaces
3. Outputs star aligned frequency list of all items as histogram.
4. Asterisks should equal frequency of word from input.

Text

Description automatically generated

4. If user input is #4

* + 1. Terminates the program.

Graphical user interface, text, application

Description automatically generated

5. Otherwise, Ask for menu choice input again.

Text

Description automatically generated with medium confidence

} END While loop if input is 4.

OUTPUT Greeting message function

**END PROGRAM**

—---------------------------------------------------------------------------------------------------------------------------------------—---------------------------------------------------------------------------------------------

—---------------------------------------------------------------------------------------------------------------------------------------—---------------------------------------------------------------------------------------------

Picture Library and Documentation:

Taking pieces and piecing together in illustrator – 4/11/2023 2:45 AM

Diagram

Description automatically generated

**Program Pseudocode: Pre - Version 1**

**START PROGRAM:**

INPUT FILE: CS210\_Project\_Three\_Input\_File.txt

* Contains a list of items that have been bought throughout the day.

OUTPUT FILE: frequency.dat

* Receives output containing items and their sale frequency.

OUTPUT GREETING MESSAGE

OUTPUT Menu options:

1. Find items frequency.
2. Prints frequency of all the items
3. Prints a histogram of all the items.
4. Terminates the program.

INPUT a menu choice (1-4) from the user.

While user input doesn’t equal 4, we will continue to loop the program. {

1. If user input is #1
   * 1. Input requested item.
     2. Outputs value of items frequency in the list
2. If user input is #2
   * 1. Outputs frequency list of all items in order from stores open to close. Example:

* Potatoes 4
* Pumpkins 5
* Onions 3

3. If user input is #3

1. Outputs frequency list of all items as histogram

2. Asterisks should equal frequency of word from input. Example:

* Potatoes \*\*\*\*
* Pumpkins \*\*\*\*\*
* Onions \*\*\*

4. If user input is #4

* + 1. Terminates the program.

5. Otherwise, Ask for menu choice input again.

} //END WHILE LOOP if input is 4.

OUTPUT ENDING MESSAGE

**END PROGRAM**

—---------------------------------------------------------------------------------------------------------------------------------------—---------------------------------------------------------------------------------------------

**Description:**

Program analyzes daily text records from input file CS210\_Project\_Three\_Input\_File.txt:

* Records consist of a list of purchased items in order from the stores open to the stores close.

A data file frequency.dat is created in the beginning of the program without user intervention and is for backup purposes.

* The frequency.dat file should include every item (represented by a word) paired with the number of times that item appears in the input file.

**Code Rules:**

Variables Names, comments, version, structure

1. Insert in-line comments to denote your changes and briefly describe the functionality of the code
2. Use appropriate variable, parameter, and other naming conventions throughout your code.
3. Please use the version of Microsoft Visual Studio outlined in the syllabus.
4. Demonstrate industry standard best practices in all your code to ensure clarity, consistency, and efficiency.

Classes and Maps

1. For this assignment, your implementation plan must include at least one class with public and private sections.
2. For this assignment, we recommend using Maps for implementation. Maps do require precise coding syntax. However, you can choose a different implementation option.

**Syntax Error Rules:**

Program should produce an error message for any issues regarding syntax. Some common syntax errors are:

* missing a semicolon
* calling a function that does not exist
* not closing an open bracket
* using double quotes and not closing them in a string

**Chapter Resources:**

1. For assistance with classes, see sections 6.2, "Using a Class," and 6.3, "Defining a Class," in zyBooks.
2. Maps (section 11.4 in zyBooks)
3. For assistance with writing files, see sections 7.1, "File Input," and 7.3, "File Output," in zyBooks.

**Program Documentation:**

1. Describe your code’s design and functionality.
2. Include screenshots to support your description.
3. As you complete this work, your manager at Chada Tech is interested to see your thought process regarding how you use C++.

—---------------------------------------------------------------------------------------------------------------------------------------—---------------------------------------------------------------------------------------------