External Documentation

1. Program Overview

This program is designed to read a text file, count the occurrences of each word, and then display a list of words along with the count in a table format on the console. The program prompts the user for a file name, reads the specified file, and generates a summary of word occurrences.

2. Documentation of the job functions

a) 'main()' Function:

- Requests a file name from the user.
-Opens the specified file and reads it.
- Counts the occurrences of each word using the 'addWord' function.
- Generates a tabular list of words and their counts using the `reportToScreen` function.

b) 'addWord' Function:

- Adds a word to the vectors ('words' and 'counts') or increments the count if the word already exists.
- Use the 'findWordPosition' function to check if the word already exists in the vector.

c) Function 'findWordPosition':

- Finds the position of a word in the vector, ignoring case.

- Returns the position if the word is found; otherwise, returns -1.

d) Function 'reportToScreen':

- Displays words and their occurrences on the screen in tabular form.

- Uses columns to display words and occurrences.

e) Results display

- Displays the list of words and their counts in an easy and organized way.

f) Error handling

- Checks for errors when opening files and notifies the user if the specified file is not found.

g) Modular structure

- Uses modular functions for better code organization and readability.

3. Description of Known Problems

a) Does not handle punctuation:

- The program does not handle punctuation marks or special characters, treating them as part of words. This can result in inaccurate word counts if the file contains punctuated words.

b) Limited column width:

- The program uses a fixed column width ('COL_WIDTH = 3') to display words and occurrences. This can lead to formatting problems if words are longer than the specified width.

Conclusion

This program efficiently reads a text file, counts word occurrences, and presents the results in a tabular format. Although it works well for basic use cases, there are known issues that need to be addressed to improve its functionality and robustness, particularly with respect to punctuation and column width. In addition, improvements can be made to error handling and user input validation for a more complete solution.

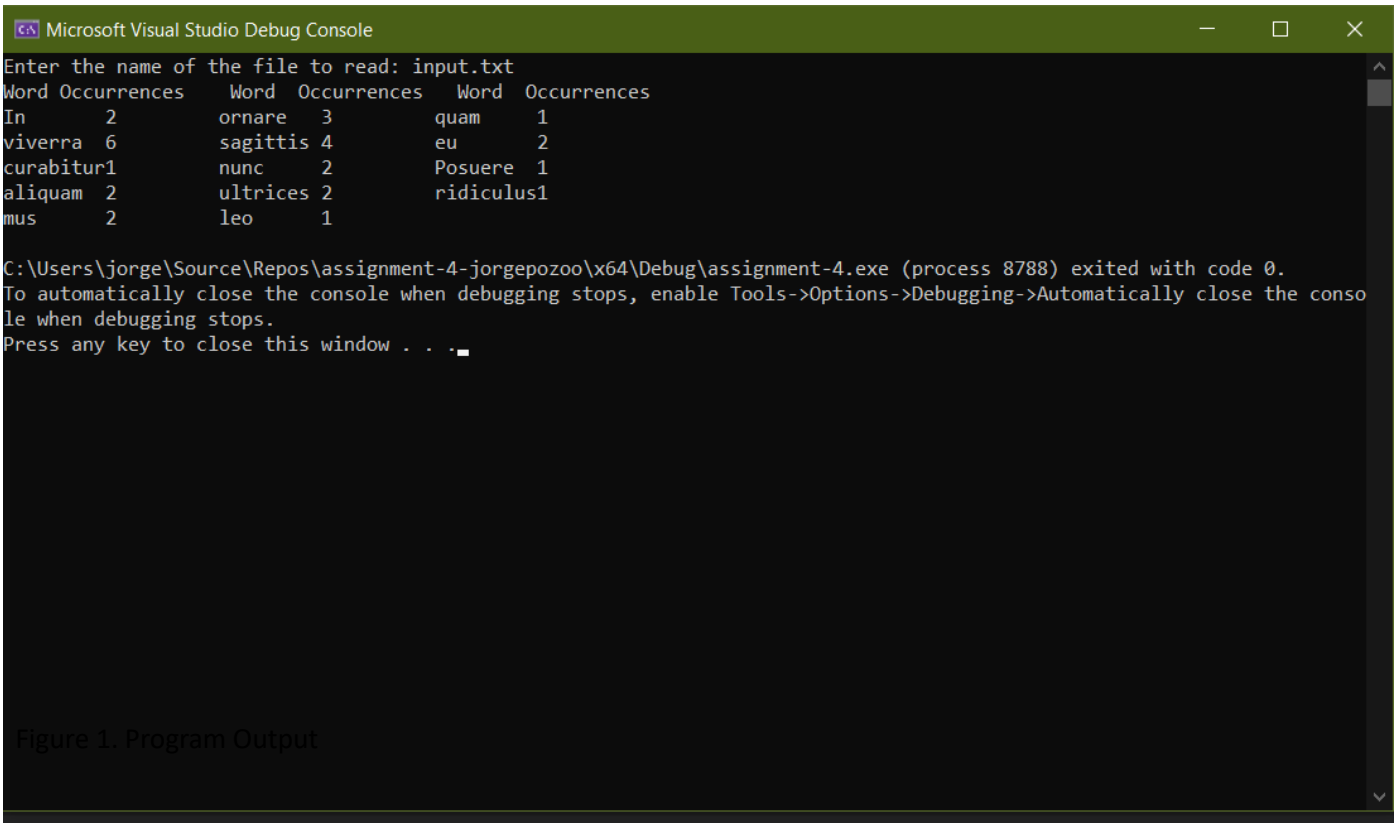

Figure 1. Program Output

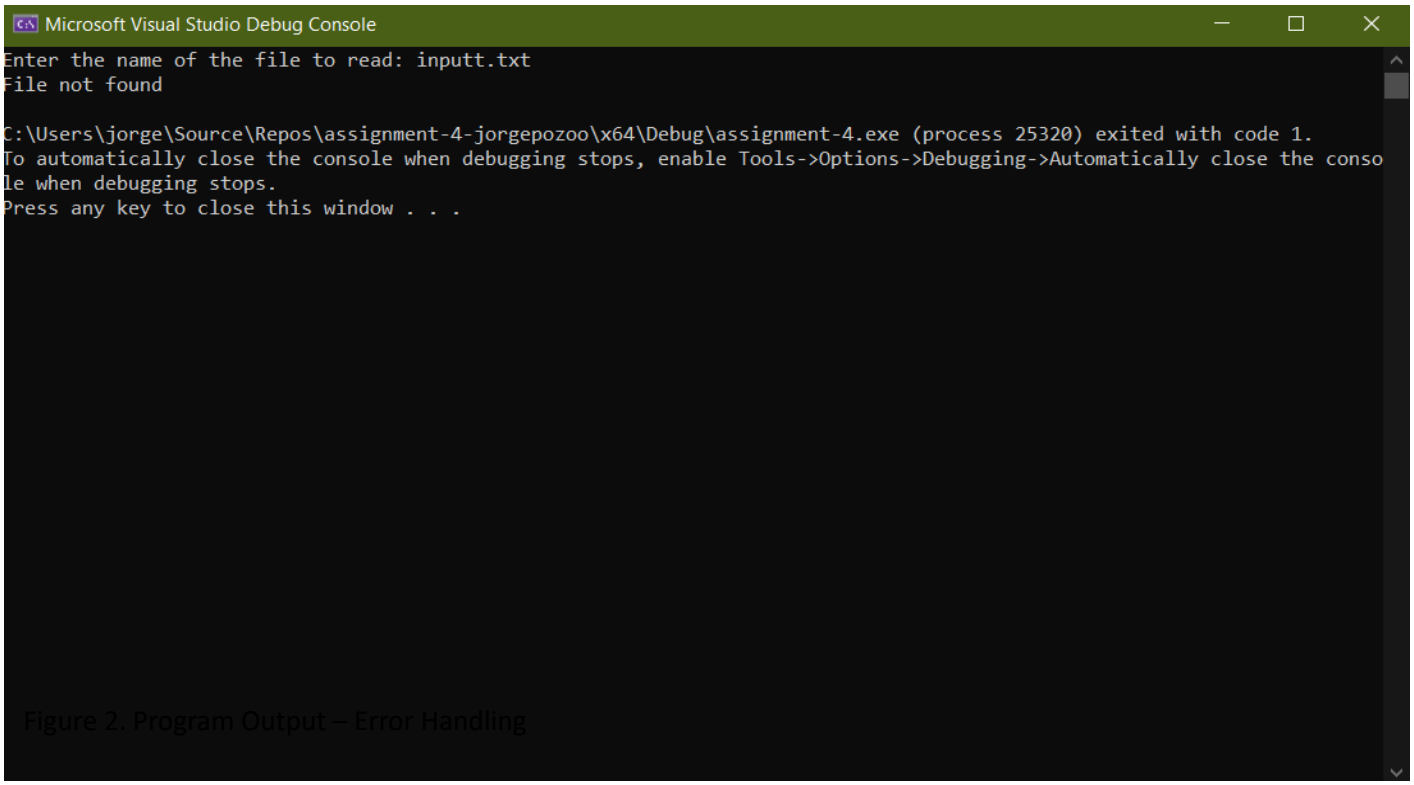

Figure 2. Program Output – Error Handling