Programming Assignment 2: Vowel Count Using String Operations

Introduction

This report outlines the approach and solution for the programming assignment focused on text processing. The primary objectives were to read text from a file, perform specific operations including counting vowels, contiguous characters, and formatting the output.

Objectives

The main goals of the assignment were to:

- Dynamically read and store text from "Gift_of_Magi.txt".

- Count the number of vowels in the text and output their frequencies.

- Identify contiguous character sequences and their counts.

- Output the first 800 characters of the text, formatted with 80 characters per line.

Approach and Design

Reading the File

A dynamic buffer was implemented to read the text file. The buffer's size starts at 512 characters and doubles whenever its capacity is exceeded during reading. This ensures efficient memory usage for any file size.

Counting Vowels

A function `count_letters` was implemented to traverse the text stored in the buffer, using a helper function `is_vowel` to check if a character is a vowel. Vowel counts were tallied in an array.

Identifying Contiguous Characters

The `count_contiguous_characters` function scans through the text, comparing each character with its predecessor to find sequences of identical characters, categorizing them into singles, doubles, triples, and sequences of four or more characters.

Formatting Output

A custom function `print_first_800_characters` was designed to output the first 800 characters of the input text, ensuring that no more than 80 characters are printed per line, to achieve a readable format.

Implementation

The program was written in C, making extensive use of the standard I/O library for file operations and dynamic memory management for efficient handling of the input text. Error handling was implemented for file operations to ensure robustness.

Challenges Encountered

One significant challenge was managing the dynamic memory allocation for the buffer, particularly in ensuring that memory was not leaked and that the buffer was resized efficiently. Additionally, formatting the output to meet the specific requirements of 80 characters per line required careful implementation to handle cases where the text did not divide evenly into lines.

Results

The program successfully reads the input text, counts vowels and contiguous characters, and formats the first 800 characters as specified. This was verified through extensive testing with various input files.

Conclusion

This assignment reinforced concepts of dynamic memory management, file I/O, and string manipulation in C. The challenges faced during implementation provided valuable lessons in problem-solving and debugging.