Development of C Program for Analyzing Text Data

D1262028 李皓鈞 Jimmy

Introduction:

The task was to develop a C program capable of analyzing text data from an input file named "Gift_of_Magi.txt". The program needed to perform several tasks, including reading the text, processing it, and generating various statistics such as the count of characters, contiguous letter sequences, and occurrences of vowels.

Development Process:

1. File Handling:

- The program starts by including necessary header files such as `<stdio.h>`, `<stdlib.h>`, `<ctype.h>`, and `<string.h>`.

- File pointers for both input and output files are declared: `fptrIn` and `fptrOut`, respectively.

2. Memory Allocation for Buffer:

- A buffer to store the input text data is dynamically allocated with an initial capacity of 512 characters.

- The `malloc()` function is used to allocate memory for the buffer.

- Error handling is included to check if memory allocation fails.

3. Reading Input Text:

- The input text file "Gift_of_Magi.txt" is opened in read mode using `fopen()`.

- Characters are read from the file one by one using `fgetc()` until the end of file (EOF) is reached.

- During reading, each character is checked using `isalpha()` to determine if it is an English letter.

- If a letter is encountered, it is converted to uppercase using `toupper()` and stored in the buffer.

- The buffer is dynamically resized if it reaches its capacity using `realloc()`.

4. Processing Text Data:

- After reading the entire text, an end-of-string character '\0' is inserted at the end of the buffer.

- The total count of English characters is calculated and displayed.

- The first 800 characters of the modified text are printed on the console, with 80 characters per line.

- Contiguous letter sequences are counted, including single characters, two, three, and four or more contiguous characters.

- The counts of contiguous letter sequences are displayed.

5. Counting Vowels:

- Vowels ('A', 'E', 'I', 'O', 'U') are counted using the `strchr()` function from `<string.h>`.

- A loop iterates through each vowel and counts its occurrences in the text.

- The counts of each vowel and the total vowel count are displayed.

6. Output File Handling:

- The modified text data is written to an output file named "result.txt" using `fwrite()`.

- Error handling is included to check if the output file fails to open.

7. Memory Deallocation:

- Memory allocated for the buffer is freed using `free()` to prevent memory leaks.

- Both input and output file pointers are closed using `fclose()`.

Conclusion:

The developed C program successfully meets the requirements outlined in the task. It effectively reads, processes, and analyzes text data from the input file, generating various statistics and outputting the results to the console and an output file. The program demonstrates the use of file handling, memory allocation, string manipulation, and character processing in C programming. Additionally, error handling ensures robustness and reliability in handling file operations and memory allocation.