COUNT

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

// Constants for command input
#define MAX_INPUT 80
#define MAX_ARGS 10

// Function to tokenize command input
int make_toks(char *input, char *args[]) {
    int i = 0;
    char *token;

    token = strtok(input, " ");
    while (token != NULL) {
        args[i++] = token;
        token = strtok(NULL, " ");
    }
    args[i] = NULL; // Null-terminate the arguments array
    return i; // Return number of tokens
}

// Function to count characters, words, and lines in a file
void count_file(char *option, char *filename) {
    FILE *file = fopen(filename, "r");
    if (!file) {
        printf("File %s not found.\n", filename);
        return;
    }

    char c;
    int charCount = 0, wordCount = 0, lineCount = 0;

    while ((c = fgetc(file)) != EOF) {
        charCount++;
        if (c == ' ' || c == '\n') wordCount++;
        if (c == '\n') lineCount++;
    }
    if (lineCount > 0) lineCount++; // Adjust for the last line if not terminated by
newline

    if (strcmp(option, "c") == 0) {
        printf("Number of characters: %d\n", charCount);
    } else if (strcmp(option, "w") == 0) {
        printf("Number of words: %d\n", wordCount);
    } else if (strcmp(option, "l") == 0) {
        printf("Number of lines: %d\n", lineCount);
    }

    fclose(file);
}

// Main shell loop
void myshell() {
    char input[MAX_INPUT];
    char *args[MAX_ARGS];

    while (1) {
        printf("myshell$ ");
        fflush(stdout);
        fgets(input, sizeof(input), stdin);

        // Remove trailing newline character
        input[strcspn(input, "\n")] = 0;

        // Tokenize the input
        int n = make_toks(input, args);

        // Handle built-in commands
        if (n > 0) {
            if (strcmp(args[0], "exit") == 0) {
                exit(0); // Exit the shell
            } else if (strcmp(args[0], "count") == 0 && n == 3) {
                count_file(args[1], args[2]); // Call count function
            } else {
                printf("Invalid command.\n");
            }
        }
    }
}

int main() {
    myshell(); // Start the shell
    return 0;
}
```