Client1.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <fcntl.h>
#include <string.h>

int main() {
    printf("\n\tClient - Listening\n");

    // Create named FIFOs, ignoring errors if they already exist
    mkfifo("fifo6.txt", 0666);
    mkfifo("fifo7.txt", 0666);

    // Open FIFOs
    int fd = open("fifo6.txt", O_RDONLY);
    int fd2 = open("fifo7.txt", O_WRONLY);

    if (fd == -1 || fd2 == -1) {
        perror("Cannot open FIFO for read/write");
        return EXIT_FAILURE;
    }

    printf("FIFO OPEN\n");

    char stringBuffer[5000] = {0}, strMessage[5000] = {0}, len;

    // Read message length and content
    read(fd, &len, 1);
    read(fd, stringBuffer, len);
    printf("\nClient Received: %s\n", stringBuffer);

    // Count words, characters, and lines
    int chars = 0, words = 0, lines = 0;
    for (int i = 0; stringBuffer[i]; i++, chars++) {
        if (stringBuffer[i] == ' ' || stringBuffer[i] == '\n') words++;
        if (stringBuffer[i] == '\n') lines++;
    }

    // Format output with word, character, and line counts
    snprintf(strMessage, sizeof(strMessage),
            "No.of Words: %d::: No.of Characters: %d::: No.of Lines: %d",
            words, chars, lines);

    // Write the length of response and message to the server
    len = strlen(strMessage);
    write(fd2, &len, 1);
    write(fd2, strMessage, len);
```

```
    close(fd);
    close(fd2);
    printf("\nCLIENT CLOSED\nSERVER CLOSED\n");
    return 0;
}
```

server1.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <fcntl.h>
#include <string.h>

int main() {
    printf("Server\n");

    char strMessage[5000], stringBuffer[5000] = {0};
    int fd = open("fifo6.txt", O_WRONLY);
    int fd2 = open("fifo7.txt", O_RDONLY);

    if (fd == -1 || fd2 == -1) {
        perror("Cannot open FIFOs");
        return EXIT_FAILURE;
    }

    printf("FIFO OPEN\n");

    while (1) {
        printf("\nEnter the Message (press ENTER to quit): ");
        fgets(strMessage, sizeof(strMessage), stdin);

        char len = (char)strlen(strMessage);
        if (len == 1) break;  // Exit if input is empty

        write(fd, &len, 1);
        write(fd, strMessage, len);

        int len2;
        read(fd2, &len2, 1);
        read(fd2, stringBuffer, len2);
        stringBuffer[len2] = '\0';

        printf("Server Received: %s\n", stringBuffer);
    }

    close(fd);
    close(fd2);
    printf("\nSERVER CLOSED\n");
    return 0;
}
```