```
// Jessica Elkins
// UAB CS332
// 2/11/20
// This program reads a variable number of keywords as command line arguments, reads te
xt from the
// standard input stream, searches the keywords in the input stream, and when the end o
f input
// stream is reached, prints the number of times each keyword appears in the input text
.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

//creating structure
typedef struct tables {

        char keyword[BUFSIZ];
        int count;

} tables;

//globally defining structure so it can be reached from different functions
struct tables *table;

//this function creates an array of structures based on command line arg keywords
//and initializes count as zero
void init_table(int argc, char** argv);

//this function searches for keywords in the input and updates the table if match is fo
und
void update_table(char* line, int argc);

//this function prints the keywords and counters to the display
void display_table();

int main(int argc, char** argv) {

        // calling init_table to create array of structures with
        // the given command line arguments
        init_table(argc, argv);

        // address of buffer
        char *line = NULL;
        // size of buffer
        size_t maxlen = 0;
        ssize_t n;

        while((n = getline(&line,  &maxlen, stdin)) > 0) {

                // calling update_table function to search for keywords
                // and update counters
                update_table(line, argc);
        }

        // calling display_table function to print
        // keywords and counters to the display
        display_table(argc);

        // freeing malloc
        free(line);
        free(table);
```

```
        return 0;

}

void init_table(int argc, char** argv) {

        int i;
        // to account for executable file
        int size = argc - 1;

        // dynamically allocation size of structure
        table = (tables *)malloc(size * sizeof(tables));

        for(i = 0; i < size; i++){
                // copying the command line arguments into the structure
                strcpy(table[i].keyword, argv[i + 1]);
                // initializing keyword count to 0
                table[i].count = 0;
        }
}

void update_table(char* line, int argc) {
        int i, size = argc - 1;

        // using strtok to split stdin lines by spaces and newline included by getline
        char *token = strtok(line, "  \n");

        while(token != NULL) {
                for(i = 0; i < size; i++) {
                        // comparing tokenized line with each keyword and
                        // updating counter if match found(string compare returned 0)
                        if(strcmp(token,table[i].keyword) == 0){
                                table[i].count += 1;
                        }
                }
                // ending while loop
                token = strtok(NULL, " \n");
        }
}

void display_table(int argc) {

        int i, size = argc - 1;
        printf("\n");
        printf("Here is the number of times each keyword appears: \n");

        for(i = 0; i < size; i++){
                printf("%s: %d \n", table[i].keyword, table[i].count);
        }

}
```