```c
/* About this program:
- This program counts words.
- The specific words that will be counted are passed in as command-line
  arguments.
- The program reads words (one word per line) from standard input until
  EOF. (Note that EOF can be typed as <CTRL-D>).
- The program prints out a summary of the number of times each word has
  appeared.
- Various command-line options alter the behavior of the program.

E.g., count the number of times 'cat', 'nap' or 'dog' appears.
> ./main cat nap dog
Given input:
 cat
 .
Expected output:
 Looking for 3 words
 Result:
 cat:1
 nap:0
 dog:0

Note: this code was automatically formatted (styled) using 'indent main.c -kr'.

This assignment was adapted from operating system programming problems by
Lawrence Angrave at the University of Illinois at Champaign-Urbana (UIUC).
*/

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

#define LENGTH(s) (sizeof(s) / sizeof(*s))

/* Structures */
typedef struct {
    char *word;
    int counter;
} word_count_entry_t;


int process_stream(word_count_entry_t entries[], int entry_count)
{
    short line_count = 0;
    char buffer[30];

    while (gets(buffer)) {
        if (*buffer == '.')
            break;
        /* Compare against each entry */
        int i = 0;
        while (i < entry_count) {

            if (!strcmp(entries[i].word, buffer))
                entries[i].counter++;
            i++;
        }
        line_count++;
    }
    return line_count;
}

void print_result(word_count_entry_t entries[], int entry_count)
{
    printf("Result:\n");
    while (entry_count-- > 0) {
        printf("%s:%d\n", entries->word, entries->counter);
    }

}


void print_help(const char *name)
{
    printf("usage: %s [-h] <word1> ... <wordN>\n", name);
}


int main(int argc, char **argv)
{
    const char *prog_name = *argv;

    word_count_entry_t entries[5];
    int entry_count = 0;

    /* Entry point for the testrunner program */
    if (argc > 1 && !strcmp(argv[1], "-test")) {
        run_c_review_tests(argc - 1, argv + 1);
        return EXIT_SUCCESS;
    }

    while (*argv != NULL) {
        if (**argv == '-') {

            switch ((*argv)[1]) {

            case 'h':
                print_help(prog_name);
            default:
                printf("%s: Invalid option %s. Use -h for help.\n",
                       prog_name, *argv);
            }
        } else {
            if (entry_count < LENGTH(entries)) {
                entries[entry_count].word = *argv;
                entries[entry_count++].counter = 0;
            }
        }
        argv++;
    }
    if (entry_count == 0) {
        printf("%s: Please supply at least one word. Use -h for help.\n",
               prog_name);
        return EXIT_FAILURE;
    }
    if (entry_count == 1) {
        printf("Looking for a single word\n");
    } else {
        printf("Looking for %d words\n", entry_count);
    }

    process_stream(entries, entry_count);
    print_result(entries, entry_count);

    return EXIT_SUCCESS;
}
```