```c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
#define max_string_size 100
#define max_pattern_size 100

int pmatch(char* string, char* pat);
void fail(char* pat);

int failure[max_pattern_size];
char string[max_string_size];
char pat[max_pattern_size];

int main() {
    FILE* fp = (FILE*)fopen("kmp.txt", "r");
    fscanf(fp, "%s", string);
    fscanf(fp, "%s", pat);
    fail(pat);
    printf("%d", pmatch(string, pat));
    return 0;
}

int pmatch(char* string, char* pat) {
    /*Knuth, Morris, Pratt string matching algorithm*/
    int i = 0, j = 0;
    int lens = (int)strlen(string);
    int lenp = (int)strlen(pat);

    while (i < lens && j < lenp) {
        if (string[i] == pat[j]) {
            i++; j++;
        }
        else if (j == 0) i++;
        else j = failure[j - 1] + 1;
    }
    return ((j == lenp) ? (i - lenp) : -1);
}

void fail(char* pat) {
    /*compute the pattern's failure function*/
    int i = 0, n = (int)strlen(pat);
    failure[0] = -1;
    for (int j = 1; j < n; j++) {
        i = failure[j - 1];
        while ((pat[j] != pat[i + 1]) && (i >= 0))
            i = failure[i];
        if (pat[j] == pat[i + 1])
            failure[j] = i + 1;
        else failure[j] = -1;
    }
}
```