

**实验十一**

**数组**

学期：2024-2025 第一学期

编制日期：2024 年 11月 28 日

编制人：江家玮

学号：22281188

班级：计科2204

**实验十一**

**知识覆盖：字符数组 & 结构体**

1. **写一个函数，统计某字母（不区分大小写）在字符串中出现的次数。**

函数原型为：int count(const char str[], char c)，其中c为待统计的字母。要求待统计字母和字符串在主程序中输入，然后在主程序中调用该函数，并将结果在主程序中输出。如：j在Beijing Jiaotong中出现的次数为2，i在Beijing Jiaotong中出现的次数为3，u在Beijing Jiaotong中出现的次数为0。

#include <stdio.h>

#include <ctype.h> // 用于tolower函数

// 统计某字母在字符串中出现的次数（不区分大小写）

int count(const char str[], char c) {

int count = 0;

char lower\_c = tolower(c); // 将待统计字母转换为小写

// 遍历字符串

for (int i = 0; str[i] != '\0'; i++) {

// 将字符串中的字符转换为小写并比较

if (tolower(str[i]) == lower\_c) {

count++;

}

}

return count;

}

int main() {

char str[100]; // 定义字符串数组

char c; // 定义待统计的字母

// 输入字符串

printf("请输入一个字符串: ");

fgets(str, sizeof(str), stdin); // 使用fgets读取字符串，避免缓冲区溢出

// 输入待统计的字母

printf("请输入待统计的字母: ");

scanf\_s(" %c", &c); // 注意前面的空格，用于跳过换行符

// 调用count函数统计字母出现的次数

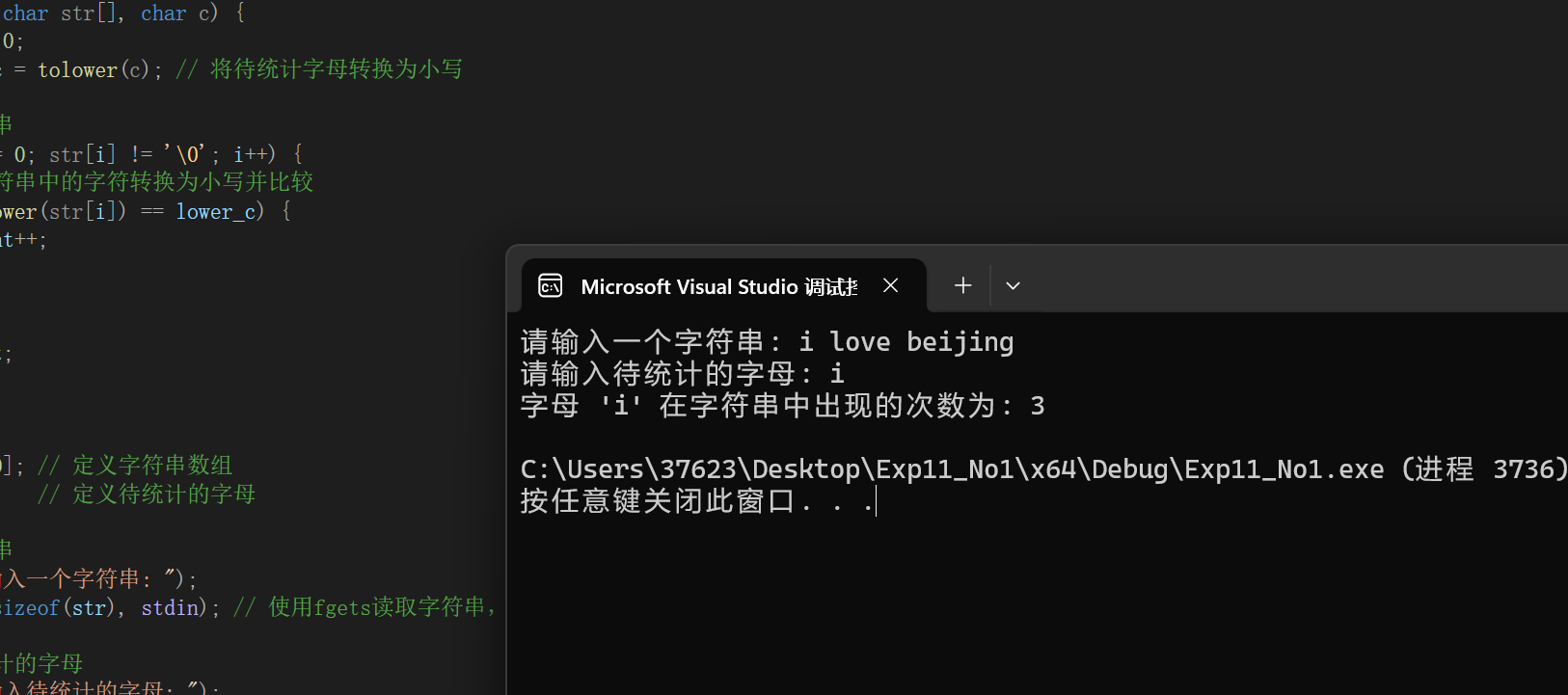
int result = count(str, c);

// 输出结果

printf("字母 '%c' 在字符串中出现的次数为: %d\n", c, result);

return 0;

}



1. **写一个函数squeeze(char s1[], char s2[])，它从字符串s1中删除所有出现在串s2里的字符，并保证不改变原字符串s1的先后顺序。**

#include <stdio.h>

#include <string.h>

// 从字符串s1中删除所有出现在字符串s2中的字符

void squeeze(char s1[], const char s2[]) {

int i, j, k;

int found; // 标记是否找到匹配字符

// 遍历字符串s1

for (i = j = 0; s1[i] != '\0'; i++) {

found = 0; // 初始化标记为未找到

// 检查当前字符s1[i]是否在s2中

for (k = 0; s2[k] != '\0'; k++) {

if (s1[i] == s2[k]) {

found = 1; // 找到匹配字符

break;

}

}

// 如果未找到匹配字符，保留该字符

if (!found) {

s1[j++] = s1[i];

}

}

// 添加字符串结束符

s1[j] = '\0';

}

int main() {

char s1[100]; // 定义字符串s1

char s2[100]; // 定义字符串s2

// 输入字符串s1

printf("请输入字符串s1: ");

fgets(s1, sizeof(s1), stdin);

s1[strcspn(s1, "\n")] = '\0'; // 去除fgets读取的换行符

// 输入字符串s2

printf("请输入字符串s2: ");

fgets(s2, sizeof(s2), stdin);

s2[strcspn(s2, "\n")] = '\0'; // 去除fgets读取的换行符

// 调用squeeze函数

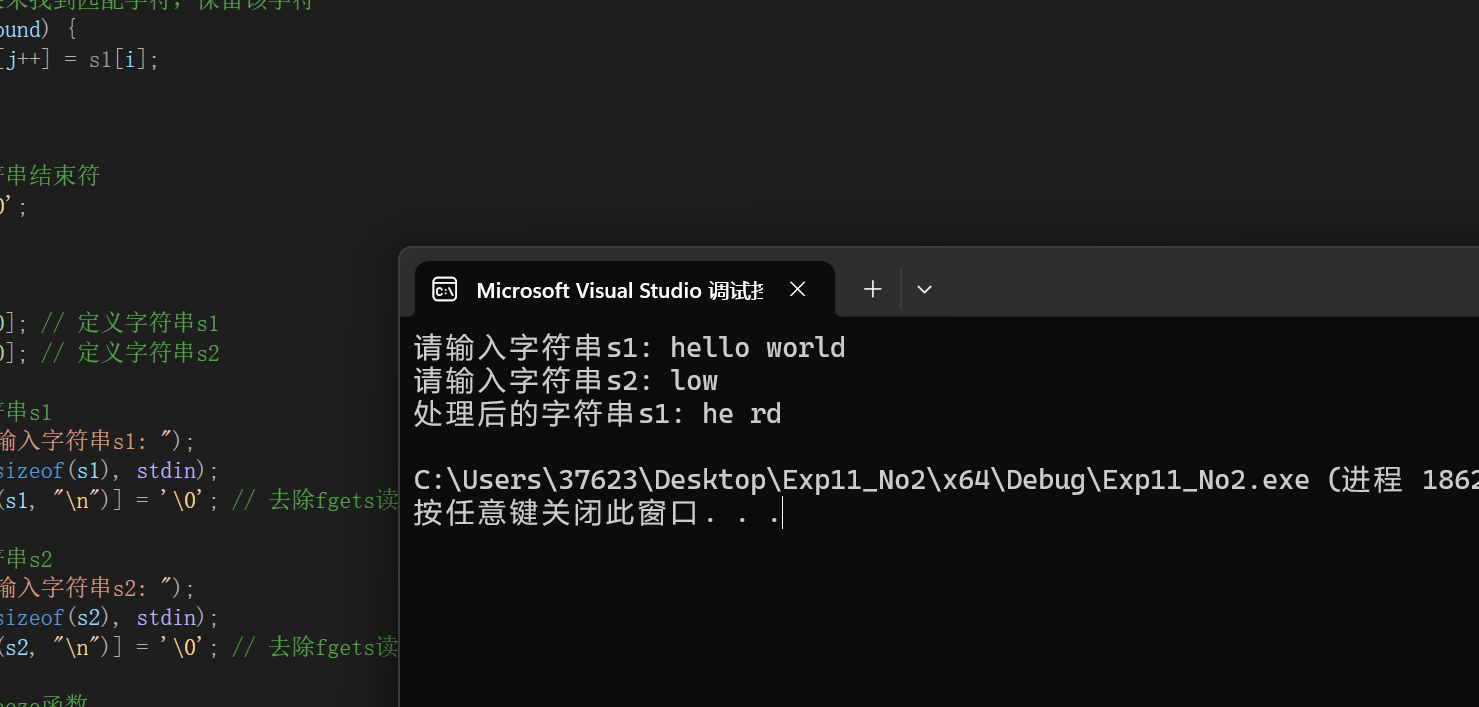
squeeze(s1, s2);

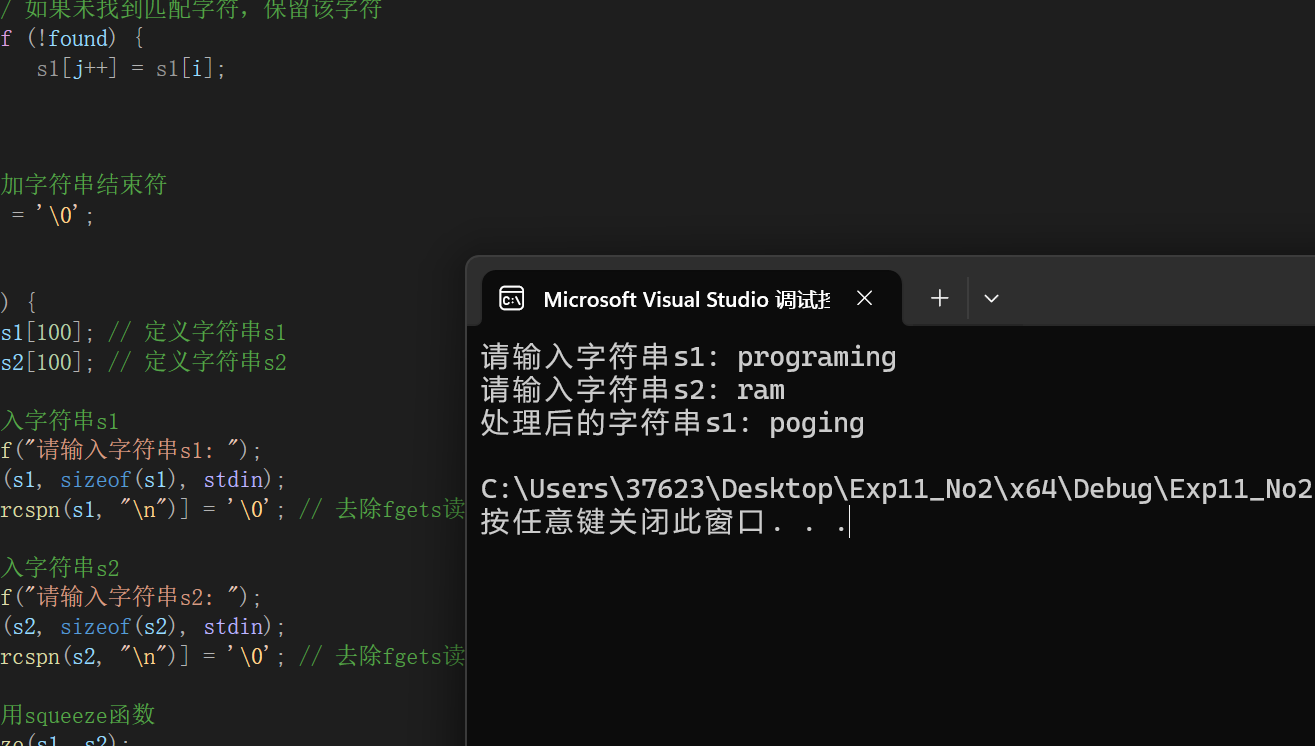
// 输出结果

printf("处理后的字符串s1: %s\n", s1);

return 0;

}





1. **回文是从前向后和从后向前读起来都一样的句子，如英文的level，amannama，请写一个函数，它能够判断一个字符串是否为回文。另外，考虑如果该字符串为中文字符时（如：落叶秋深秋叶落）如何处理。**

提示：一个中文字符由占两个字节，在字符数组中由连续的两个元素编码。

#include <stdio.h>

#include <string.h>

#include <stdbool.h>

// 判断是否为中文字符（UTF-8编码）

bool isChineseChar(const char\* str, int index) {

// UTF-8中文字符的前两个字节满足特定条件

return (str[index] & 0x80) && (str[index + 1] & 0x80);

}

// 判断字符串是否为回文

bool isPalindrome(const char\* str) {

int len = strlen(str);

int i = 0, j = len - 1;

while (i < j) {

// 如果是中文字符，跳过两个字节

if (isChineseChar(str, i)) {

if (str[i] != str[j - 1] || str[i + 1] != str[j]) {

return false;

}

i += 2;

j -= 2;

}

else {

// 如果是单字节字符，直接比较

if (str[i] != str[j]) {

return false;

}

i++;

j--;

}

}

return true;

}

int main() {

char str[100];

// 输入字符串

printf("请输入一个字符串: ");

fgets(str, sizeof(str), stdin);

str[strcspn(str, "\n")] = '\0'; // 去除fgets读取的换行符

// 判断是否为回文

if (isPalindrome(str)) {

printf("字符串 \"%s\" 是回文。\n", str);

}

else {

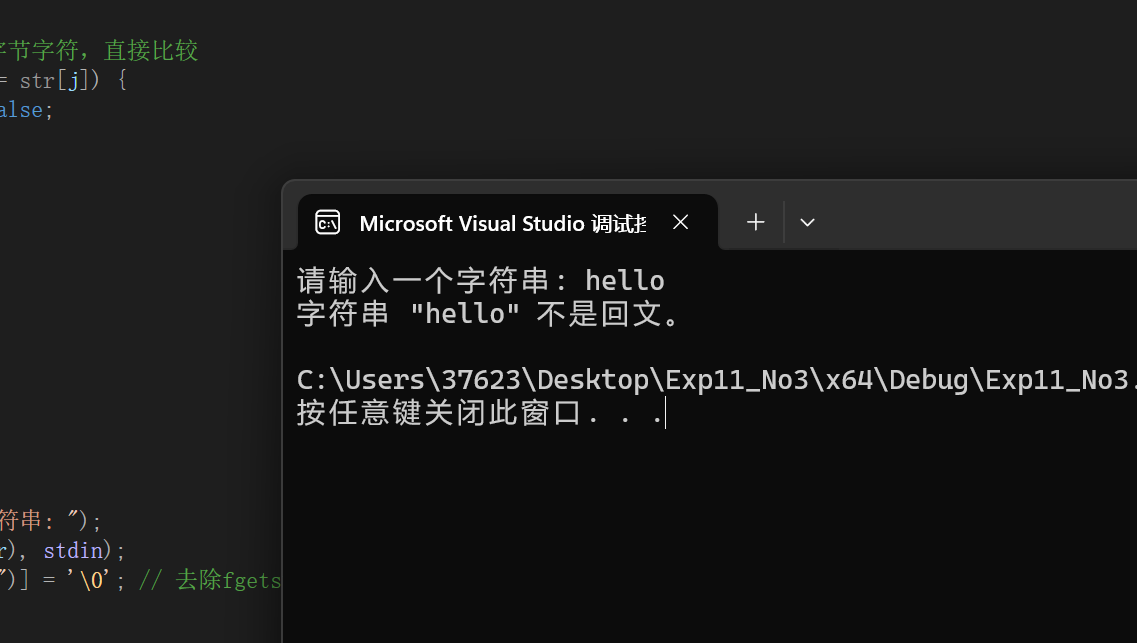
printf("字符串 \"%s\" 不是回文。\n", str);

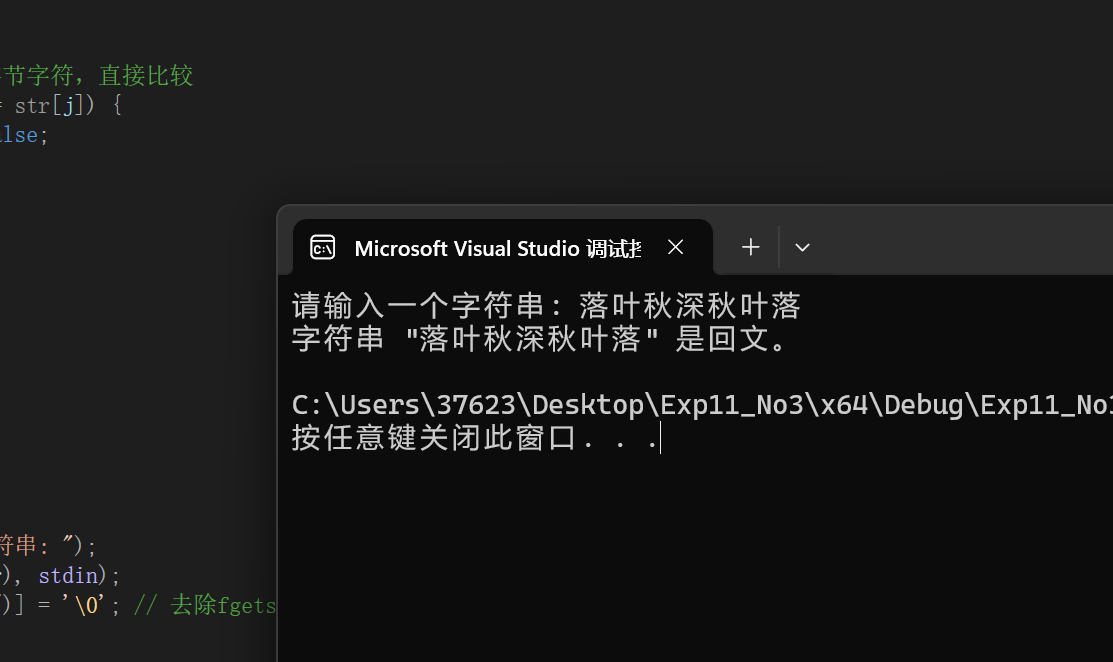
}

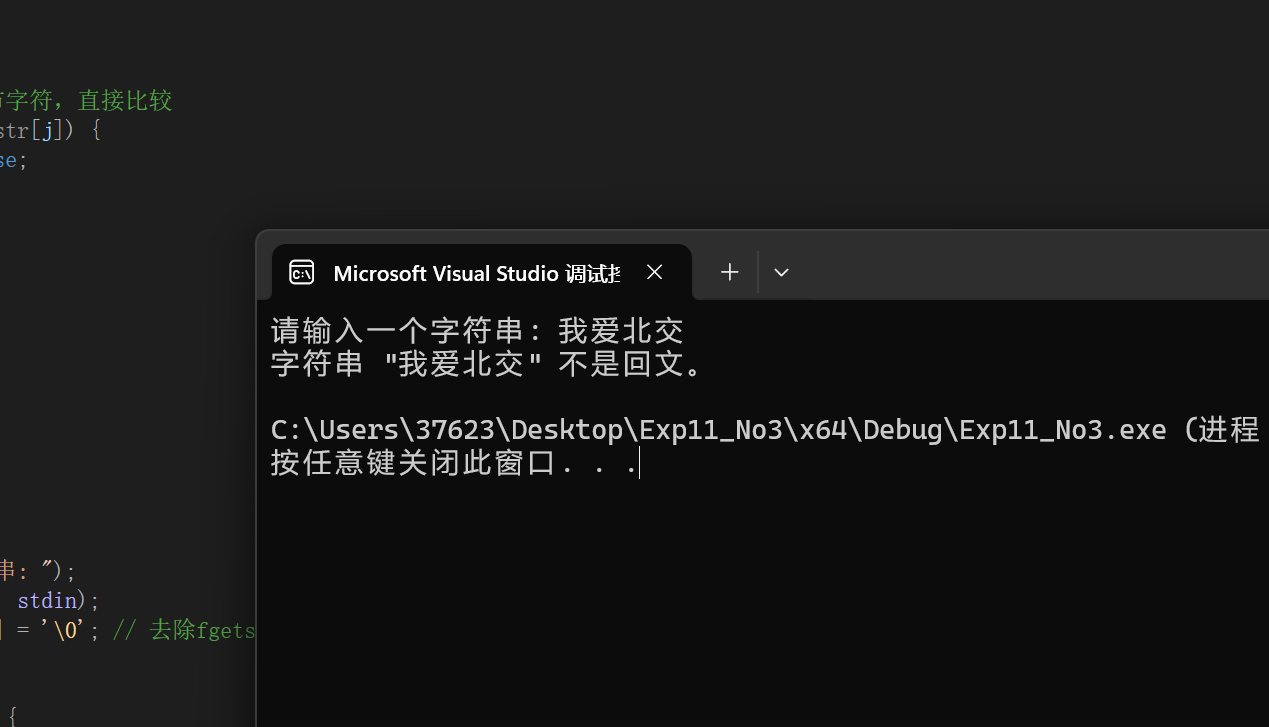
return 0;

}









1. **写一个程序，它从键盘读入多行字符（用scanf），直到遇到ctrl+z，读入结束，输出读入的多行字符中最长的词（词以空格、标点符号分割）。说明：也可以考虑用gets，判断返回值是不是NULL。**

#define \_CRT\_SECURE\_NO\_WARNINGS 1

#include <stdio.h>

#include <string.h>

#include <ctype.h>

// 获取字符串中最长的单词

void getLongestWord(const char\* str, char\* longestWord) {

int maxLength = 0;

int currentLength = 0;

int start = 0;

int i;

for (i = 0; str[i] != '\0'; i++) {

// 如果是字母，增加当前单词长度

if (isalpha(str[i])) {

currentLength++;

}

else {

// 如果不是字母，比较当前单词长度和最大长度

if (currentLength > maxLength) {

maxLength = currentLength;

strncpy(longestWord, &str[start], maxLength);

longestWord[maxLength] = '\0'; // 添加字符串结束符

}

// 重置当前单词长度和起始位置

currentLength = 0;

start = i + 1;

}

}

// 检查最后一个单词

if (currentLength > maxLength) {

maxLength = currentLength;

strncpy(longestWord, &str[start], maxLength);

longestWord[maxLength] = '\0'; // 添加字符串结束符

}

}

int main() {

char line[1000]; // 存储每行输入

char longestWord[1000] = ""; // 存储最长的单词

char currentWord[1000] = ""; // 存储当前行的最长单词

int maxLength = 0;

printf("请输入多行字符（按Ctrl+Z结束输入）:\n");

// 读取多行输入

while (fgets(line, sizeof(line), stdin) != NULL) {

// 去除换行符

line[strcspn(line, "\n")] = '\0';

// 获取当前行的最长单词

getLongestWord(line, currentWord);

// 比较当前行的最长单词和全局最长单词

if (strlen(currentWord) > maxLength) {

maxLength = strlen(currentWord);

strcpy(longestWord, currentWord);

}

}

// 输出结果

if (maxLength > 0) {

printf("最长的单词是: %s\n", longestWord);

}

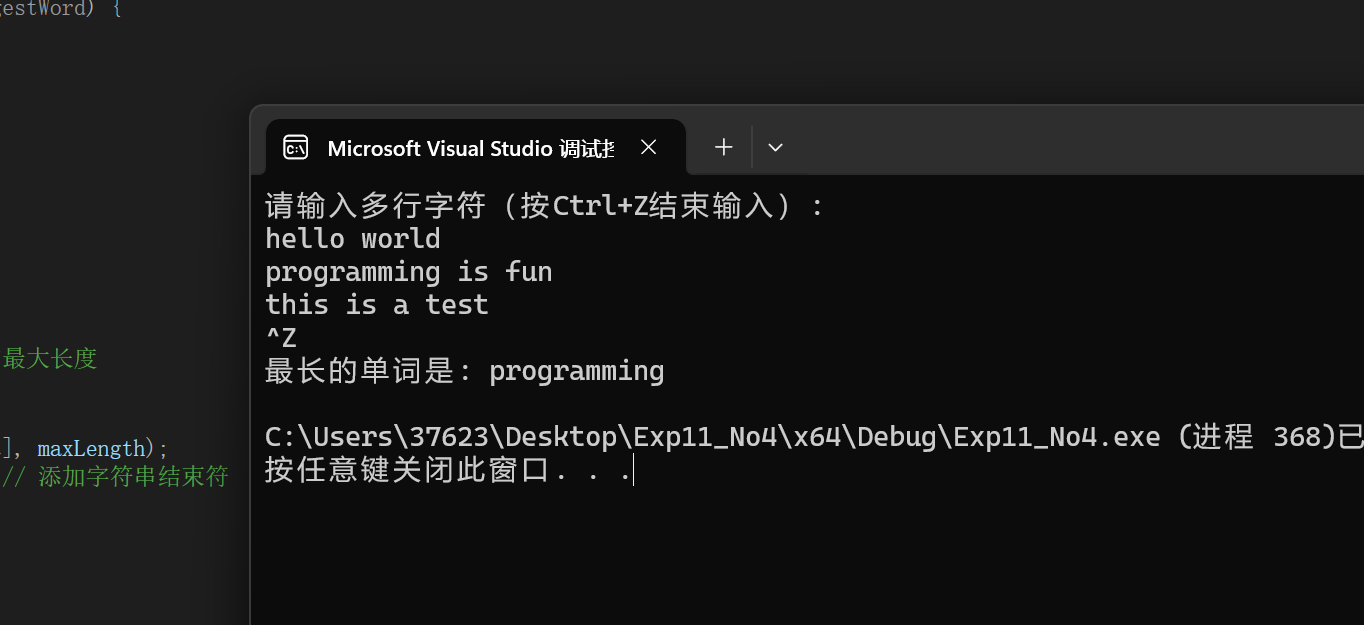
else {

printf("没有输入有效的单词。\n");

}

return 0;

}



**5. 设有如下学生结构：**

typedef struct{

char szName[20];//姓名

double dScore[2]; //两门课的成绩

}STUDENT;

设班里有一批学生成绩如下(以下是测试样例，可以有多名学生)：

3

LiMing 97 51

WangTao 71 85

ZhaoLi 45 71

输入包含两行：第一行是一个整数nLen（0 < nLen <= 10）, nLen个学生。第二行至nLen +1行是nLen个学生的记录: 姓名 分数，每个学生占一行，姓名与分数之间用一个空格隔开。

请编程求该班至少有一门课不及格的人数。

#define \_CRT\_SECURE\_NO\_WARNINGS 1

#include <stdio.h>

#include <string.h>

// 定义学生结构

typedef struct {

char szName[20]; // 姓名

double dScore[2]; // 两门课的成绩

} STUDENT;

// 判断学生是否至少有一门课不及格

int isFail(const STUDENT\* student) {

return (student->dScore[0] < 60 || student->dScore[1] < 60);

}

int main() {

int nLen; // 学生人数

STUDENT students[10]; // 学生数组，最多10个学生

int failCount = 0; // 不及格人数

char line[100]; // 用于存储每行输入

// 输入学生人数

printf("请输入学生人数: ");

scanf("%d", &nLen);

getchar(); // 读取换行符，避免影响后续输入

//printf("%d\n", nLen);

// 输入每个学生的信息

for (int i = 0; i < nLen; i++) {

printf("请输入第 %d 个学生的姓名和两门课的成绩: ", i + 1);

fgets(line, sizeof(line), stdin); // 读取一行输入

sscanf(line, "%s %lf %lf", students[i].szName, &students[i].dScore[0], &students[i].dScore[1]);

//getchar();

}

// 统计至少有一门课不及格的学生人数

for (int i = 0; i < nLen; i++) {

if (isFail(&students[i])) {

failCount++;

}

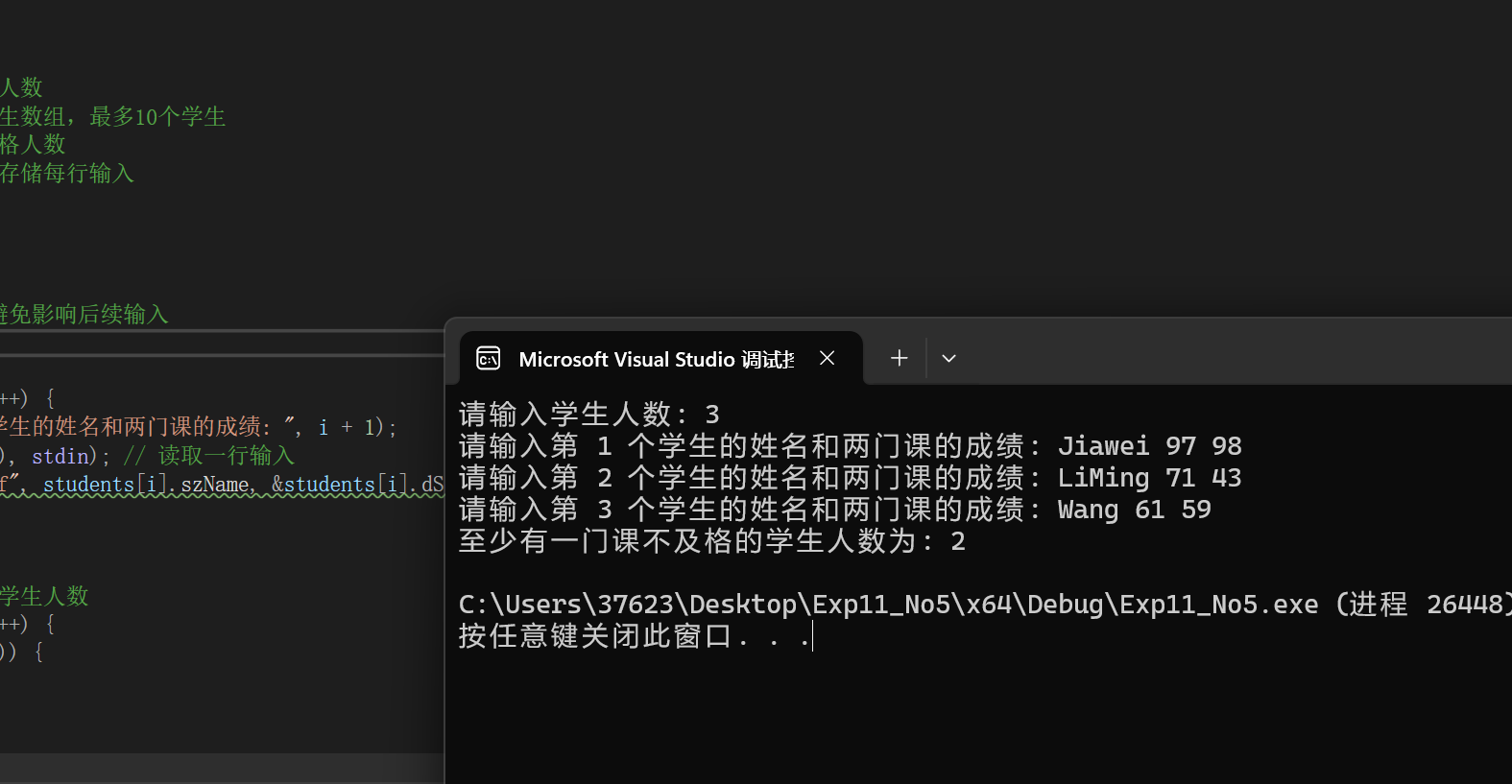
}

// 输出结果

printf("至少有一门课不及格的学生人数为: %d\n", failCount);

return 0;

}



**附录1：Scanf的返回值**

[scanf（计算机语言函数）\_百度百科 (baidu.com)](https://baike.baidu.com/item/scanf/10773316?fr=aladdin)



**Scanf返回EOF示例**

