МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Факультет автоматики и вычислительной техники

Кафедра автоматики



**ЛАБОРАТОРНАЯ РАБОТА №2**

по дисциплине: «Информатика»

«УКАЗАТЕЛИ, ФУНКЦИИ»

Вариант №13

Выполнили:Проверил:

Студенты гр. АВТ-419 Ядрышников О.Д.

Енина Ксения Сергеевна

Мерзляков Кирилл Андреевич

«16» октября 2024 г.«\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка, подпись)

Новосибирск

2024

**Цель работы:** Освоить правила написания и использования функций в языке СИ. Научиться использовать указатели при обработке массивов данных.

**Исходные данные и задание для варианта:**

4.1. Написать программу сортировки массива строк по вариантам (табл. 2.2.). Ввод данных, сортировку и вывод результатов оформить в виде функций. Входные и выходные параметры функции сортировки указаны в таблице. Входные и выходные параметры функций для ввода-вывода приведены в таблице 2.1.

4.2. Модифицировать программу п.1., применив в функциях передачу параметров и возврат результатов по ссылке (с использованием указателей). Сравнить результаты.

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | Расположить строки по возрастанию количества пробелов в строке | 1. Массив  2. Размерность массива | 1. Количество перестановок  2. Максимальное количество пробелов в строке |

**Текст программы 4.1:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define STRINGS\_COUNT 5

#define STRING\_MAX\_LEN 64

/\*

Лабораторная работа №2 - Вариант 13

Работу выполнили: Мерзляков К., Енина К.

\*/

// Объявляем прототипы функций

int inp\_str(char \*string, int maxlen);

void inp\_strings(char \*\*strings);

void out\_str(char \*string, int length, int number);

void out\_strings(char \*\*strings);

int get\_spaces\_count(char \*string);

void sort\_strings\_by\_spaces\_count(char \*\*strings, int strings\_count);

void free\_memory(char \*\*strings);

int main() {

// Инициализируем массив указателей на строки

// и запрашиваем их ввод у пользователя

char \*strings[STRINGS\_COUNT];

inp\_strings(strings);

// Выводим введенные пользователем строки

printf("\nДо сортировки:\n");

out\_strings(strings);

// Сортируем их согласно таблице 2.2 вариант 13

sort\_strings\_by\_spaces\_count(strings, STRINGS\_COUNT);

// Выводим отсортированные строки

printf("\nПосле сортировки:\n");

out\_strings(strings);

// Освобождаем выделенную для строк память

free\_memory(strings);

return 0;

}

// Ввод строки

int inp\_str(char \*string, int maxlen) {

/\*

char \*string: указатель, куда считать строку

int maxlen: максимальная длина вводимой строки

\*/

printf("\nВведите строку: ");

// Считываем строку

fgets(string, maxlen, stdin);

// Ищем индекс перевода строки (\n) в str

// на его место вставляем символ конца строки (\0)

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

return strlen(string);

}

// Ввод STRINGS\_COUNT строк в виде указателей на строки

void inp\_strings(char \*\*strings) {

/\*

char \*\*strings: массив строк

\*/

for (int i = 0; i < STRINGS\_COUNT; i++) {

// Выделяем память для строки

strings[i] = (char \*)malloc(STRING\_MAX\_LEN \* sizeof(char));

if (strings[i] == NULL) {

printf("Ошибка выделения памяти\n");

exit(1);

}

// Считываем ее и выводим в консоль ее длину

int length = inp\_str(strings[i], STRING\_MAX\_LEN);

printf("Длина введенной строки: %d\n", length);

}

}

// Вывод в консоль строки, ее длины и порядкового номера

void out\_str(char \*string, int length, int number) {

/\*

char \*string: указатель на строку

int length: длина строки

int number: порядковый номер строки

\*/

printf("%d) [длина строки = %d] \"%s\"\n", number, length, string);

}

// Вывод в консоль всех строк

void out\_strings(char \*\*strings) {

/\*

char \*\*strings: массив строк

\*/

for (int i = 0; i < STRINGS\_COUNT; i++) {

out\_str(strings[i], strlen(strings[i]), i + 1);

}

}

// Получение количества пробелов в строке

int get\_spaces\_count(char \*string) {

/\*

char \*string: указатель на строку

\*/

int count = 0;

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

if (string[i] == ' ') {

count++;

}

}

return count;

}

// Сортировка строк по возрастанию количества пробелов

// Вывод кол-ва перестановок и наибольшего числа пробелов

void sort\_strings\_by\_spaces\_count(char \*\*strings, int strings\_count) {

/\*

char \*\*strings: массив строк

int strings\_count: его размерность

\*/

int swaps\_count = 0;

int max\_spaces\_count = 0;

for (int i = 0; i < strings\_count - 1; i++) {

for (int j = i + 1; j < strings\_count; j++) {

int str\_i\_spaces\_count = get\_spaces\_count(strings[i]);

int str\_j\_spaces\_count = get\_spaces\_count(strings[j]);

if (str\_i\_spaces\_count > max\_spaces\_count) {

max\_spaces\_count = str\_i\_spaces\_count;

}

if (str\_j\_spaces\_count > max\_spaces\_count) {

max\_spaces\_count = str\_i\_spaces\_count;

}

if (str\_i\_spaces\_count > str\_j\_spaces\_count) {

char tmp[STRING\_MAX\_LEN];

strcpy(tmp, strings[i]);

strcpy(strings[i], strings[j]);

strcpy(strings[j], tmp);

swaps\_count++;

}

}

}

printf("При сортировке было совершено перестановок: %d\n", swaps\_count);

printf("Наибольшее количество пробелов среди строк: %d\n", max\_spaces\_count);

}

// Освобождаем выделенную для строк память

void free\_memory(char \*\*strings) {

/\*

char \*\*strings: массив строк

\*/

for (int i = 0; i < STRINGS\_COUNT; i++) {

free(strings[i]);

}

}

**Текст программы 4.2:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define STRINGS\_COUNT 5

#define STRING\_MAX\_LEN 64

/\*

Лабораторная работа №2 - Вариант 13

Работу выполнили: Мерзляков К., Енина К.

\*/

// Объявляем прототипы функций

void inp\_str(char \*string, int maxlen, int \*string\_len);

void inp\_strings(char \*\*strings);

void out\_str(char \*string, int \*length, int \*number);

void out\_strings(char \*\*strings);

void get\_spaces\_count(char \*string, int \*spaces\_count);

void sort\_strings\_by\_spaces\_count(char \*\*strings, int strings\_count);

void free\_memory(char \*\*strings);

int main() {

// Инициализируем массив указателей на строки

// и запрашиваем их ввод у пользователя

char \*strings[STRINGS\_COUNT];

inp\_strings(strings);

// Выводим введенные пользователем строки

printf("\nДо сортировки:\n");

out\_strings(strings);

// Сортируем их согласно таблице 2.2 вариант 13

sort\_strings\_by\_spaces\_count(strings, STRINGS\_COUNT);

// Выводим отсортированные строки

printf("\nПосле сортировки:\n");

out\_strings(strings);

// Освобождаем выделенную для строк память

free\_memory(strings);

return 0;

}

// Ввод строки

void inp\_str(char \*string, int maxlen, int \*string\_len) {

/\*

char \*string: указатель, куда считать строку

int maxlen: максимальная длина вводимой строки

\*/

printf("\nВведите строку: ");

// Считываем строку

fgets(string, maxlen, stdin);

// Ищем индекс перевода строки (\n) в str

// на его место вставляем символ конца строки (\0)

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

\*string\_len = strlen(string);

}

// Ввод STRINGS\_COUNT строк в виде указателей на строки

void inp\_strings(char \*\*strings) {

/\*

char \*\*strings: массив строк

\*/

for (int i = 0; i < STRINGS\_COUNT; i++) {

// Выделяем память для строки

strings[i] = (char \*)malloc(STRING\_MAX\_LEN \* sizeof(char));

if (strings[i] == NULL) {

printf("Ошибка выделения памяти\n");

exit(1);

}

// Считываем ее и выводим в консоль ее длину

int length;

inp\_str(strings[i], STRING\_MAX\_LEN, &length);

printf("Длина введенной строки: %d\n", length);

}

}

// Вывод в консоль строки, ее длины и порядкового номера

void out\_str(char \*string, int \*length, int \*number) {

/\*

char \*string: указатель на строку

int length: длина строки

int number: порядковый номер строки

\*/

printf("%d) [длина строки = %d] \"%s\"\n", \*number, \*length, string);

}

// Вывод в консоль всех строк

void out\_strings(char \*\*strings) {

/\*

char \*\*strings: массив строк

\*/

for (int i = 0; i < STRINGS\_COUNT; i++) {

int length = strlen(strings[i]);

int number = i + 1;

out\_str(strings[i], &length, &number);

}

}

// Получение количества пробелов в строке

void get\_spaces\_count(char \*string, int \*spaces\_count) {

/\*

char \*string: указатель на строку

\*/

int count = 0;

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

if (string[i] == ' ') {

count++;

}

}

\*spaces\_count = count;

}

// Сортировка строк по возрастанию количества пробелов

// Вывод кол-ва перестановок и наибольшего числа пробелов

void sort\_strings\_by\_spaces\_count(char \*\*strings, int strings\_count) {

/\*

char \*\*strings: массив строк

int strings\_count: его размерность

\*/

int swaps\_count = 0;

int max\_spaces\_count = 0;

for (int i = 0; i < strings\_count - 1; i++) {

for (int j = i + 1; j < strings\_count; j++) {

int str\_i\_spaces\_count;

get\_spaces\_count(strings[i], &str\_i\_spaces\_count);

int str\_j\_spaces\_count;

get\_spaces\_count(strings[j], &str\_j\_spaces\_count);

if (str\_i\_spaces\_count > max\_spaces\_count) {

max\_spaces\_count = str\_i\_spaces\_count;

}

if (str\_j\_spaces\_count > max\_spaces\_count) {

max\_spaces\_count = str\_i\_spaces\_count;

}

if (str\_i\_spaces\_count > str\_j\_spaces\_count) {

char \*tmp = strings[i];

strings[i] = strings[j];

strings[j] = tmp;

swaps\_count++;

}

}

}

printf("При сортировке было совершено перестановок: %d\n", swaps\_count);

printf("Наибольшее количество пробелов среди строк: %d\n", max\_spaces\_count);

}

// Освобождаем выделенную для строк память

void free\_memory(char \*\*strings) {

/\*

char \*\*strings: массив строк

\*/

for (int i = 0; i < STRINGS\_COUNT; i++) {

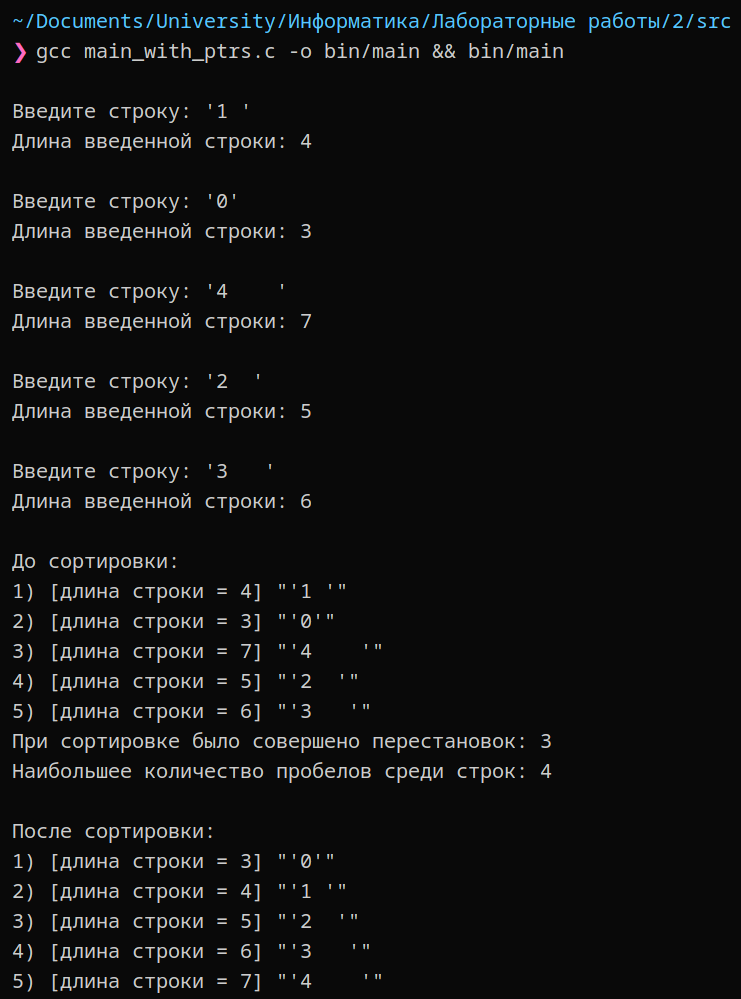
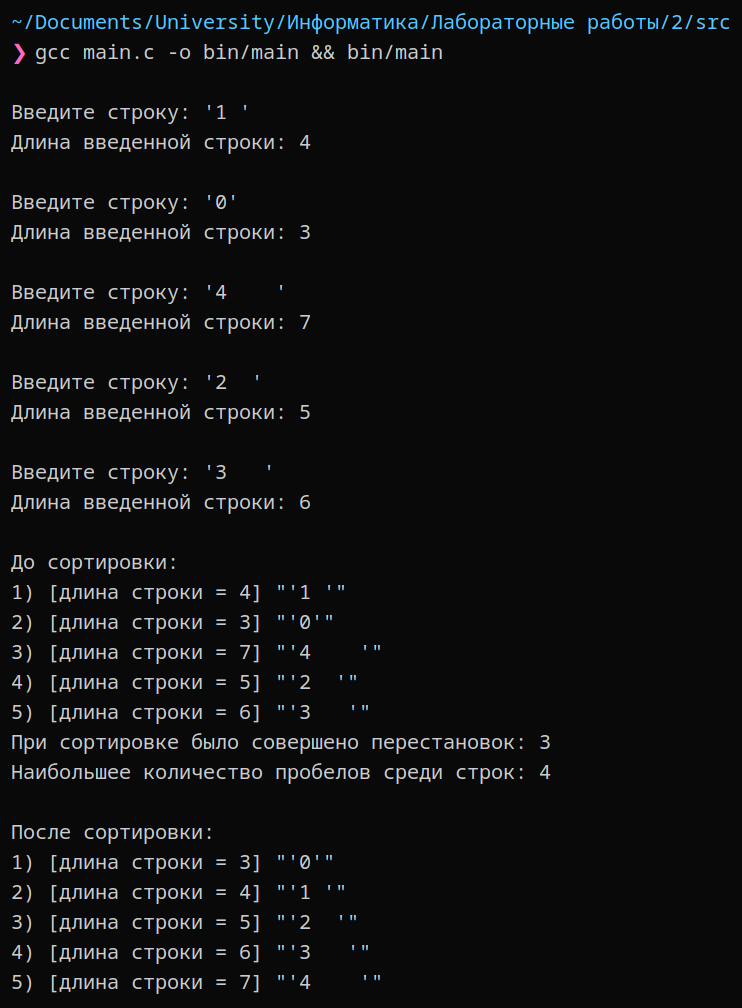
free(strings[i]);

}

}

**Результаты:**

Слева представлен вывод программы 4.1, справа – программы 4.2.



**Вывод:** Нами были освоены правила написания и использования функций в языке СИ. Мы научились использовать указатели при обработке массивов данных и применили полученные знания на практике.