

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных
систем

Лабораторная работа №10.1

по дисциплине: Основы программирования
тема: «Обработка текстовых файлов»

Выполнил: ст. группы ПВ-201
Машуров Дмитрий Русланович

Проверил:
Притчин Иван Сергеевич
Брусенцева Валентина
Станиславовна

Белгород 2021 г.

Лабораторная работа № 10.1

Обработка текстовых файлов

Цель работы: получение навыков работы с потоками.

Задания для подготовки к работе

1. Изучить организацию работы с текстовыми и бинарными потоками.
2. Разработать алгоритм и составить программы для решения каждой из двух задач соответствующего варианта. В бинарных файлах информация хранится в машинном представлении.
3. Для бинарных файлов составить программу для создания файла.
4. Подобрать тестовые данные.

Задание варианта №16

Из данного текстового файла удалить слова, длина которых превышает данное число n .

Выполнение:

1. Описание алгоритма и выделение подзадач

Исходя из условия задачи, будем считывать каждое слово в буфер *buf* из входного файла, сравнивать его длину с данным числом *n* и записывать считанное слово *buf* в выходной файл, если его длина меньше, чем *n*, иначе – переходим к следующему слову

Выделение подзадач:

- 1) Удаление слов из файла, длина которых превышает заданное число
- 2) Считывание слова с файла в буфер

2. Блок-схема с укрупнёнными блоками



3. Спецификации подзадач

1) Считывание слов с файла в буфер

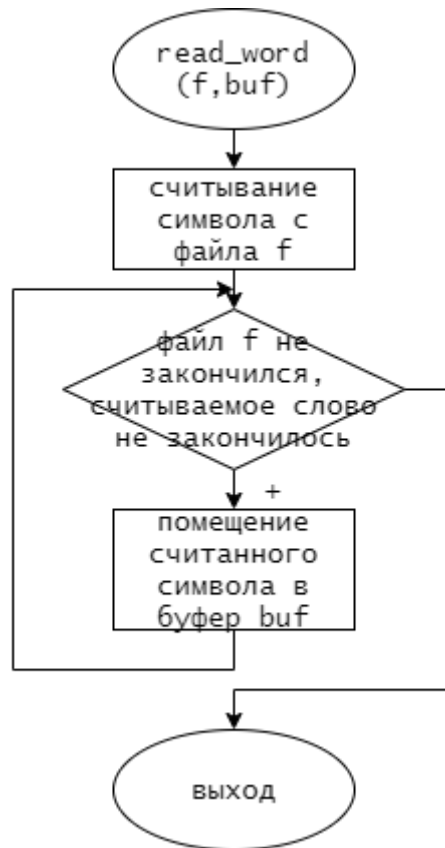
a. Выделение подзадач:

i. Очистка буфера

b. Заголовок: `void read_word(FILE *f, char *buf)`

c. Назначение: считывает слово с файла `f` в буфер `buf`

d. Блок-схема:

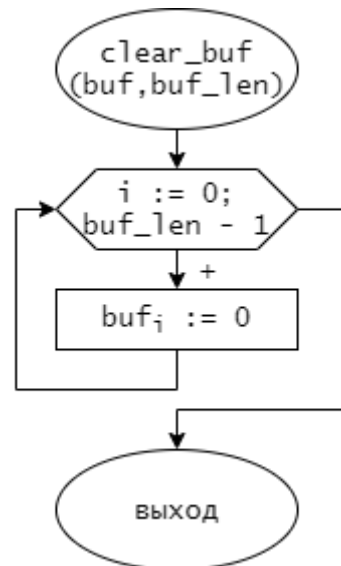


i. Очистка буфера

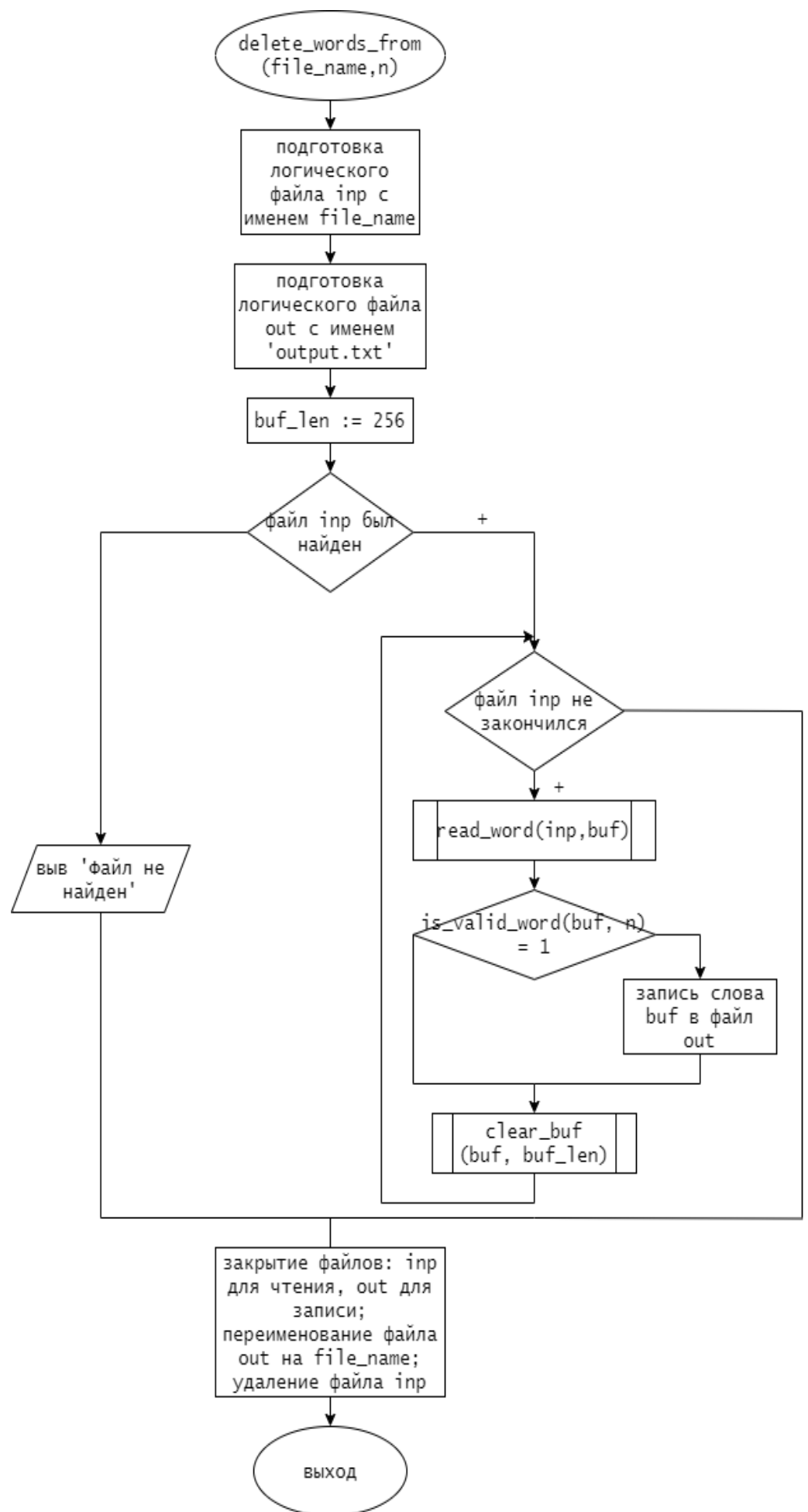
a) Заголовок: `void clear_buf(char buf[], size_t buf_len)`

b) Назначение: очищает буфер `buf` длины `buf_len`

c) Блок-схема:



- 2) Удаление слов из файла, длина которых превышает заданное число
- Выделение подзадач:
 - Закрытие, переименование и удаление файлов
 - Определение, является ли длина слова меньше заданного числа
 - Заголовок: `void delete_words_from(char *file_name, int n)`
 - Назначение: удаляет слова из файла с именем `file_name`, длина которых превышает (или равна) `n`
 - Блок-схема:

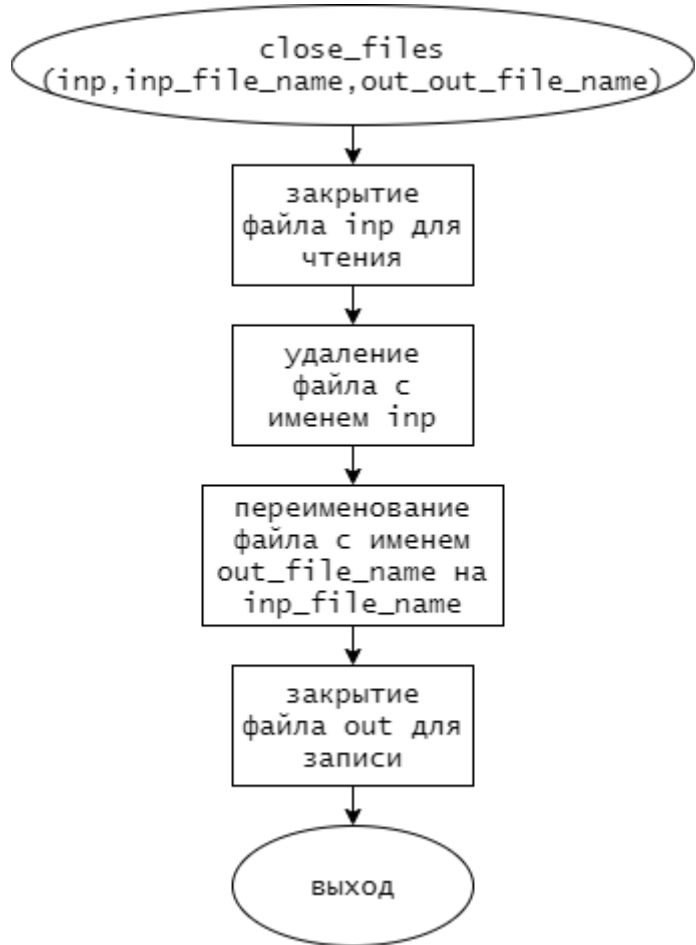


i. Закрытие, переименование и удаление файлов

a) Заголовок: `void close_files(FILE *inp, char *inp_file_name, FILE *out, char *out_file_name)`

b) Назначение: закрывает для чтения, а затем удаляет файл `inp`; переименовывает файл с именем `out_file_name` на `inp_file_name` (имя входного файла); закрывает файл `out` для записи;

c) Блок-схема:



ii. Определение, является ли длина слова меньше заданного числа

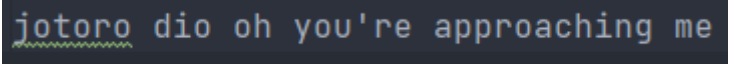
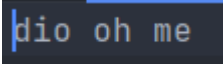
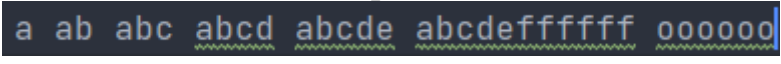
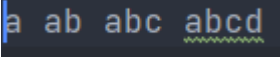
a) Заголовок: `int is_valid_word(char *w, int n)`

b) Назначение: возвращает '1', если длина слова *w* меньше, чем *n*, иначе – '0'

c) Блок-схема:



4. Тестовые данные

№	Вход	Выход
1	<p>“inp.txt”</p>  <p>4</p>	
2	<p>“inp.txt”</p>  <p>5</p>	

5. Текст программы

```
#include <stdio.h>

/* считывает слово с файла f в буфер buf */
void read_word(FILE *f, char *buf) {
    char c = fgetc(f);
    size_t i = 0;
    while (c != '\n' && c != ' ' && c != '\377') {
        buf[i] = c;
        i++;
        c = fgetc(f);
    }

    buf[i] = 0;
}

/*
 * возвращает '1', если длина слова w не больше n,
 * иначе - '0'
 */
int is_valid_word(char *w, int n) {
    size_t c = 0;

    while (*w != '\0' && c < n) {
        c++; w++;
    }

    if (c == n) {
        return 0;
    } else {
        return 1;
    }
}

/* очищает буфер buf длины buf_len */
void clear_buf(char buf[], size_t buf_len) {
    for (size_t i = 0; i < buf_len; ++i) {
        buf[i] = 0;
    }
}

/*
 * закрывает для чтения, а затем удаляет файл inp;
```

```

    * переименовывает файл с именем out_file_name на inp_file_name (имя
    входного файла);
    * закрывает файл out для записи;
    */
void close_files(FILE *inp, char *inp_file_name, FILE *out, char
*out_file_name) {
    fclose(inp);
    fclose(out);

    rename(out_file_name, inp_file_name);
    remove(inp_file_name);
}

/*
 * удаляет слова из файла с именем file_name, длина
 * которых превышает (или равна) n
 */
void delete_words_from(char *file_name, int n) {
    FILE *inp = fopen(file_name, "r");
    FILE *out = fopen("output.txt", "w");
    size_t buf_len = 256;

    if (inp != NULL) {
        while (feof(inp) == 0) {
            char buf[buf_len];
            read_word(inp, buf);

            if (is_valid_word(buf, n)) {
                fprintf(out, "%s ", buf);
            }

            clear_buf(buf, buf_len);
        }
    } else {
        perror("A file with this name wasn't found");
    }

    close_files(inp, file_name, out, "output.txt");
}

int main() {
    printf("Input source file name\n");
    char file_name[50];
    scanf("%s", file_name);

    printf("Input number\n");
    int n;
    scanf("%d", &n);

    delete_words_from(file_name, n);
}

```

6. Результаты работы:

Пример №1:

```
Input source file name  
inp.txt  
Input number  
4
```

Пример №2:

```
Input source file name  
inp.txt  
Input number  
5
```