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.

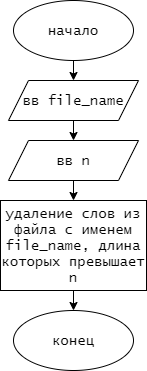
**Выполнение:**

1. **Описание алгоритма и выделение подзадач**

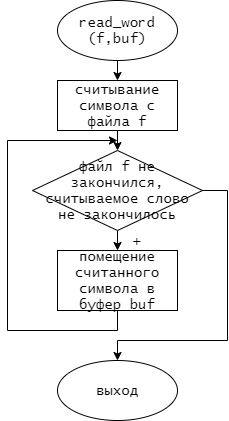
Исходя из условия задачи, будем считывать каждое слово в буфер из входного файла, сравнивать его длину с данным числом и записывать считанное слово в выходной файл, если его длина меньше, чем , иначе – переходим к следующему слову

Выделение подзадач:

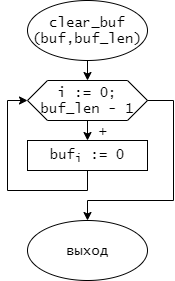
1. Удаление слов из файла, длина которых превышает заданное число
2. Считывание слова с файла в буфер
3. **Блок-схема с укрупнёнными блоками**



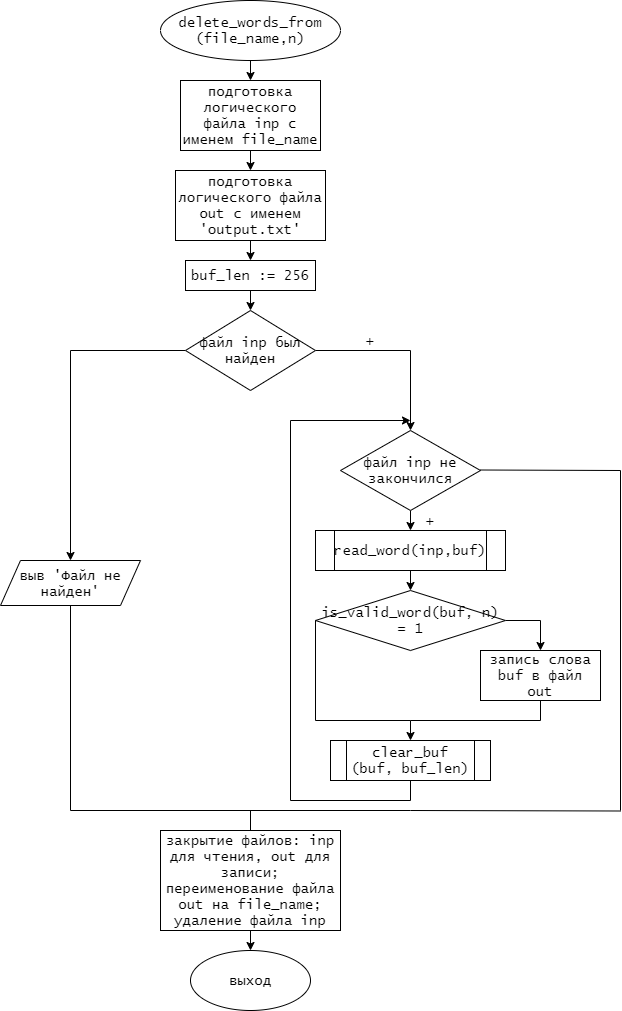
1. **Спецификации подзадач**
2. Считывание слов с файла в буфер
   1. Выделение подзадач:
      1. Очистка буфера
   2. Заголовок: void read\_word(FILE \*f, char \*buf)
   3. Назначение: считывает слово с файла f в буфер buf
   4. Блок-схема:



* + 1. Очистка буфера
       1. Заголовок: void clear\_buf(char buf[], size\_t buf\_len)
       2. Назначение: очищает буфер buf длины buf\_len
       3. Блок-схема:



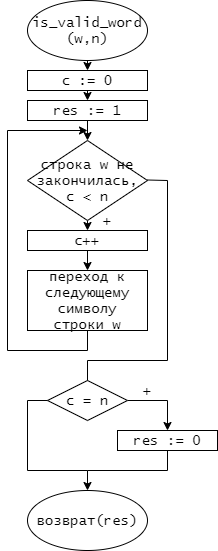
1. Удаление слов из файла, длина которых превышает заданное число
   1. Выделение подзадач:
      1. Закрытие, переименование и удаление файлов
      2. Определение, является ли длина слова меньше заданного числа
   2. Заголовок: void delete\_words\_from(char \*file\_name, int n)
   3. Назначение: удаляет слова из файла с именем file\_name, длина которых превышает (или равна) n
   4. Блок-схема:



* + 1. Закрытие, переименование и удаление файлов
       1. Заголовок: void close\_files(FILE \*inp, char \*inp\_file\_name, FILE \*out, char \*out\_file\_name)
       2. 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;
       3. Блок-схема:



* + 1. Определение, является ли длина слова меньше заданного числа
       1. Заголовок: int is\_valid\_word(char \*w, int n)
       2. Назначение: возвращает '1', если длина слова w меньше, чем n, иначе – '0'
       3. Блок-схема:



1. **Тестовые данные**

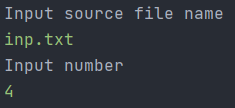
|  |  |  |
| --- | --- | --- |
| № | Вход | Выход |
| 1 | “inp.txt”    4 |  |
| 2 | “inp.txt”    5 |  |

1. **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);  
  
}

1. **Результаты работы:**

*Пример №1:*



*Пример №2:*

