**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**

(**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**)» (**МАИ)**

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра 304 Вычислительные машины, системы и сети

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ ВЫСОКОГО УРОВНЯ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10

«Изучение функций для работы со строками»

Вариант 4

Выполнил:

студент группы М30-209Б-19: Кузнецов И.И.

Проверил:

Доцент к. 304, к.т.н., Новиков П. В.

Старший преподаватель, Ивашенцев И. В.

Москва 2020

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**

(**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**)» (**МАИ)**

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра 304 Вычислительные машины, системы и сети

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ ВЫСОКОГО УРОВНЯ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10

«Изучение функций для работы со строками»

Вариант 4

Выполнил:

студент группы М30-209Б-19: Мухаметгалиев А.Р.

Проверил:

Доцент к. 304, к.т.н., Новиков П. В.

Старший преподаватель, Ивашенцев И. В.

Москва 2020

**Содержание**

1. **Задание.**
2. **Структурная схема алгоритма программы и подпрограмм.**
3. **Текст программы.**
4. **Результаты работы программы.**
5. **Вывод.**

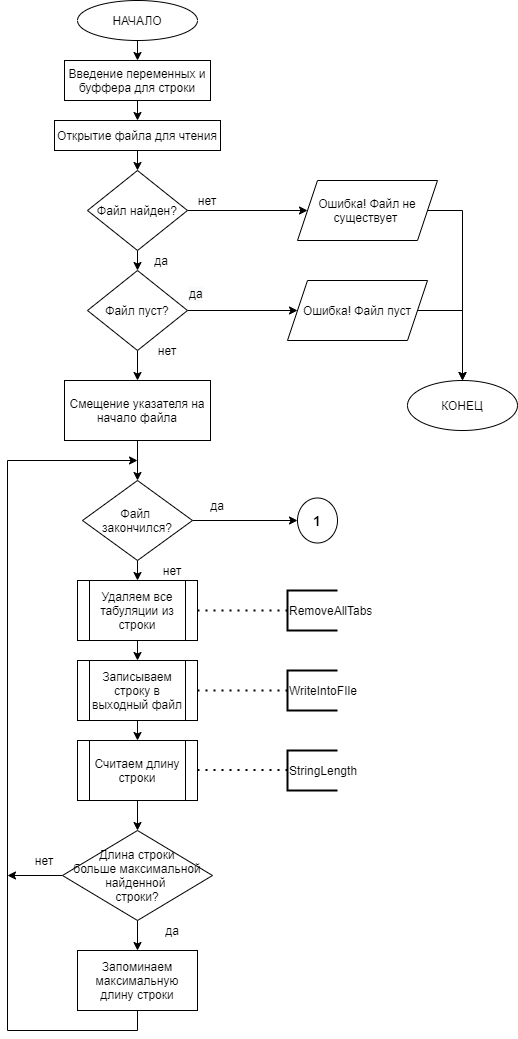
**Задание**

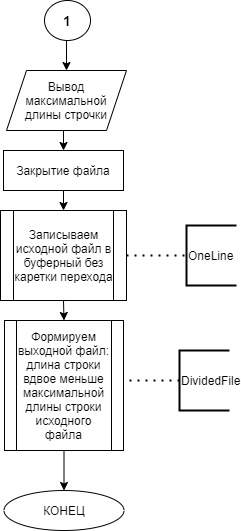
**Вариант 4**

Разработать программу и подпрограмму (подпрограммы), выполняющую действия согласно варианту задания:

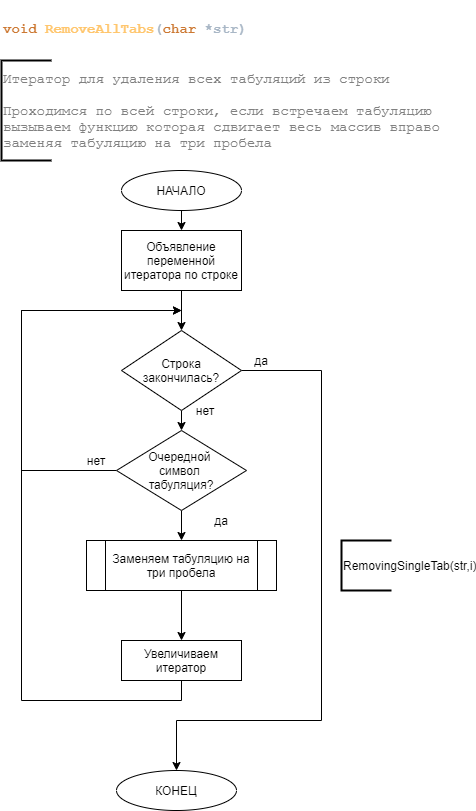
Формирование строк текста одинаковой длины (в два раза меньше максимальной в исходном тексте) путем разбиения или объединения строк. Каждый символ табуляции во входном файле заменяется на 3 пробела. Сформатированный текст записать в выходной файл.

**Структурная схема алгоритма программы.**

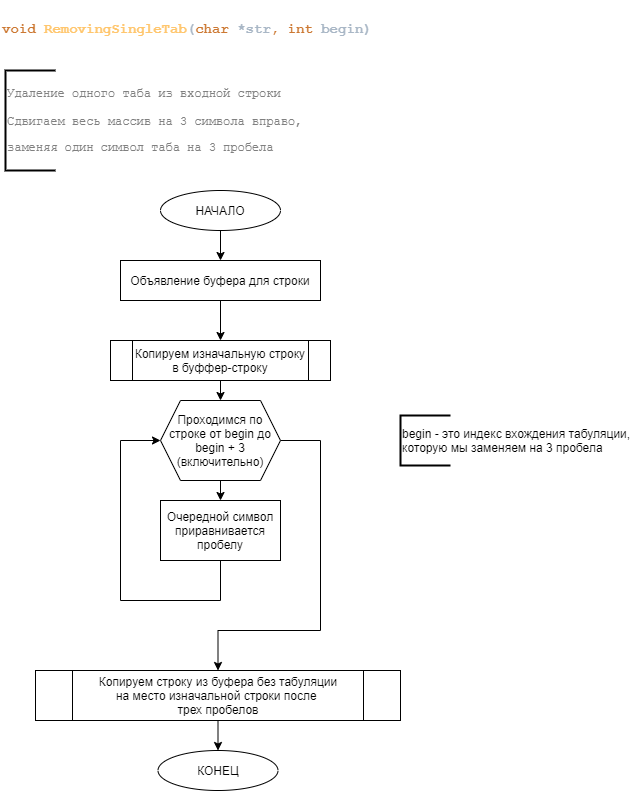




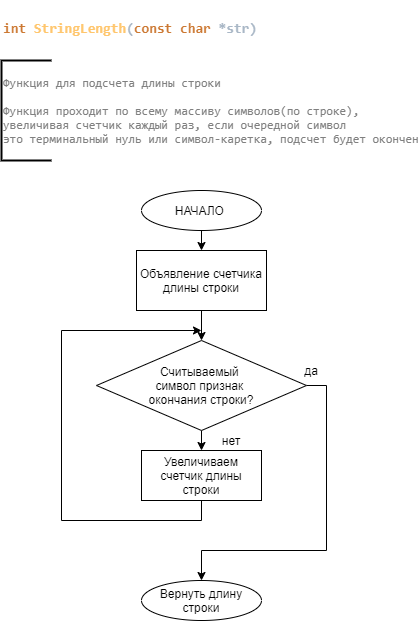
**Структурная схема алгоритма подпрограммы RemoveAllTabs.**

****

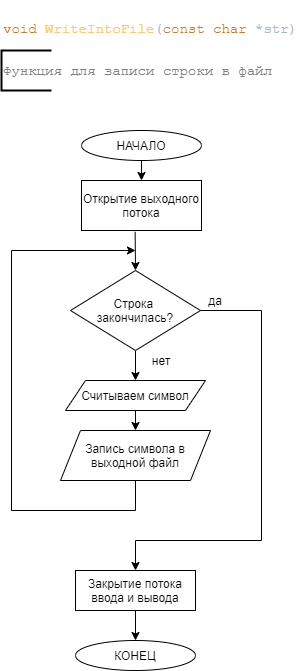
**Структурная схема алгоритма подпрограммы RemovingSingleTab.**



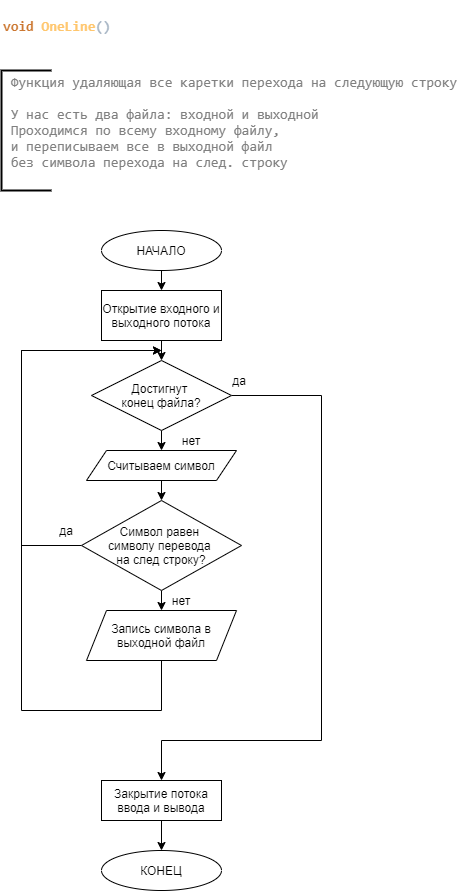
**Структурная схема алгоритма подпрограммы StringLength.**



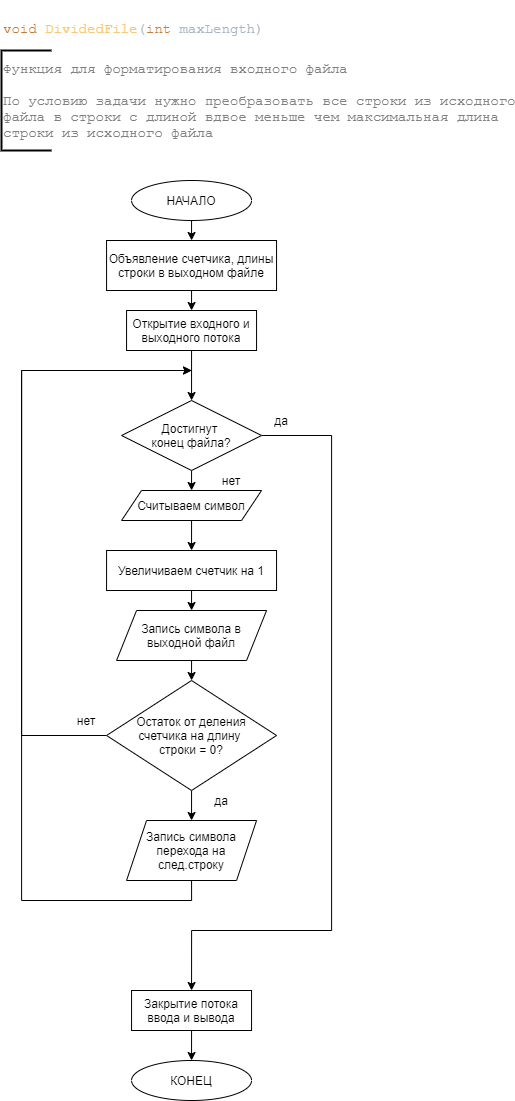
**Структурная схема алгоритма подпрограммы WriteIntoFile.**



**Структурная схема алгоритма подпрограммы OneLine.**

****

**Структурная схема алгоритма подпрограммы DividedFile.**



**Текст программы**

#include <stdio.h>

#include "string.h"

#include <locale.h>

/\*

Формирование строк текста одинаковой длины (в два раза меньше

максимальной в исходном тексте) путем разбиения или объединения строк.

Каждый символ табуляции во входном файле заменяется на 3 пробела.

Сформатированный текст записать в выходной файл

\*/

void RemoveAllTabs(char\* str);

void RemovingSingleTab(char\* str, int begin);

int StringLength(const char\* str);

void WriteIntoFile(const char\* str);

void OneLine();

void DividedFile(int maxLength);

int main() {

setlocale(LC\_ALL, "Rus");

const char\* filenameIn = "../Fin.txt";

remove("../Fout.txt");

remove("../Buff.txt");

FILE\* in;

char str[256]; //буфферная строка

in = fopen(filenameIn, "rt");

int maxLength = 0;

int length;

if (in == NULL) { //проверка на наличие файла

perror("ERROR with a reading file ");

return -1;

}

if (getc(in) == EOF) { //проверка на пустой файла

printf("File is empty!\n");

return -2;

}

rewind(in); //переход на начало файла

while (fgets(str, sizeof(str), in) != NULL) { //считать построчно, пока не конец файла

RemoveAllTabs(str);

WriteIntoFile(str);

length = StringLength(str);

if (length > maxLength) maxLength = length;

}

printf("Max length: %i", maxLength);

fclose(in);

OneLine();

DividedFile(maxLength);

return 0;

}

/\*\*

\* Итератор для удаления всех табуляций из строки

\*

\* Проходимся по всей строки, если встречаем табуляцию

\* вызываем функцию которая сдвигает весь массив вправо

\* заменяя табуляцию на три пробела

\* @param str входная строка

\*/

void RemoveAllTabs(char\* str) {

int i = 0;

while (str[i] != '\0') {

if (str[i] == '\t') {

RemovingSingleTab(str, i);

}

i++;

}

}

/\*\*

\* Удаление одного таба из входной строки

\*

\* Сдвигаем весь массив на 3 символа вправо, заменяя

\* один символ таба на 3 пробела

\* @param str входная строка из которой нужно удалить таб

\* @param begin индекс в строки, где находится таб

\*/

void RemovingSingleTab(char\* str, int begin) {

char buff[256];

memcpy(buff, str, strlen(str) + 1);

for (int i = begin; i <= begin + 3; ++i) {

str[i] = ' ';

}

memcpy(&str[begin + 3], &buff[begin + 1], strlen(str));

}

/\*\*

\* Функция для подсчета длины строки

\*

\* Функция проходит по всему массиву символов(по строке),

\* увеличивая счетчик каждый раз, если очередной символ

\* это терминальный нуль или символ-каретка, подсчет будет окончен

\* @param str - входная строка

\* @return длина строки

\*/

int StringLength(const char\* str) { //подсчет длины строки

int counter = 0; //счетчик длины строки и итератор по этой строке

while (str[counter] != '\0' && str[counter] != '\n')

counter++;

return counter;

}

/\*\*

\* Функция для записи строки в файл

\*

\* @param str записываемая входная строка

\*/

void WriteIntoFile(const char\* str) {

FILE\* out; //выходной поток

out = fopen("../Fout.txt", "a+t"); //открываем выходной поток

int i = 0; //итератор по строке

while (str[i] != '\0') { //пока не закончилась строка

fwrite(&str[i], sizeof(char), 1, out); //записываем очередной символ в файл

i++;

}

fclose(out); //закрываем выходной поток

}

/\*\*

\* Функция удаляющая все каретки перехода на следующую строку

\*

\* У нас есть два файла: входной и выходной

\* Проходимся по всему входному файлу,

\* и переписываем все в выходной файл

\* без символа перехода на след. строку

\*/

void OneLine() {

int buff;

const char\* filenameIn = "../Fout.txt";

const char\* filenameOut = "../Buff.txt";

FILE\* in;

FILE\* out;

in = fopen(filenameIn, "rt");

out = fopen(filenameOut, "a+t");

rewind(in);

rewind(out);

while ((buff = getc(in)) != EOF) { //считать посимвольно, пока не конец файла

if (buff != '\n') {

fwrite(&buff, sizeof(char), 1, out);

}

}

fclose(in);

fclose(out);

}

/\*\*

\* Функция для форматирования входного файла

\*

\* По условию задачи нужно преобразовать все строки из исходного

\* файла в строки с длиной вдвое меньше чем максимальная длина

\* строки из исходного файла

\* @param maxLength максимальная длина строки из исходного файла

\*/

void DividedFile(int maxLength) {

int stringLength = maxLength / 2;

int counter = 0;

int buff;

const char\* filenameIn = "../Buff.txt";

const char\* filenameOut = "../Fout.txt";

FILE\* in;

FILE\* out;

remove(filenameOut);

in = fopen(filenameIn, "rt");

out = fopen(filenameOut, "a+t");

while ((buff = getc(in)) != EOF) {

counter++;

fwrite(&buff, sizeof(char), 1, out);

if (counter % stringLength == 0) {

fwrite("\n", sizeof(char), 1, out);

}

}

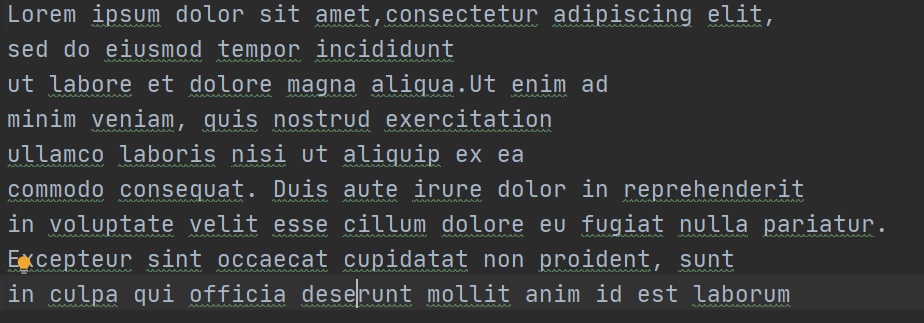
fclose(in);

fclose(out);

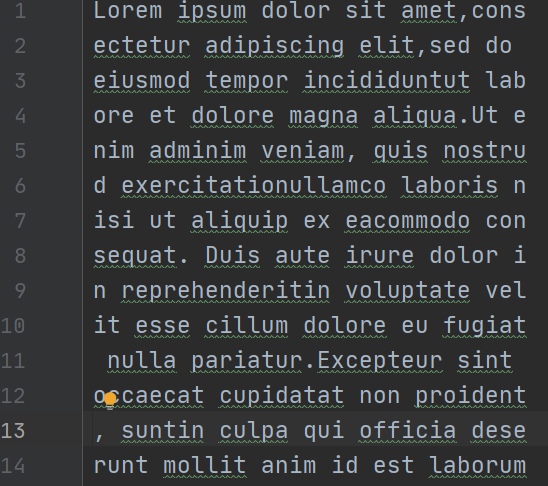
}

**Результаты работы программы**

**Входные данные**

****

**Результат**

****

Вывод: Программа работает корректно

**Вывод**

Мы разработали программу и подпрограммы, выполняющие действия согласно варианту задания:

Формирование строк текста одинаковой длины (в два раза меньше максимальной в исходном тексте) путем разбиения или объединения строк. Каждый символ табуляции во входном файле заменяется на 3 пробела. Сформатированный текст записать в выходной файл.

А также изучили функции из библиотеки <string.h> такие как для strlen и memcopy.