**Правительство Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего профессионального образования   
"Национальный исследовательский университет   
"Высшая школа экономики"**

Московский институт электроники и математики Национального

исследовательского университета "Высшая школа экономики"

Департамент прикладной математики

**ОТЧЕТ**

**По лабораторной работе №9**

**РАБОТА С ТЕКСТОВЫМИ ФАЙЛАМИ**

**По курсу «Алгоритмизация и программирование»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | ФИО студента | | Номер группы | Дата |
| Кармаев Александр Андреевич | БПМ211 | 15.10.2021 |
|  |
|  |
|  |

**Москва – 2021 г.**

**ЗАДАНИЕ (вариант №13)**

*Программа должна быть разбита на несколько функций и обязательно содержать:*

*1. Функции формирования исходного файла;*

*2. Функции вывода результата работы программы;*

*3. Одну или более функций, реализующих вычислительную часть алгоритма.*

*Основная программа должна содержать только операторы вызова функций.*

*В задании подразумевается, что исходный файл остаётся неизменным.*

*Изображение выглядит как текст

Автоматически созданное описание*

**РЕШЕНИЕ**

#include <stdio.h> // в данной библиотеке содержатся функции для работы с файлами

#include <string.h>

void fill\_file\_f() { // функция для записи строк в исходный файл f

FILE\* fp = fopen("f.txt", "w"); // создаём указатель на структуру типа FILE - поток файла // открываем файл(первый параметр) в определённом режиме(2-ой параметр - для записи\чтения\...)

int f = 1; // флаг для возможности прекратить создание новых строк в исходном файле

printf(" Input several strings by task condition:\n");

while (f != 0)

{

char str[256];

gets(str); // считываем всю введённую строку до нажатия Enter // используем gets, вместо scanf, т.к. scanf считает пробел и Enter концом для считывания, а это не позволит нам считывать пробелы

fputs(str, fp); // записываем строку в файл

putc('\n', fp); // добавляем символ перехода на следующую строку

printf("If you want to continue input '1', else '0' ");

scanf("%d", &f);

getchar(); // прописываем getchar, чтобы избавиться от последствий оставшегося после scanf символа перехода на следующую строку(Enter)

}

fclose(fp); // закрываем поток fp

}

void create\_file\_g() { // функция для записи ответов в файл g

place:;

FILE\* fp = fopen("f.txt", "r"); // поток для чтения из исходного файла

int nums[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }; // массив, в котором будем хранить количество повторений n-символьных слов

int now = 0; // количество символов текущего слова

char ch = getc(fp); // переменная для хранения текущего символа из файла

while (ch != EOF) { // EOF будет возвращено после считывания всех элементов файла

if (ch != ' ' && ch != '\n') {

now++;

if (now > 10) { // если вдруг пользователь введёт слово более чем из 10 символов, то просим его заново начать вводить строки

printf("\nInput strings again, couse length of some word more than 10\n");

fill\_file\_f();

goto place; // начинаем функцию заново, так как у нас появились новые строки и прошлые результаты нам не нужны

}

}

else {

if (now != 0) { // делаем обработку, чтобы не выйти за границы массива, когда now == 0 => будет браться значение по индексу = -1

nums[now - 1] += 1;

now = 0;

}

}

ch = getc(fp);

}

fclose(fp);

// теперь записываем нужные значения в файл g

fp = fopen("g.txt", "w");

for (int i = 0; i < 10; i++) {

fprintf(fp, "%d %d", i + 1, nums[i]); // функция записывает форматированную строку в поток

if (i != 9) // делаем, чтобы в самом конце не создать 11 строку

putc('\n', fp);

}

fclose(fp);

}

void print\_file\_g() { // функция для вывода ответов из файла g

FILE\* fp = fopen("g.txt", "r");

char ch;

printf("\n Answer:\n");

while ((ch = getc(fp)) != EOF)

printf("%c", ch);

fclose(fp);

}

int main(void) {

// используем наши функции в нужном порядке для решения задачи

fill\_file\_f();

create\_file\_g();

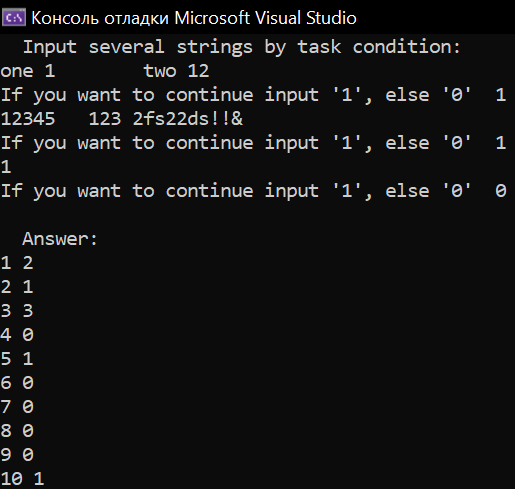
print\_file\_g();

return 0;

}

**ТЕСТЫ**

**Тест № 1** // обычный пример

****

**Тест № 2** // введены один или несколько пробелов, или ничего (сразу Enter)

Изображение выглядит как текст

Автоматически созданное описание

**Тест № 3 //** пример в котором происходит обработка при введении слова больше, чем из 10 символов

**Изображение выглядит как текст

Автоматически созданное описание**

**Тест № 4** // пример содержит обработку на некорректный ввод

Изображение выглядит как текст

Автоматически созданное описание