Государственное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ   
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«Донецкий национальный технический университет»

Д09.03.04-ПОИС.20-20/5847.ЛР

***Кафедра*** искусственного интеллекта  
 и системного анализа

Лабораторная работа №4

по дисциплине "Основы программирования"

на тему: "Обработка текстовой информации"

Проверили:

\_\_\_\_\_\_\_\_\_\_\_\_ асс. А.П. Семёнова

(дата, подпись)

\_\_\_\_\_\_\_\_\_\_\_\_ асс. И.В. Савицкая

(дата, подпись)

Выполнил:

\_\_\_\_\_\_\_\_\_\_\_\_ ст.гр.ПИ-20г М. А. Евсеев

(дата, подпись)

# Описание работы с файлами

Откроем файл «In.txt», если это невозможно – запишем в файл «Out.txt» сообщение об ошибке и закончим программу.

Открыв файл, считываем каждую строку до тех пор, пока не дойдем до конца файла. Считая каждую строку, проверим ее посимвольно на правильность записи вещественного числа. Для этого пропустим отступы в строке, если они имеются, после проверим, что символ – цифра, иначе пропускаем проверку данной строки. Зная что после отступов идет цифра, пропускаем последующие цифры, образующие число перед точкой. Когда цифры кончились - символ должен являться точкой, иначе неверная запись вещественного числа. После точки символы – цифры, а после цифр возможны отступы. В случае успешной проверки строки – увеличим счетчик правильно записанных вещественных чисел .

Проверив все строки файла, запишем в результирующий файл «Out.txt» сообщение о количестве правильно записанных слов и закончим программу.

Донецк – 2020

## Задание

1. Выполнить задание, согласно заданному варианту.
2. Для используемых в программе стандартных функций языка Си для работы со строками привести алгоритмы.
3. При написании программы обязательно использование адресной арифметики. В листинге, к строкам программы, относящимся к адресной арифметике принести комментарии.

Вариант 9

Отредактировать заданное предложение, удаляя из него слова, которые встречаются в предложении заданное число раз.

# Ход работы

## Постановка задачи

Пользователь вводит предложение и число, определяющее минимальное количество встречающихся слов для их удаления, после чего программа выводит редактированное предложение.

Входные данные:

str[512] – символьный массив, предложение;

n – целочисленное число.

Промежуточные данные:

p[2]s – символьный массив указателей;

t, i, j, x – целочисленные числа, счетчики цикла;

bl – целочисленное число, логическое значение 1 или 0;

k – целочисленное число, счетчик повторений слов.

c – символ.

Выходные данные:

str\_copy[512] – символьный массив, предложение.

# Алгоритм

# Листинг программы

Код программы, созданной в Visual Studio 2019:

#include <stdio.h>

#include <string.h>

#include <locale.h>

#include <Windows.h>

#define letter(x) (x >= 'A' && x <= 'Z' || x >= 'a' && x <= 'z') || (x >= 'А' && x <= 'Я' || x >= 'а' && x <= 'я')

#define lenw(x) (arr[x][1] - arr[x][0]+1)

//преобразует строку в нижний регистр

**void** **to\_lower**(**char**\* p) {

**char**\* c;

**for** (**size\_t** i = **0**; i < strlen(p); i++) {

c = &p[i]; //присваиваем указателю адрес элемента массива

(\*c >= 'A' && \*c <= 'Z') ? (\*c = \*c + 'a' - 'A') : **0**;

(\*c >= 'А' && \*c <= 'Я') ? (\*c = \*c + 'a' - 'A') : **0**;

}

}

**int** **main**() {

SetConsoleCP(**1251**);// установка кодовой страницы win-cp 1251 в поток ввода

SetConsoleOutputCP(**1251**); // установка кодовой страницы win-cp 1251 в поток вывода

**char** \*p[**2**], str[**512**], str\_copy[**512**] = "", c = '#';

**int** n, t, k = **0**, i = **0**, x = **0**, j = **0**, arr[**100**][**3**], bl;

p[**0**] = &str; //присваиваем указателю адрес массива str

\*(p + **1**) = &str\_copy; //сложение адреса и присвоение указателю адреса массива str\_copy

printf("Введите предложение:**\n\n**");

**while** (p[**0**][i - **1**] != '\n') {

scanf("%c", &p[**0**][i]);

//если буква

**if** (letter(p[**0**][i])) {

arr[j][**0**] = i;

**while** (letter(p[**0**][i])) {

i++;

scanf("%c", &p[**0**][i]);

}

arr[j][**1**] = i - **1**;

j++;

}

i++;

}

p[**0**][i] = '\0';

strcat(p[**1**], p[**0**]);

to\_lower(\*p);

**for** (i = **0**; i < j; i++) {

k = **0**;

**for** (x = **0**; x < j; x++) {

bl = **1**;

//находим повторяющиеся слова, записуем их количество в массив arr[i][2]

**if** (lenw(i) == lenw(x) && i != x) {

**for** (t = **0**; t < lenw(i) && t != -**1**; t++) {//проверяем сходность каждой буквы, в случае несходности

(p[**0**][arr[i][**0**] + t] != p[**0**][arr[x][**0**] + t]) ? (t = -**2**) : **0**;

}

(t != -**1**) ? (k++) : **0**;

}

}

arr[i][**2**] = k + **1**;

}

printf("**\n**Введите n - минимальное количество повторений слов для их удаления:**\n\n**");

scanf("%d", &n);

**if** (n) {

k = **0**;

**for** (i = **0**; i < j; i++) //если количество слов не меньше n, переносим строку с начала повторяющего слова на количество символов равная длине слова.

{

**if** (!bl) {

k += lenw(i - **1**);

}

bl = **1**;

**if** (n <= arr[i][**2**]) {

// от начала повторяющего слова до конца строки

**for** (t = arr[i][**0**] - k; t < strlen(p[**1**]) - lenw(i); t++) {

p[**1**][t] = p[**1**][t + lenw(i)];

}

p[**1**][t] = '\0';

bl = **0**;

}

}

printf("**\n**Редактированное предложение:**\n\n**%s", \*(p + **1**));

}

**else** {

printf("%s", p[**0**]);

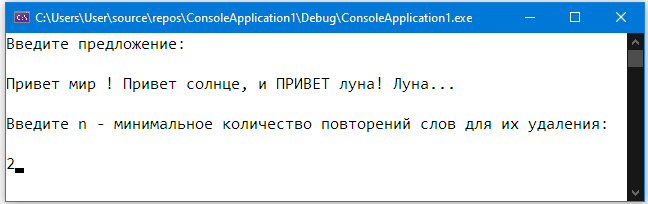
}

system("pause");

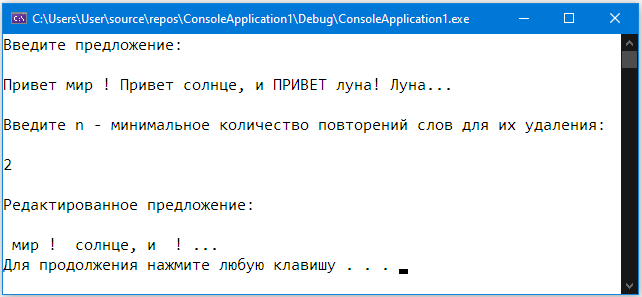
}

# Контрольные примеры

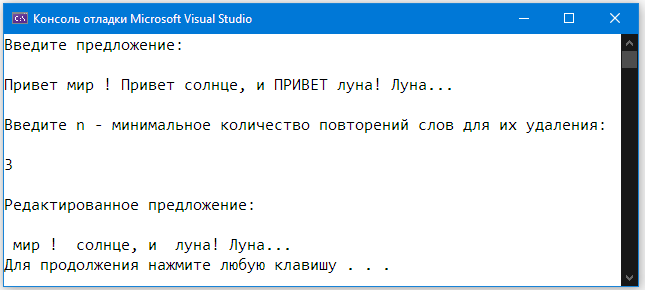
Запустив программу, пользователь вводит предложение и необходимое количество повторяемых слов для их удаления:



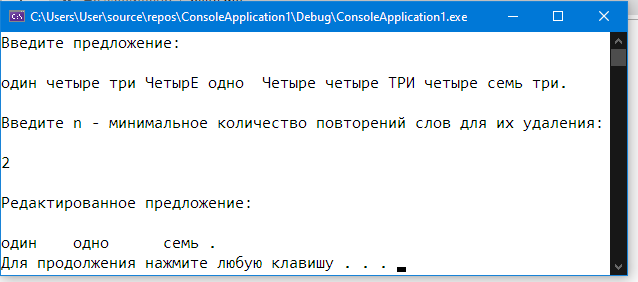
После ввода которых, программа выдает нам отредактированное предложение:



Пример №2:



Пример №3:



Чтобы закрыть программу достаточно нажать любую клавишу.

Вывод: Я научился использовать функции стандартной библиотеки для работы со строками. Умею преобразовывать текст и использовать адресную арифметику.