Приложение 1. Тесты (

run file with args: ./low\_letter\_words\_max\_ru.out sample\_input.txt sample\_result.txt

<< Test! Hello world!!! Did I write Somebody?... Maybe yes, maybe no

<< Тест! Привет мир!!! Пока мир, доброе утро.

<< Да нет да Нет нет

<< Low-letter words in sample\_input.txt: 12

Содержание файла sample\_input.txt:

Test! Hello world!!! Did I write Somebody?... Maybe yes, maybe no

Тест! Привет мир!!! Пока мир, доброе утро.

Да нет да Нет нет

Содержание файла sample\_result.txt:

world write yes maybe no мир мир доброе утро нет да нет

Приложение 1. Код.

#include <stdlib.h>

#include <stdio.h>

#include <windows.h>

int is\_lowercase\_char(char ch) {

    // Проверка на строчные буквы английского и русского алфавитов

    return ((ch >= 'a' && ch <= 'z') || (ch >= (char)224 && ch <= (char)255));

}

int is\_uppercase\_char(char ch) {

    // Проверка на заглавные буквы английского и русского алфавитов

    return ((ch >= 'A' && ch <= 'Z') || (ch >= (char)192 && ch <= (char)223));

}

int main(int argc, char \* argv[]) {

    SetConsoleOutputCP(1251);  // Windows-1251

    SetConsoleCP(1251);

    if(argc < 3) {

        // Проверка ввода

        printf("Error: 2 file names were expected, but were received %d\n", argc - 1);

        exit(-1);

    }

    const char \* inputFileName = argv[1];

    const char \* outputFileName = argv[2];

    FILE \* file = fopen(inputFileName, "r");

    FILE \* outFile = fopen(outputFileName, "w");

    if(file == NULL) {

        printf("Error: cant open file %s\n", inputFileName);

        exit(1);

     }

    if(outFile == NULL) {

        printf("Error: cant open outFile %s\n", outputFileName);

        exit(1);

    }

    size\_t allocated = 15;

    size\_t i\_tempword = 0;

    char \*tempword = malloc(sizeof(char) \* allocated);

    if (tempword == NULL) {

        fprintf(stderr, "Ошибка выделения памяти.\n");

        exit(1);

    }

    size\_t counter = 0;

    int inWord = 0;

    int isLowletter = 0;

    char c = 0;

    while (!feof(file)) { // Пока не достигнут конец файла

        c = fgetc(file); // Читаем символ

        if(!inWord && is\_lowercase\_char(c)) { // Если прочитали строчную букву и inWord == False

            inWord = 1;

            isLowletter = 1;

        } else if(!inWord && is\_uppercase\_char(c)) {  // Если прочитали заглавную букву и inWord == False

            inWord = 1;

            isLowletter = 0;

        } else if(inWord && is\_uppercase\_char(c)) { // Если прочитали заглавную букву и inWord == 1

            // Слово не малобуквенное

            isLowletter = 0;

        /\*

        Если символ не является заглавной или строчной буквой

        или файл закончился

        \*/

        } else if((!is\_lowercase\_char(c) && !is\_uppercase\_char(c)) || feof(file)) {

            if (isLowletter) {

                printf("\e[4;32m");

                for (int i = 0; i < i\_tempword; ++i) {

                    fputc(tempword[i], outFile);

                }

                fputc(' ', outFile);

            }

            for (int i = 0; i < i\_tempword; ++i) {

                printf("%c", tempword[i]);

            }

            counter += isLowletter;

            i\_tempword = 0;

            isLowletter = 0;

            inWord = 0;

        }

        if(inWord) {

            tempword[i\_tempword] = c;

            ++i\_tempword;

            if (i\_tempword >= allocated) {

                allocated \*= 2;

                char \*tempmem = realloc(tempword, allocated \* sizeof(char));

                if (tempmem == NULL) {  // Проверка на успешное выделение

                    fprintf(stderr, "Ошибка расширения памяти.\n");

                    free(tempword);

                    return 1;

                }

                tempword = tempmem;

            }

        } else if (!feof(file)) {

            printf("\e[0;0m%c", c);

        }

    }

    printf("\n\e[0;0mLow-letter words in \e[0;3m%s\e[0;0m: %zu\n\e[0;0m", inputFileName, counter);

    free(tempword);

    fclose(file);

    fclose(outFile);

    return 0;

}