МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Программирование»

Тема: Использование указателей

Студент гр. 1304 _____ Крупин Н. С.

Преподаватель _____ Чайка К. В.

Санкт-Петербург

2021

Цель работы.

Освоение работы с указателями и динамической памятью.

Задание.

Вариант 2.

«Напишите программу, которая форматирует некоторый текст и выводит результат на консоль.

На вход программе подается текст, который заканчивается предложением "Dragon flew away!"

Предложение (кроме последнего) может заканчиваться на:

. (точка);

; (точка с запятой);

? (вопросительный знак).

Программа должна изменить и вывести текст следующим образом:

каждое предложение должно начинаться с новой строки;

табуляция в начале предложения должна быть удалена;

все предложения, которые заканчиваются на '?' должны быть удалены;

текст должен заканчиваться фразой "Количество предложений до n и количество предложений после m", где n – количество предложений в изначальном тексте (без учета терминального предложения "Dragon flew away!") и m – количество предложений в отформатированном тексте (без учета предложения про количество из данного пункта).

Порядок предложений не должен меняться.

Статически выделять память под текст нельзя.

Пробел между предложениями является разделителем, а не частью какого-то предложения».

Выполнение работы.

Фукция main начинается с создания указателя для хранения адреса начала массива предложений text и вызова функции scan_text для записи этого массива, в качестве параметров передаются адрес указателя text, строка с перечислением символов окончания предложения и строка с терминальным предложением – признаком окончания ввода. Возвращаемое функцией scan_text количество считанных предложений записывается в целочисленную переменную count. Далее вызывается функция remove_questions для обработки массива, в качестве параметров передаются указатель text и количество count, возвращаемое функцией новое количество предложений записывается в целочисленную переменную new_count. Вывод результата состоит из вызова функции print_text для печати на экран изменённого массива, в качестве параметров передаются указатель text, количество new_count и строка-разделитель (в данном случае состоит из одного символа перевода строки), и отдельной печати предложения со сравнением искомых n и m (равны соответственно count-1 и new_count-1). Функция main заканчивается освобождением динамической памяти, для этого вызывается функция free_text с параметрами text и new_count.

Функция scan_text принимает в качестве аргументов text – адрес указателя на начало пока не существующего массива предложений, строку с перечислением символов конца предложения sep_str и строку с терминальным предложением – признаком конца ввода – end_sen. Предназначена для записи введённого текста в динамически созданный массив предложений. Использует целочисленные переменные i_sen (номер записываемого предложения, начиная с 0) и i_ch (номер записываемого символа в предложении, начиная с 0). Для выделения динамической памяти используется стандартная функция realloc, также используются стандартные функции strchr для определения наличия записанного символа в строке sep_str и strcmp для определения совпадения записанного предложения с

end_sen, предварительное исключение пробела-разделителя и символов табуляции перед предложением реализуется с помощью вызова функции scanf с аргументом-строкой, состоящей из одного пробела. Функция возвращает количество записанных предложений, накопленное в i_sen.

Функция remove_questions принимает в качестве аргументов указатель на начало массива text и количество предложений count. Предназначена для удаления вопросительных предложений. Использует переменную sen – указатель на адрес блока памяти, в котором лежит обрабатываемое предложение. Также использует стандартные функции strchr для определения наличия вопросительного знака в предложении, free для освобождения блока памяти удаляемого предложения и memmove для смещения оставшейся части массива адресов text на позицию адреса удалённого предложения. Функция возвращает значение count, уменьшающееся с каждым удалением.

Функция print_text принимает в качестве аргументов указатель на начало массива text, количество предложений count и строку-разделитель sep, которая будет выводиться после каждого предложения. Предназначена для печати текста на экран. Использует переменную sen – указатель на адрес блока памяти, в котором лежит печатаемое предложение. Функция ничего не возвращает.

Функция free_text принимает в качестве аргументов указатель на начало массива text и количество предложений count. Предназначена для освобождения динамической памяти, выделенной для хранения текста. Использует переменную sen – указатель на адрес выделенного под предложение блока памяти. Для освобождения памяти используется стандартная функция free. Функция ничего не возвращает.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в таблице 1. Все результаты соответствуют ожидаемым.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	\tab\tc. b3*cg; \t\tg? \t\tThg. Fgh? Dragon flew away! Dragon flew away! absc.	ab\tc. b3*cg; Thg. Dragon flew away! Количество предложений до 5 и количество предложений после 3
2.	Dragon flew away!	Dragon flew away! Количество предложений до 0 и количество предложений после 0

Выводы.

Были изучены основы работы с указателями и динамической памятью в языке С.

Разработана программа, выполняющая запись введённого текста в динамическую память, обработку в соответствии с заданным условием и вывод результата на экран.

Программа может быть ускорена организацией «блочного» выделения динамической памяти (не после каждого символа, а после некоторого определённого их количества).

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Krupin_Nikita_lb3.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


int scan_text(char*** text, char* sep_str, char* end_sen) {
     int i_sen, i_ch;

    *text = NULL;
    i_sen = 0;
    do{
        *text = realloc(*text, sizeof(char*)*(i_sen+1));
        (*text)[i_sen] = NULL;
        i_ch = 0;

        scanf(" ");
        do {
            (*text)[i_sen] = realloc((*text)[i_sen],
                                              sizeof(char)*(i_ch+1));
            scanf("%c", (*text)[i_sen]+i_ch);
            i_ch++;
        } while (!strchr(sep_str, (*text)[i_sen][i_ch-1]));

        (*text)[i_sen] = realloc((*text)[i_sen],
                                              sizeof(char)*(i_ch+1));
        (*text)[i_sen][i_ch] = '\0';
        i_sen++;
    } while (strcmp((*text)[i_sen-1], end_sen));

    return i_sen;
}



int remove_questions(char** text, int count){
    char** sen;

    sen = text;
    while (sen < text+count)
        if (strchr(*sen, '?')){
            free(*sen);
            count--;
            memmove(sen, sen+1, sizeof(char*)*(text+count-sen));
        } else sen++;

    return count;
}
```

```c
void print_text(char** text, int count, char* sep){
    char** sen;

    for (sen = text; sen < text+count; sen++)
        printf("%s%s", *sen, sep);
}



void free_text(char** text, int count){
    char** sen;

    for (sen = text; sen < text+count; sen++)
        free(*sen);
    free(text);
}



int main(){
    char** text; int count, new_count;

    count = scan_text(&text, ";.?!", "Dragon flew away!");

    new_count = remove_questions(text, count);

    print_text(text, new_count, "\n");
    printf("Количество предложений до %d и количество предложений
                                    после %d", count-1, new_count-1);

    free_text(text, new_count);
    return 0;
}
```