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
Спасов Д.В.

Преподаватель Чайка К.В.

Санкт-Петербург

2021

Цель работы.

Написание программы для форматирования текста по определенным правилам и последующего его вывода

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, в которых есть цифры внутри слов, должны быть удалены (это не касается слов, которые начинаются/заканчиваются цифрами). Если слово начинается с цифры, но имеет и цифру в середине, удалять его все равно требуется (4a4a).
- 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.

txt — указатель на двумерный массив строк.

end_of_txt — содержит сторку Dragon flew away!

indicator — переменная , являющаяся индикатором для прекращения ввода строк с клавиатуры

checker - переменная являющяяся флагом для начала проверки предложения на наличие цифр в словах

old_sentence_counter – считает количество введенных предложений

new_sentence_counter – считает количесто предложений после форматирования

del_sentence – является флагом для удаления конкретного предложения несоответсвтующего условию

до того момента как переменная indicator = 1 программа будет считыват предложения с клавиатуры выделять под них память и отсеивать не подходящие по условию

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1	never more. sqw12ds 312; Dragon flew away!	never more. Dragon flew away! Количество предложений до 2 и количество предложений после 1	Пример из условия, выполнился верно

Выводы.

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

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb3.c

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

int main()
{
   char **txt;
   char end_of_txt[] = "Dragon flew away!";
   int old_sentence_counter, new_sentence_counter, counter, i, indicator, checker, del_sentence, z;
   indicator = 1;
   char c;
   txt = malloc(sizeof(char*));
   new_sentence_counter = 0;
   old_sentence_counter = 0;

   while (indicator){

      txt = realloc(txt,sizeof(char*)*(new_sentence_counter + 2));
      txt[new_sentence_counter] = malloc(sizeof(char)*2);
      scanf("%c",&c);
      txt[new_sentence_counter][0] = c;
```

```
        txt[new_sentence_counter][1] = '\0';
        counter = 0;
        checker = 0;
        while ((c!='.') && (c!=';') && (c!='?') &&
(strcmp(txt[new_sentence_counter],end_of_txt))){

            counter = counter +1;
            txt[new_sentence_counter] =
realloc(txt[new_sentence_counter],sizeof(char)*(counter+2));
            scanf("%c",&c);
            txt[new_sentence_counter][counter] = c;
            txt[new_sentence_counter][counter+1] = '\0';
        }
        for(i = 0; i < counter; i++){
            if (isdigit(txt[new_sentence_counter][i])){
                del_sentence = 0;
                z = i;
                if (i != 0){
                    while (del_sentence != 1){
                        if (z == 0){
                            break;
                        }
                        if (txt[new_sentence_counter][z] == ' ' ||
txt[new_sentence_counter][z] == '.' || txt[new_sentence_counter][z] == ';' ||
txt[new_sentence_counter][z] == '?' || txt[new_sentence_counter][z] == '\n'){
                            break;
                        }
                        if (isalpha(txt[new_sentence_counter][z])){
                            del_sentence = 1;
```

```
                    break;
                }
                z = z - 1;
            }
            if (del_sentence == 1){
                z = i;
                while (del_sentence != 2){
                    if (z == counter){
                        break;
                    }
                    if (txt[new_sentence_counter][z] == ' ' ||
txt[new_sentence_counter][z] == '.' || txt[new_sentence_counter][z] == ';' ||
txt[new_sentence_counter][z] == '?' || txt[new_sentence_counter][z] == '\n'){
                        break;
                    }
                    if (isalpha(txt[new_sentence_counter][z])){
                        del_sentence = 2;
                        checker = 1;
                        break;
                    }
                    z = z + 1;
                }
            }
        }
    }
    scanf("%c",&c);
    if ((strcmp(txt[new_sentence_counter],end_of_txt)) == 0){
        indicator = 0;
```

```
        }

        if (checker == 1){
            free(txt[new_sentence_counter]);
            new_sentence_counter = new_sentence_counter - 1;
        }

        new_sentence_counter = new_sentence_counter + 1;
        old_sentence_counter = old_sentence_counter + 1;
    }

    for(i = 0; i < new_sentence_counter; i++){
        printf("%s\n",txt[i]);
    }

    for(i = 0; i < new_sentence_counter; i++){
        free(txt[i]);
    }

    free(txt);
    printf("Количество предложений до %d и количество предложений после %d\n",old_sentence_counter-1,new_sentence_counter-1);

    return 0;
}
```