**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**

Студентка гр. 0382 _____ Михайлова О.Д.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

**Цель работы.**

Научиться работать с символьными массивами, динамической памятью и указателями в си.

**Задание.**

Вариант 4.

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 (\t, ' ') в начале предложения должна быть удалена.
- Все предложения, в которых есть число 555, 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**

**Основные теоретические положения.**

Указатель – это переменная, содержащая адрес другой переменной.

Синтаксис объявления указателя:

<тип_переменной_на_которую_ссылается_указатель>* <название переменной>;

Унарная операция **&** даёт адрес объекта.  Она применима только к переменным и элементам массива.

Для работы с динамической памятью используются следующие функции:

- malloc ( *void* malloc (size_t size) )* - выделяет блок из **size** байт и возвращает указатель на начало этого блока;
- calloc ( *void* calloc (size_t num, size_t size)* ) - выделяет блок для **num** элементов, каждый из которых занимает **size** байт и инициализирует все биты выделеного блока нулями;
- realloc ( *void* realloc (void* ptr, size_t size)* ) - изменяет размер ранее выделенной области памяти на которую ссылается указатель **ptr**. Возвращает указатель на область памяти, измененного размера;
- free ( *void free (void* ptr)* ) - высвобождает выделенную ранее память.

Строка в Си - массив символов, массив строк это двумерный массив символов, где каждая строка - массив, хранящий очередную символьную строку.

Функция int isalnum (int c) стандартной библиотеки ctype.h проверяет, является ли символ с буквой или цифрой.

Функция size_t strlen (const char* str) стандартной библиотеки string.h возвращает длину строки str.

**Выполнение работы.**

Переменные:

int size_sent – переменная, в которую записывается размер памяти, выделяемой в динамической памяти для хранения предложения;

int length – переменная, в которую записывается текущий размер вводимого предложения;

char *sent – переменная, которая хранит адрес на блок памяти, выделенной для хранения вводимого предложения;

int c – переменная, в которую записывается вводимый символ;

int size_text – переменная, в которую записывается размер памяти, выделяемой в динамической памяти для хранения текста;

int len_text – переменная, в которую записывается текущий размер текста;

char** text – переменная, которая хранит адрес на блок памяти, выделенной для хранения текста;

char* end_sent – переменная, в которую записывается предложение, которым заканчивается ввод текста;

int count1 – переменная, в которую записывается количество предложений в тексте до преобразований;

int count2 – переменная, в которую записывается количество предложений в тексте после преобразований;

Функции:

В функции char* readsentence() посимвольно считывается предложение с помощью функции getchar(), пока не будет введен один из символов, которым может заканчиваться предложение. Перед этим выделяется блок памяти для хранения предложения с помощью функции malloc и затем, после считывания нового символа, при необходимости блок памяти расширяется с помощью функции realloc. После считывания в конец предложения записывается символ «\0». Функция возвращает адрес введенного предложения.

Функция int stop_input(char *sent1, char  *sent2) принимает на вход два предложения и посимвольно сравнивает их с помощью цикла for. Если предложения одинаковые, то функция возвращает 1, если нет – 0.

Функция char* delete_t(char *sent) принимает на вход предложение и проверяет, есть ли в начале символы пробела, табуляции или перевода строки. Если да, то эти символы удаляются и все символы предложения сдвигаются влево с помощью цикла for. Функция возвращает адрес измененного предложения.

Функция int delete_sent(char *sent) принимает на вход предложение  и с помощью цикла for и функции isalnum проверяет, есть ли в нем число 555. Если есть, то функция возвращает 0, если нет – 1.

В функции int main() сначала выделяется блок памяти для хранения текста с помощью функции malloc. В цикле while вызываются функции readsentence и delete_t и увеличивается счетчик количества предложений до преобразований count1. Далее вызывается функция delete_sent, и если в

предложении нет числа 555, то предложение добавляется в текст и увеличивается счетчик количества предложений после преобразований count2. При необходимости блок памяти для хранения текста расширяется с помощью функции realloc. Затем с помощью функции stop_input сравнивается предложение, которым заканчивается ввод текста, с только что введенным. Если они одинаковые, то считывание прерывается. После цикла while с помощью функции free происходит очистка сначала память каждого предложения, потом блока памяти, выделенного для хранения текста. В конце выводятся количество предложений до преобразований и количество предложений после преобразований без учета предложения, которым заканчивается ввод текста.

Разработанный программный код см. в приложении А.

**Тестирование.**

Результаты тестирования см. в приложении Б.

**Выводы.**

Были изучены основные управляющие конструкции языка для работы с одномерными и двумерными массивами.

Разработана программа, которая выполняет считывание текста и помещает его в двумерный массив строк, для которого выделяется динамическая память. Реализованы функции, с помощью которых выполняется обработка текста. Затем программа выводит данные и освобождает память.

## ПРИЛОЖЕНИЕ А

## ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: pr_lb_3.c

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

#define INIT_SIZE 50
#define TRUE 1

char* readsentence(){
    int size_sent = INIT_SIZE;
    int length = 0;

    char *sent = malloc(size_sent*sizeof(char));
    int c;

    while (TRUE){
        c = getchar();
     sent[length++]=c;
        if(length == size_sent){
            size_sent += INIT_SIZE;
            sent = realloc(sent, size_sent);
        }
        if(c == '.' || c == ';' || c == '?' || c == '!'){
            break;
        }
    }

    if(length>0){
        sent[length]='\0';
        return sent;
    }
}

int stop_input(char *sent1, char  *sent2){
    int a, i;
    for (i=0; i < strlen(sent2); i++){
        if (sent1[i] == sent2[i]){
            a = 1;
        }
        else{
            a = 0;
        }
    }
    return a;
}

char* delete_t(char *sent){
    int i = 0;
    int a = 0;
    while(sent[i] == ' ' || sent[i] == '\t' || sent[i] == '\n'){
         int j;
         for (j=0; j<strlen(sent)-1; j++){
              sent[j] = sent[j+1];
         }
```

```
            a++;
        }
        sent[strlen(sent)-a] = '\0';

        return sent;
    }

    int delete_sent(char *sent){
        int a = 1;
        int i;
        for (i=0; i<strlen(sent)-3; i++){
            if ((sent[i] == '5') && (sent[i+1] == '5') && (sent[i+2]
== '5') && (!(isalnum(sent[i-1]))) && (!(isalnum(sent[i+3])))){
                a = 0;
            }
        }
        return a;
    }

    int main(){
        int count1 = 0;
        int count2 = 0;
        char *sent;

        int size_text = INIT_SIZE;
        int len_text = 0;
        char** text = malloc(size_text*sizeof(char*));

        char* end_sent = "Dragon flew away!";
        while(TRUE){
            sent = readsentence();
            sent = delete_t(sent);
            count1++;
            if (delete_sent(sent)){
                count2++;
                text[len_text++] = sent;
            }
            if (len_text == size_text){
                size_text += INIT_SIZE;
                text = realloc(text, size_text*sizeof(char*));
            }
            if (stop_input(end_sent, sent)){
                break;
            }
        }

        for (int i = 0; i<len_text; i++){
            printf("%s\n", text[i]);
            free(text[i]);
        }
        free(text);


        printf("Количество предложений до %d и количество предложений
после %d", count1-1, count2-1);

        return 0;

}
```

# ПРИЛОЖЕНИЕ Б

# ТЕСТИРОВАНИЕ

Таблица Б.1 – Результаты тестирования

| № п/п | Входные данные | Выходные данные | Комментарии |
|---|---|---|---|
| 1. | Nulla facilisi.<br>Class aptenT taciti sociosqu ad litora torquent per cOnubia nostra, per inceptos himenaeos. 40 Nu555lla rutrum feugiat felis a pharetra.<br>Sed finibus magna et mauris elementum tempus? Integer at quam et erat iaculis iaculis hendrerit a te4llus? Donec at nunc ac mauris suscipit venenatis.<br>Sed finibus magna et mauris elementum tempus? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi co7ndimentum 555 ex justo, nec pharetra mauris vestibulum a.<br>Suspendisse quis mi neque7.<br>1 Vivamus eu nibh rhoncus, da456pibus ex non, sodales mi.<br>Donec accumsan convallis ipsum vitae lacinia. Donec accumsan convallis ipsum vitae lacinia.<br>Fusce finibus sapien magna, quis scelerisque ex sodales tristique. Nulla facilisi.<br>Lorem ipsum dolor sit amet, consectetur adipiscing elit.<br>Dragon flew away! | Nulla facilisi.<br>Class aptenT taciti sociosqu ad litora torquent per cOnubia nostra,<br>per inceptos himenaeos.<br>40 Nu555lla<br>rutrum feugiat felis a pharetra.<br>Sed finibus magna et mauris elementum tempus?<br>Integer at<br>quam et erat iaculis iaculis hendrerit a te4llus?<br>Donec at<br>nunc ac mauris suscipit venenatis.<br>Sed finibus magna et mauris elementum tempus?<br>Lorem ipsum<br>dolor sit amet, consectetur adipiscing elit.<br>Suspendisse quis mi neque7.<br>1 Vivamus eu nibh rhoncus, da456pibus ex non, sodales mi.<br>Donec accumsan convallis ipsum vitae lacinia.<br>Donec accumsan<br>convallis ipsum vitae lacinia.<br>Fusce finibus sapien magna, quis scelerisque ex sodales tristique.<br>Nulla facilisi.<br>Lorem ipsum dolor sit amet, consectetur adipiscing elit.<br>Dragon flew away!<br>Количество предложений до 16 и количество предложений после 15 | Результат верный |
| 2. | Odio contentiones sed cu, usu commodo prompta prodesset id.<br>Eam id posse dictas voluptua, veniam 555 laoreet oportere no mea, quis regione suscipiantur mea an.<br>Vel in dicant cetero phaedrum, usu populo interesset cu, eum ea facer nostrum pericula. Dragon flew away! | Odio contentiones sed cu, usu commodo prompta prodesset id.<br>Vel in dicant cetero phaedrum, usu populo interesset cu, eum ea facer nostrum pericula.<br>Dragon flew away!<br>Количество предложений до 3 и количество предложений после 2 | Результат верный |

| | | | |
|---|---|---|---|
| 3. | Ius dicat feugiat no, vix cu modo dicat principes.<br>Magna copiosae apeirian ius at. Odio contentiones sed cu, usu commodo prompta prodesset id.Per in illud petentium iudicabit, integre sententiae pro no.<br>Eu eam dolores lobortis percipitur, 555quo te equidem deleniti disputando.<br>Sale liber et vel. Elitr accommodare deterruisset eam te, vim munere pertinax consetetur at. Dragon flew away! | Ius dicat feugiat no, vix cu modo dicat principes.<br>Magna copiosae apeirian ius at. Odio contentiones sed cu, usu commodo prompta prodesset id.<br>Per in illud petentium iudicabit, integre sententiae pro no.<br>Eu eam dolores lobortis percipitur, 555quo te equidem deleniti disputando.<br>Sale liber et vel.<br>Elitr accommodare deterruisset eam te, vim munere pertinax consetetur at.<br>Dragon flew away!<br>Количество предложений до 7 и количество предложений после 7 | Результат верный |