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

Вариант 2

Студент гр. 0382 Афанасьев Н. С.

Преподаватели Чайка К. В.,
Жангиров Т. Р

Санкт-Петербург
2020

Цель работы.

Изучение указателей и динамического выделения памяти для массивов, освоение работы со строками и двумерными массивами в языке C.

Задание.

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).

Выполнение работы.

Для начала, в теле функции **main()** объявляются переменные: *terminationString* (константа, хранящая финальную строку "*Dragon flew away!*"), *n* (кол-во предложений до), *m* (кол-во предложений после), *textSize*=50 (начальное предполагаемое количество предложений), *text* (для хранения текста). Дальше через метод calloc выделяется место для 50 указателей на тип *char* (предложения).

Далее начинается цикл *while*, в котором считаются и отбираются предложения. Сначала объявляются переменные *sentence* и *lastSym* для хранения текущего предложения и последнего символа в нём.

Выполняется функция **char* getSentence(char* lastSym).** В начале выделяется место для предложения (изначально на 50 символов). Далее, в переменную *sym* записывается первый символ, причём все пробельные до него игнорируются. В цикле *while*, проверяется наличие памяти для следующего символа (если нет, то она выделяется с помощью метода *realloc*), символ записывается в массив предложения, проверяется наличие завершающего знака (точка, точка с запятой, вопросительный или восклицательный знак): если такой есть – цикл завершается, если нет – считается следующий символ. Функция возвращает полученное предложение, а последний символ записывает в переменную, на которую указывает *lastSym*.

Далее, в конструкции *switch*, отбираются нужные предложения: если *lastSym* равен *"."* или *";"*, то *n* и *m* увеличиваются, предложение записывается в *text*, проверяется наличие памяти для следующего предложения (если нет, то она выделяется с помощью метода *realloc*); если *lastSym* равен *"?"*, то только увеличивается *n* (так как предложения с "?" не нужны); если *lastSym* равен *"!"*, то предложение записывается в *text*, и если оно равно завершающей строке, то цикл завершается.

В конце выводятся все предложения, очищается память, где были массив указателей и строки, и выводятся значения *m* и *n*.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	hello. world? Dragon flew away!	hello. Dragon flew away! Количество предложений до 2 и количество предложений после 1	Верно
2.	. . . Dragon flew away!	. . . Dragon flew away! Количество предложений до 3 и количество предложений после 3	Верно
3.	a. bb? ccc; Dragon flew away!	a. ccc; Dragon flew away! Количество предложений до 3 и количество предложений после 2	Верно

Выводы.

Был изучена работа с указателями, строками, двумерными массивами и динамичной памятью в языке С.

Разработана программа, считывающая текст, удаляющая вопросительные предложения и пробельные символы между предложениями, считающая количество предложений в тексте до и после выполнения программы, выводящая полученные предложения и значения на экран. Программа динамически выделяет память для хранения текста и предложений.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

char* getSentence(char* lastSymbol){
    short sentenceSize = 50;
    char* sentence = calloc(sentenceSize, sizeof(char));

    char sym;
    do sym = getchar(); while (sym == ' ' || sym == '\t' || sym == '\n');

    short count = 0;
    short endOfSentence = 0;
    while(!endOfSentence){
        if(count == sentenceSize){
            sentenceSize += 15;
            sentence = realloc(sentence, sentenceSize * sizeof(char));
        }

        sentence[count++] = sym;
        if(sym == '.' || sym == ';' || sym == '?' || sym == '!')
endOfSentence = 1;
        else sym = getchar();
    }

    *lastSymbol = sym;
    sentence[count] = '\0';
    return sentence;
}

int main()
{
    const char* terminationString = "Dragon flew away!";
    short n = 0, m = 0;
    short textSize = 20;
    char** text = calloc(textSize, sizeof(char*));

    short endOfText = 0;
    while(!endOfText){
        char lastSym;
        char* sentence = getSentence(&lastSym);

        switch (lastSym){
            case '?':
                n++;
                break;
            case '.':
            case ';':
                n++;
                text[m++] = sentence;
                if(m == textSize){
                    textSize += 10;
```

```
                    text = realloc(text, textSize * sizeof(char*));
                }
                break;
            case '!':
                if(!strcmp(sentence, terminationString)) {
                    text[m] = sentence;
                    endOfText = 1;
                }
                break;
        }
    }

    for(int i = 0; i < m+1; i++){
        puts(text[i]);
        free(text[i]);
    }
    free(text);
    printf("Количество предложений до %d и количество предложений
после %d", n, m);
    return 0;
}
```