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. 0382 _____ Куликов М.Д

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Получение опыта работы с динамическим выделением памяти , символьными массивами и указателями.

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, 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).

Основные теоретические положения.

printf() - функция, выводящая данные в консоль

scanf() - функция, считывающая данные из консоли

getchar() - функция, считывающая 1 символ из консоли

malloc() - функция, выделяющая определенное количество памяти, возвращает адрес на выделенный блок памяти.

realloc() - функция, изменяющая размер данного блока памяти, возвращает новый адрес блока памяти.

isalpha() - функция, проверающая, является ли символ буквой.

isdigit() - функция, проверяющая, является ли символ числом.

strcmp() - функция, сравнивающая две строки.

free() - функция, высвобождающая память в данном блоке.D

Выполнение работы.

В начале функции main объявляются переменные, отвечающие за количество предложений до изменений, после изменений, первоначальный размер текста. Также объявляется переменная для предложения и для текста, выделяется память для текста.

Далее начинается цикл while, который будет выполняться до тех пор, пока программа не достигнет команды break. С помощью пользовательской функции input_sentence() выполняется считывание предложения, потом с помощью пользовательской функции no_spaces(),удаляются лишние пробелы, табуляция и символы переноса строки. После этого с помощью пользовательской функции sentence_selection() проверяется, удовлетворяет ли предложение заданным условиям. Если удовлетворяет, то оно записывается в массив строк, после чего проверяется, является ли оно конечным предложением. Если не удовлетворяет, то мы освобождаем память,

выделенную под него. И в работе с предложениями, и в работе со строками, проверяется, хватает ли места для записи.Если не хватает, то выделяется доп память с помощью realloc(). После получения измененного массива со строками мы выводим его, выводим количество предложений до и после работы программы, после чего выполням высвобождения памяти.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	One. Two; Three? Dragon flew away!	One. Two; Three? Dragon flew away! Количество предложений до 3 и количество предложений после 3	Корректно.
2.	One3. Two; Three53? Dragon flew away!	One3. Two; Three53? Dragon flew away! Количество предложений до 3 и количество предложений после 3	Корректно
3.	One. 2Two; 3Three; Dragon flew away!	One. 2Two; 3Three; Dragon flew away! Количество предложений до 3 и количество предложений после 3	Корректно
4.	One? Tw2o;	One? Dragon flew away!	Корректно

	Thr45ee. Dragon flew away!	Количество предложений до 3 и количество предложений после 1	

Выводы.

Была написана программа, в ходе написании которой был получен опыт работы с динамическим выделением памяти, символьными массивами и указателями.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

char* input_sentence(){
    int sentence_size = 100, sym_count = 0;
    char sym = '0';
    char* sentence = malloc(sentence_size * sizeof(char));
    while(sym != '.' && sym!= ';' && sym!= '?' && sym!= '!'){
        sym = (char)getchar();
        sentence[sym_count] = sym;
        sym_count += 1;
        if (sym_count == sentence_size){
            sentence_size += 100;
            sentence = realloc(sentence,sentence_size);
        }
    }
    sentence[sym_count] = '\0';
    return sentence;
}

int sentence_selection(char* sentence){
    int i;
```

```
for(i = 0;i < strlen(sentence) - 1;i++){
   if(isdigit(sentence[i])){
      while(isdigit(sentence[i+1]) || isalpha(sentence[i+1]) ){
         if(isalpha(sentence[i+1])){
            while(isdigit(sentence[i+2]) || isalpha(sentence[i+2]) ) {
               if(isdigit(sentence[i+2]))
                  return 0;
               i++;
            }
         }
         i++;
      }
   }
   if(isalpha(sentence[i])){
      while(isdigit(sentence[i]) || isalpha(sentence[i]) ){
         if(isdigit(sentence[i])){
            while(isdigit(sentence[i+1]) || isalpha(sentence[i+1]) ) {
               if(isalpha(sentence[i+1]))
                  return 0;
               i++;
            }

         }
         i++;
      }
   }
}
return 1;
```

```
}

char* no_spaces(char* sentence){
    while(sentence[0] == ' ' || sentence[0] == '\t' || sentence[0] == '\n'){
        int i;
        for (i = 0; i < strlen(sentence) - 1; i++){
            sentence[i] = sentence[i+1];
        }
        sentence[i] = '\0';
    }
    return sentence;
}

int main(){
    int text_size = 10;
    int sent_before_count = 0, sent_after_count = 0;
    char** text = malloc(text_size * sizeof(char*));
    char* sentence;
    while(1){
        sentence = input_sentence();
        sentence = no_spaces(sentence);
        if(sentence_selection(sentence)) {
            text[sent_after_count] = sentence;
            sent_after_count++;
            if (!strcmp(text[sent_after_count - 1],"Dragon flew away!")){
                break;
            }
        }
```

```
            else
                free(sentence);
            sent_before_count += 1;
            if (sent_after_count == text_size ){
                text_size += 10;
                text = realloc(text,text_size * sizeof(char*));
            }
        }
        for(int i = 0; i < sent_after_count ; i++)
            printf("%s\n",text[i]);
          printf("Количество предложений до %d и количество предложений после %d",sent_before_count  ,sent_after_count - 1 );
        for(int i = 0; i < sent_after_count ; i++){
            free(text[i]);
        }
        free(text);
        return 0;
    }
```