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. 9304 _____ **Мохаммед .А**

Преподаватель _____ **Чайка К. В.**

Санкт-Петербург

2019

Цель работы.

Научится как выделит память, как пользоваться динамические массив и еще работать символами и текстом используя стандартные библиотеки языка программирования 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 (\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 (англ. pointer) — переменная, диапазон значений которой

состоит из адресов ячеек памяти или специального значения — нулевого адреса. Последнее используется для указания того, что в данный момент указатель не ссылается ни на одну из допустимых ячеек.

Стандартные функции динамического выделения памяти

Функции динамического выделения памяти находят в оперативной памяти непрерывный участок требуемой длины и возвращают начальный адрес этого участка.

Функции динамического распределения памяти:

```
void* malloc(РазмерМассиваВБайтах);
void* calloc(ЧислоЭлементов, РазмерЭлементаВБайтах);
```

Для использования функций динамического распределения памяти необходимо подключение библиотеки <malloc.h> или <stdlib.h>:

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

или

```
#include <malloc.h>
```

Поскольку обе представленные функции в качестве возвращаемого значения имеют указатель на пустой тип void, требуется явное приведение типа возвращаемого значения.

Для определения размера массива в байтах, используемого в качестве аргумента функции malloc() требуется количество элементов умножить на размер одного элемента. Поскольку элементами массива могут быть как данные простых типов, так и составных типов (например, структуры), для точного определения размера элемента в общем случае рекомендуется использование функции

```
int sizeof(тип);
```

которая определяет количество байт, занимаемое элементом указанного типа.

Память, динамически выделенная с использованием функций calloc(), malloc(), может быть освобождена с использованием функции

free(указатель);

«Правилом хорошего тона» в программировании является освобождение динамически выделенной памяти в случае отсутствия ее дальнейшего использования. Однако если динамически выделенная память не освобождается явным образом, она будет освобождена по завершении выполнения программы.

string.h — заголовочный файл стандартной библиотеки языка Си, содержащий функции для работы с нуль-терминированными строками и различными функциями работы с памятью.

Выполнение работы.

Вес операции исходит внутри функции main() поскольку у нас

программа не большая.

Переменные:

- Text – массив указателей, содержащий все отсортированный предложение.
- Sentence – массив строк, содержащий актуальный предложение.
- Symbol – считываемый символ.
- Sum_of_sentence_befor - количество предложений в изначальном тексте (без учета терминального предложения "Dragon flew away!")
- Sum_of_sentence_after - количество предложений в отформатированном тексте (без учета предложения про количество из данного пункта).

Принцип работы программы:

Используется цикл for() чтобы получить отсортированные предложения.

Внутри цикл инициализируется переменное Sentence – массив строк, содержащий актуальный предложение, которого получается с помощью функцией scanf() до определённого знака. А переменное Symbol поучает тот знак. И знак добавляется переложению с помощью функцией strncat().

Дале проверяется есть ли у предложения слова 555,

если да то:

переходит конца цикла

если нет то:

инициализируется динамический массив который является элементом массива Text, с длиной предложением и копируется в него массив Sentence.

Дале проверяется окажется ли это предложение завершающим

если да то:

завершится цикл

если нет то:

массив Text расширяется с помощью функцией realloc(), на единицу чтобы получит следившие предложение.

А значения Sum_of_sentence_after увеличится на единицу и переходит на концу цикла.

На конце цикла освобождает временно выделенную память с помощью функцией free(), и значения Sum_of_sentence_befor увеличится на единицу.

Дале удаляет табуляцию на начале предложения. И вводит все отсортированные предложения и с помощью функцией puts() и цикл.

Конца работы освобождает всю выделенную память.

Тестирование

Входные данные:

Nulla facilisi. Class aptenT taciti sociosqu ad litora torquent per cOnubia nostra, per inceptos himenaeos. 40 Nu555lla rutrum feugiat felis a pharetra. Sed finibus magna et mauris elementum tempus? Integer at quam et erat iaculis iaculis hendrerit a te4llus? Donec at nunc ac mauris suscipit venenatis. 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. Suspendisse quis mi neque7. 1 Vivamus eu nibh rhoncus, da456pibus ex non, sodales mi. Donec accumsan convallis ipsum vitae lacinia. Donec accumsan convallis ipsum vitae lacinia. Fusce finibus sapien magna, quis scelerisque ex sodales tristique. Nulla facilisi. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Dragon flew away!

Выходные данные:

Nulla facilisi.
Class aptenT taciti sociosqu ad litora torquent per cOnubia nostra, per inceptos

himenaeos. 40 Nu555lla rutrum feugiat felis a pharetra.
Sed finibus magna et mauris elementum tempus? Integer at quam et erat iaculis iaculis hendrerit a te4llus? Donec at nunc ac mauris suscipit venenatis.
Sed finibus magna et mauris elementum tempus? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse quis mi neque7.
1 Vivamus eu nibh rhoncus, da456pibus ex non, sodales mi. Donec accumsan convallis ipsum vitae lacinia.
Donec accumsan convallis ipsum vitae lacinia.
Fusce finibus sapien magna, quis scelerisque ex sodales tristique. Nulla facilisi.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Dragon flew away!
Количество предложений до 16 и количество предложений после 15

Тест пройден.

Выводы

В ходе выполнение лабораторной работы было изучено как выделит память, как пользоваться динамические массив и еще работать символами используя стандартные библиотеки

Приложение А

Исходный код программы

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main()
{
    int Sum_of_sentence_after = 0, Sum_of_sentence_befor = 0;
    char **Text = (char **)calloc(Sum_of_sentence_after, sizeof(char*)),
Symbol = 0;
    for (;;)
    {
        char *Sentence = (char *)malloc(10000*sizeof(char));
        scanf("%[^.;?!]%c", Sentence, &Symbol);
        strncat(Sentence, &Symbol, 1);
        if(!(strstr(Sentence, "555 ") || strstr(Sentence, " 555 ") ||
strstr(Sentence,   "   555.")   ||   strstr(Sentence,   "   555?")    ||
strstr(Sentence, " 555;")))
        {
            Text[Sum_of_sentence_after]                                  =
(char*)malloc((strlen(Sentence)+1)*sizeof(char));
            strcpy(Text[Sum_of_sentence_after], Sentence);
            Sum_of_sentence_after++;
            if (strstr(Sentence, "Dragon flew away!")) break;
            Text                    =                    realloc(Text,
(Sum_of_sentence_after+1)*sizeof(char*));
        }
        free(Sentence);
        Sum_of_sentence_befor++;
    }
    for(int i = 0; i<Sum_of_sentence_after; i++)
        if((Text[i][0]=='\t')||(Text[i][0]==' '))
            for(int    j    =    0;    j    <    strlen(Text[i]);    j++)
Text[i][j]=Text[i][j+1];

    for(int i = 0; i<Sum_of_sentence_after; i++) puts(Text[i]);
    printf("Количество предложений до %d и количество предложений после
%d\n", Sum_of_sentence_befor, Sum_of_sentence_after - 1);
    for(int i = 0; i<Sum_of_sentence_after; i++) free(Text[i]);
return 0;
}
```