**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: Указатели и массивы**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1384 |  | Усачева Д.В. |
| Преподаватель |  | Жангиров Т. Р. |

Санкт-Петербург 2021

**Цель работы.**

Целью работы является освоение использования массивов в языке C, работа с указателями, управлением памяти.

**Задание.**

(Вариант 2)

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).

**\* Порядок предложений не должен меняться**

**\* Статически выделять память под текст нельзя**

**\* Пробел между предложениями является разделителем, а не частью какого-то предложения**

**Выполнение работы.**

Для удобства работы предложение было определено как отдельная

структура Sentence. Она содержит два поля: str (массив символов, предложение), len (текущее количество символов в массиве). Также отдельной структурой Text был определен и текст, содержащий поля: sents (массив предложений, текст), len (количество предложений в изначальном тексте, **без учета** терминального предложения) и real\_len (количество предложений в отформатированном тексте).

Для считывания предложений была реализована функция readSentence(). Она считывает предложения и записывает в массив посимвольно, размера MEMORY, память для которого выделена динамически, пока не встретит символ конца предложения. В случае недостатка выделенной памяти, выделяется новая область памяти. Если введенным символом является «/n», он заменяется на пробел. Возвращаемым значением этой функции является переменная типа Sentence\*.

Функция readText() считывает входящий текст по предложениям, она записывает в динамический массив только те предложения, которые не являются вопросительными, память для массива выделена динамически. А также функция находит количество считанных предложений и количество предложений записанных в массив. Возвращаемым значением этой функции является переменная типа Text\*.

Для удаления пробелов и табуляции до начала предложения была реализована функция del\_tab, которая изменяет исходный текст. Данная функция ничего не возвращает. Она вызывается дважды, для удаления табуляции и удаления пробела, который являлся разделителем предложений.

В теле функции main создается переменная n, хранящая количество предложений в изначальном тексте, и переменная k, хранящая количество предложений в отформатированном тексте. В переменной text хранится значение, возвращаемое функцией readText(). На экран текст выводится при помощи цикла. Каждое предложение выводится с новой строки. С помощью функции free очищается выделенная память.

Разработанный программный код см. в приложении А.

Результаты тестирования см. в приложении Б.

**Выводы.**

Были освоены использование массивов в языке C, работа с указателями

и управление памятью.

**ТЕСТИРОВАНИЕ**

Таблица 2.1 - Пример тестовых случаев

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1. | Donec accumsan convallis ipsum vitae  lacinia. Donec at nunc ac mauris  suscipit venenatis. Etiam quis neque  tincidunt, porta odio vitae, scelerisque  ante? Phasellus nunc augue, dApibus quis  molestie iD, gravida a velit? Fusce  finibus sapien magna, quis scelerisque  ex sodales tristique. Integer at quam  et erat iaculis iaculis hendrerit a  te4llus? Praesent egestas nunc mattis  imperdiet posuere. Nam 7elementum id  enim eu congue; Aliquam at ultricies  nisl, sed pretium nulla; Phasellus  nunc augue, dApibus quis molestie iD,  gravida a velit? Donec congue mauris  sed lacus pulvinar, quis semper orci  sol4licitudin? 40 Nu555lla rutrum  feugiat felis a pharetra. Donec  accumsan convallis ipsum vitae lacinia.  Vivamus sit amet viverra arcu, sed  ultricies nulla. Ut auctor augue vel  tincidunt tincidunt 555. Aenean sem  ligula, laoreet ac sodales a, congue  euismod neque; Aliquam 555 condimentum  ligula arcu, non mollis ex pell555entesque  finibus. Aenean sem ligula, laoreet ac  sodales a, congue euismod neque; Maecenas  555 posuere velit efficitur, egestas nunc  quis, dictum purus? Phasellus nunc augue,  dApibus quis molestie iD, gravida a velit?  Donec congue mauris sed lacus pulvinar,  quis semper orci sol4licitudin? 40 Nu555lla  rutrum feugiat felis a pharetra. Integer at  quam et erat iaculis iaculis hendrerit a  te4llus? Morbi co7ndimentum 555 ex justo,  nec pharetra mauris vestibulum a.  Suspendisse quis mi neque7. Sed finibus  magna et mauris elementum tempus? Sed  finibus magna et mauris elementum tempus?  Class aptenT taciti sociosqu ad litora  torquent per cOnubia nostra, per inceptos  himenaeos. Donec at nunc ac mauris suscipit  venenatis. Nam 7elementum id enim eu congue;  Donec accumsan convallis ipsum vitae lacinia.  Dragon flew away! | Donec accumsan convallis ipsum vitae lacinia.  Donec at nunc ac mauris suscipit venenatis.  Fusce finibus sapien magna, quis scelerisque ex sodales tristique.  Praesent egestas nunc mattis imperdiet posuere.  Nam 7elementum id enim eu congue;  Aliquam at ultricies nisl, sed pretium nulla;  40 Nu555lla rutrum feugiat felis a pharetra.  Donec accumsan convallis ipsum vitae lacinia.  Vivamus sit amet viverra arcu, sed ultricies nulla.  Ut auctor augue vel tincidunt tincidunt 555.  Aenean sem ligula, laoreet ac sodales a, congue euismod neque;  Aliquam 555 condimentum ligula arcu, non mollis ex pell555entesque finibus.  Aenean sem ligula, laoreet ac sodales a, congue euismod neque;  40 Nu555lla rutrum feugiat felis a pharetra.  Morbi co7ndimentum 555 ex justo, nec pharetra mauris vestibulum a.  Suspendisse quis mi neque7.  Class aptenT taciti sociosqu ad litora torquent per cOnubia nostra, per inceptos himenaeos.  Donec at nunc ac mauris suscipit venenatis.  Nam 7elementum id enim eu congue;  Donec accumsan convallis ipsum vitae lacinia.  Dragon flew away!  Количество предложений до 31 и количество предложений после 20 | Реализованная программа корректно работает: обрабатывает введенный текст и выводит количество предложений до обработка и после (без учета терминального). |

**ПРИЛОЖЕНИЕ 1**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: main.c

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

#define MEMORY 200

typedef struct Sentence {

char \*str;

int len;

} Sentence;

typedef struct Text {

struct Sentence \*\*sents;

int size;

int len;

int real\_len;

} Text;

Sentence \*readSentence() {

int size = MEMORY;

char \*buf = malloc(size \* sizeof(char));

char temp;

int n = 0;

do {

if (n >= size - 2) {

char \*t = realloc(buf, size \* 2);

if (t) {

size \*= 2;

buf = t;

}

}

temp = getchar();

if (temp == '\n') {

temp = ' ';

}

buf[n] = temp;

n++;

} while (temp != ';' && temp != '.' && temp != '?' && temp != '!');

buf[n] = '\0';

Sentence \*sentence = malloc(sizeof(Sentence));

sentence->str = buf;

sentence->len = n;

return sentence;

}

Text \*readText() {

int size = MEMORY;

Sentence \*\*text = malloc(size \* sizeof(Sentence \*));

char end = '\0';

int n = 0;

int k = 0;

Sentence \*temp;

do {

if (n >= size - 2) {

Sentence \*\*t = realloc(text, size \* 2);

if (t) {

text = t;

size \*= 2;

}

}

temp = readSentence();

int end\_s = strlen(temp->str);

if (temp->str[end\_s - 1] != '?') {

text[k] = temp;

k++;

}

n++;

} while ((temp->str[(temp->len) - 1]) != '!');

Text \*txt = malloc(sizeof(Text));

txt->real\_len = k;

txt->len = n;

txt->sents = text;

txt->size = size;

return txt;

}

void del\_tab(Text txt, int n) {

for (int i = 0; i < n; i++) {

if (((((txt.sents)[i]->str)[0]) == '\t') || ((((txt.sents)[i]->str)[0]) == ' ')) {

char \*old\_in = &txt.sents[i]->str[0];

char \*new\_in = &(txt.sents[i]->str[0]) + 1;

int l = strlen(txt.sents[i]->str);

memmove(old\_in, new\_in, l);

}

}

}

int main() {

Text \*text = readText();

Text t = \*text;

int size = t.size;

int n = text->len;

int k = t.real\_len;

del\_tab(t, k);

del\_tab(t, k);

for (int i = 0; i < k; i++)

printf("%s\n", t.sents[i]->str);

printf("Количество предложений до %d и количество предложений после %d\n", n - 1, k - 1);

free(text);

return 0;

}