МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №3 по

дисциплине «Программирование»

ТЕМА: Использование указателей

Студент гр. 6304 Юсковец А.В.

Преподаватель Берленко Т.А.

Санкт-Петербург

2017

Оглавление

Цель работы

Написать 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 цифра 7, 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).

Ход работы

- **Код программы**

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

#define FACTOR 1.62
#define END "Dragon flew away!"
#define FIRST_ALLOC_LEN 50

int is_end(char const* text, char const* end);

int buffer_realloc(char* text, int text_len);

int main() {
    int before_edit = 0;
    int after_edit = 0;
    char* text = (char*)calloc(FIRST_ALLOC_LEN, sizeof(char));
    int text_len = FIRST_ALLOC_LEN;
    char* end = END;
    int end_len = strlen(end);

    int sentence_begin = 0;
    int text_index = 0;
    char c = 0;
    while (c = getchar()) {
        if (text_index == sentence_begin && (c == '\t' || c == '\n' || c == ' ')) { continue; }

        if (c != '.' && c != ';' && c != '?' && c != '7') {
            if (text_index == text_len) { text_len = buffer_realloc(text, text_len); }

            text[text_index++] = c;

            if (c == end[end_len - 1] && is_end(text, end)) {
                printf("%s\n", text);
                break;
            }
        } else if (c == '7') {
            while ((c = getchar()) != '.' && c != ';' && c != '?');
            ++before_edit;

            text_index = sentence_begin;
        } else {
            ++after_edit;
            ++before_edit;
```

```
            if (text_index == text_len - 1) { text_len = buffer_realloc(text, text_len); }
            text[text_index++] = c;
            text[text_index++] = '\n';

            sentence_begin = text_index;
        }
    }

    printf("Количество предложений до %d количество предложений после %d", before_edit,
after_edit);

    return 0;
}

int is_end(char const* text, char const* end) {
    int text_size = strlen(text);
    int end_size = strlen(end);

    int j = end_size - 1;
    int i = 0;
    for (i = text_size - 1; i != text_size - end_size - 1; --i, --j) {
        if (text[i] != end[j]) { return 0; }
    }

    if (text[i - 1] != '.' && text[i - 1] != ';' && text[i - 1] != '?') { return 0; }

    return 1;
}

int buffer_realloc(char* text, int text_len) {
    text_len = text_len * FACTOR;
    text = realloc(text, text_len);
    text[(int)(text_len - 1)] = '\0';
    return text_len;
}
```

- **Директивы препроцессора**

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

#define FACTOR 1.62
#define END "Dragon flew away!"
#define FIRST_ALLOC_LEN 50
```

Подключаются соответствующие библиотеки для использования функций: **getchar**(), **calloc**(), **realloc**(), **strlen**().

Определены **FACTOR** — множитель для реаллокации, **END** — терминальное предложение, **FIRST_ALLOC_LEN** - начальный размер буфера.

- **Объявление вспомогательных функций**

```
int is_end(char const* text, char const* end);

int buffer_realloc(char* text, int text_len);
```

is_end – проверяет наличие терминальной последовательности в буффере.
buffer_realloc – реаллоцирует буффер.

- **Инициализация переменных**

```
int before_edit = 0;
int after_edit = 0;
char* text = (char*)calloc(FIRST_ALLOC_LEN, sizeof(char));
int text_len = FIRST_ALLOC_LEN;
char* end = END;
int end_size = strlen(end);
int sentence_begin = 0;
int text_index = 0;
char c = 0;
```

before_edit — количество предложений до редактирования;
after_edit — количество предложений после редактирования;
text — буффер, заполняется '\0'.
text_len — длина буфера
end — терминальное предложение

end_size — длина терминального предложения

sentence_begin — индекс начала предложения

text_index — текущий индекс в буффере

c — для хранения текущего символа

- **Посимвольное считывание**

```
if (text_index == sentence_begin && (c == '\t' || c == '\n' || c == ' ')) { continue; }
      if (c != '.' && c != ';' && c != '?' && c != '7') {
         if (text_index == text_len) { text_len = buffer_realloc(text, text_len); }

         text[text_index++] = c;

         if (c == end[end_size - 1] && is_end(text, end)) {
            printf("%s\n", text);
            break;
         }

      }
```

Первое условие игнорирует некоторые символы в начала предложения (text_index == sentence_begin).

Второе условие проверяет, есть ли место в буфере.

Третье условие проверяет, было ли передано терминальное предложение, в случае, если было, выводится содержимое буфера и цикл прекращается.

- **Удаления предложения с 7**

```
else if (c == '7') {
   while ((c = getchar()) != '.' && c != ';' && c != '?');
   ++before_edit;

   text_index = sentence_begin;

}
```

Игнорируются все символы до тех пор, пока не встретится символ конца предложения.

Увеличивается количество предложений до редактирования на еденицу.

Передвигается индекс буффера на начало предложения и все символы, считанные ранее в предложении с семеркой буду перезаписаны на последующих итерациях.

- **Конец предложения**

```
else {
   ++after_edit;
   ++before_edit;

   if (text_index == text_len - 1) { text_len = buffer_realloc(text, text_len); }

   text[text_index++] = c;
   text[text_index++] = '\n';

   sentence_begin = text_index;
}
```

Увеличиваются счетчики количества предложений до и после обработки на один.

В условии проверяется хватает ли места в буффере для записи символа конца предложения и переноса строки.

Индекс начала предложения передвигается на текущий индекс буфера.

- **is_end**

```
int text_size = strlen(text);
int end_size = strlen(end);

int j = end_size - 1;
int i = 0;
for (i = text_size - 1; i != text_size - end_size - 1; --i, --j) {
   if (text[i] != end[j]) { return 0; }
}

if (text[i - 1] != '.' && text[i - 1] != ';' && text[i - 1] != '?') { return 0; }

return 1;
```

text_size — длина буфера.

end_size — длина терминального предложения.

j — индекс последнего элемента терминального предложения.

i — индекс последнего элемента буфера.

Цикл пробегается по терминальному предложению и по буферу с конца до начала терминального предложения и в случае не равенства одного из символов функция возвращает 0.

Следующее условие проверят, что перед терминальной последовательностью нет постфиксов, в случае если есть, функция возвращает 0.

Иначе, функция возвращает 1.

- **buffer_realloc**

```
text_len = text_len * FACTOR;
text = realloc(text, text_len);
text[(int)(text_len - 1)] = '\0';
return text_len;
```

Длина буфера увеличивается на FACTOR.

Последний элемента буфера инициализируется '\0'.

Функция возвращает измененную длину буфера.

Вывод

Была создана програма, реализующая требуемый функционал с использованием указателей и динамической памяти. При компилировании, сборке и выполнении исполняемого файла не возникает ошибок и предупреждений.