**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: *Динамические структуры данных*

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Фокин К.С. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**Цель работы:** требуется написать программу, моделирующую работу стека для определения валидности кода html-страницы.

**Ход работы:**

В ветке **fokin\_sem2\_lr2** репозитория **pr-2016-6303** создал папку **fokin\_sem2\_lr2** с проектом, состоящим из 1 файла ***main\_lr2.c***

Формулировка задания:

*Требуется написать программу, получающую на вход строку, (без кириллических символов и не более 500 символов) представляющую собой код "простой" html-страницы и проверяющую ее на валидность. Программа должна вывести****correct****если страница валидна или****wrong****.  
  
html-страница, состоит из тегов и их содержимого, заключенного в эти теги. Теги представляют собой некоторые ключевые слова, заданные в треугольных скобках. Например, <****tag****> (где tag - имя тега). Область действия данного тега распространяется до соответствующего закрывающего тега </****tag****> который отличается символом /. Теги могут иметь вложенный характер, но не могут пересекаться   
<tag1><tag2></tag2></tag1> - верно  
<tag1><tag2></tag1></tag2> - не верно  
  
Существуют теги, не требующие закрывающего тега.   
  
Валидной является html-страница, в коде которой всякому открывающему тегу соответствует закрывающий (за исключением тегов, которым закрывающий тег не требуется)  
  
Во входной строке могут встречаться любые парные теги, но гарантируется, что в тексте, кроме обозначения тегов, символы < и > не встречаются. аттрибутов у тегов также нет.   
Теги, которые не требуют закрывающего тега: <br>, <hr>  
  
Стек (который потребуется для алгоритма проверки парности тегов) требуется реализовать самостоятельно на базе****списка****.*

Ниже приведен код программы:

/\*

Программа main\_lr2.c (Л/р №2) определяет валидность кода html-страницы.

Необходима проверка на наличие закрывающей скобки (">") в теге

(в условии не сказано, но иначе "Failed test #22. Runtime error:").

By foksen98 02.04.2017.

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

// максимальная длина входной строки

#define LENGTH 500

// структура элемента списка

typedef struct bracket

{

char\* tag;

struct bracket\* prev;

} bracket;

// функция освобождает динамическую память (весь список)

void delete (bracket\* current)

{

bracket\* temp;

while (current->prev)

{

temp = current->prev;

free(current);

current = temp;

}

free(current);

}

// функция добавляет в список структуру с именем последнего открывающего

// (но требующего закрытия) тега

bracket\* add (char\* tag, bracket\* current)

{

// если тег не требует закрытия, возвращаем указатель на текущий

// (последний) элемент списка

if (!strcmp(tag, "br") || !strcmp(tag, "hr"))

return current;

// иначе создаем новый элемент списка с именем последнего открывающего

// тега и возвращаем указатель на него

else

{

bracket\* next = (bracket\*)malloc(sizeof(bracket));

next->tag = tag;

next->prev = current;

return next;

}

}

// функция проверяет, закрывает ли этот тег последний отрывающий тег из списка

bracket\* check (char\* tag, bracket\* current)

{

// если имя закрывающего тега соответствует имени последнего

// открывающего тега, то удаляем последний элеменент списка и

// возвращаем указатель на предпоследний

if (!strcmp(tag, current->tag))

{

bracket\* temp = current->prev;

free(current);

return temp;

}

// иначе освобождаем память всего списка и возвращаем NULL

else

{

delete(current);

return NULL;

}

}

int main()

{

char str[LENGTH + 2];

char\* ptr;

char\* temp;

bracket\* current = (bracket\*)malloc(sizeof(bracket));

current->tag = "null";

current->prev = NULL;

fgets(str, (LENGTH + 2), stdin);

ptr = strstr(str, "<");

// проверка на наличие закрывающей скобки для открывающей

// (в данном случае строка "<tag1" считается валидной)

if (!strstr(str, ">"))

ptr = NULL;

while (ptr)

{

// если тег закрывающий, то вызываем функцию check() для проверки

if (\*(ptr + 1) == '/')

{

temp = (strstr(ptr, ">") + 1);

current = check((strtok(ptr, ">") + 2), current);

// если функция check() вернула NULL, выводим "wrong"

// и завершаем программу

if (!current)

{

printf("wrong\n");

return 0;

}

}

// иначе вызываем функцию add() для добавления нового открывающего

// тега в список

else

{

temp = (strstr(ptr, ">") + 1);

current = add((strtok(ptr, ">") + 1), current);

}

ptr = strstr(temp, "<");

// проверка на наличие закрывающей скобки для открывающей

// (строка "<tag1><tag2></tag2></tag1" считается невалидной)

if (!strstr(temp, ">"))

ptr = NULL;

}

// если были закрыты все открывающие теги (осталась только голова списка),

// то выводим "correct"

if (!current->prev)

printf("correct\n");

// иначе "wrong"

else

printf("wrong\n");

// очищаем оставшуюся динамическую память

delete(current);

return 0;

}

1. Добавление в список структуры с именем последнего открывающего (но требующего закрытия) тега реализовано при помощи функции **add()**.
2. Проверка на то, закрывает ли этот тег последний отрывающий тег из списка выполняется функцией **check()**.
3. Функция **delete()** отвечает за освобождение динамической памяти, выделенной под элементы списка.

Созданная папка с файлом загружена в репозиторий на **Github** с помощью следующих команд:

* git add fokin\_sem2\_lr2
* git commit –m “Fokin LR №2 done”
* git push origin fokin\_sem2\_lr2

**Вывод:** в ходе лабораторной работы изучены возможности применения динамических структур в языке Си. Полученные знания закреплены в ходе разработки программы **main\_lr2.c** для определения валидности кода html-страницы.