**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: **Динамические структуры данных. Тестирование.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Тимофеев А.А. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

Оглавление

[Оглавление 2](#__RefHeading___Toc6485_1258972700)

[Цель 3](#__RefHeading___Toc6491_1258972700)

[Задание 3](#__RefHeading___Toc6493_1258972700)

[Содержание 3](#__RefHeading___Toc6495_1258972700)

[Работа с репозиторием 5](#__RefHeading___Toc6487_1258972700)

[Вывод 7](#__RefHeading___Toc6489_1258972700)

# **Цель**

Написание программы с использованием динамических структур данных.

# **Задание**

Расстановка тегов.

Требуется написать программу, получающую на вход строку, (без кириллических символов и не более 3000 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.

# **Содержание**

В ходе выполнения задачи, поставленной в данной лабораторной работе, мною были использованы следующие функции:

1. Функция добавления элемента в стек.

stack \*push(stack \*head,char\* elem) {

stack\* tmp = (stack\*)[malloc](http://www.opengroup.org/onlinepubs/009695399/functions/malloc.html)(sizeof(stack));

[strcpy](http://www.opengroup.org/onlinepubs/009695399/functions/strcpy.html)(tmp->value, elem);

if(head!=NULL) tmp->prev = head;

else tmp->prev = NULL;

return tmp;

}

2. Функция удаления элемента из стека

stack \*pop(stack \*head) {

stack\* tmp = head->prev;

[free](http://www.opengroup.org/onlinepubs/009695399/functions/free.html)(head);

return tmp;

}

3. Функция освобождения памяти, выделенной под стек

void freeStack(stack \*head){

while (head->prev) {

stack\* tmp=head->prev;

[free](http://www.opengroup.org/onlinepubs/009695399/functions/free.html)(head);

head = tmp;

}

}

4. Функция проверки тега на исключение.(Нужен ли тегу закрывающий, или нет).

int checkingException(char\* str, int i){

char\* exception=(char\*)[malloc](http://www.opengroup.org/onlinepubs/009695399/functions/malloc.html)(sizeof(char)\*5);

int j=0;

for(j=0;j<4;j++)

{ \*(exception+j)=\*(str+i);

i++;

}

\*(exception+j)='**\0**';

if (([strcmp](http://www.opengroup.org/onlinepubs/009695399/functions/strcmp.html)(exception,EXCEPTIONAL\_TAG\_1))==0 || [strcmp](http://www.opengroup.org/onlinepubs/009695399/functions/strcmp.html)(exception,EXCEPTIONAL\_TAG\_2)==0) {

[free](http://www.opengroup.org/onlinepubs/009695399/functions/free.html)(exception);

return 1;

}

return 0;

}

В которой, теги исключения сделаны константами

#define EXCEPTIONAL\_TAG\_1 "<br>"

#define EXCEPTIONAL\_TAG\_2 "<hr>"

5. Функция, создающая тег

char\* creatingTag(int i,char\* str, char\* strTag){

int j=0;

while(\*(str+i)!='>'){

if(\*(str+i)=='/') {

i++;

continue;

}

{

\*(strTag+j)=\*(str+i);

j++;

i++;

}

}

\*(strTag+j)='**\0**';

return strTag;

}

6. Функция, проверяющая правильность расстановки тегов

void checkingStr(char\* str){

char\* strTag=(char\*)[malloc](http://www.opengroup.org/onlinepubs/009695399/functions/malloc.html)(sizeof(char)\*15);

int i=0;

int len;

len=[strlen](http://www.opengroup.org/onlinepubs/009695399/functions/strlen.html)(str);

char tag[15];

while(i<len){

if(\*(str+i)=='<'){

if(checkingException(str,i)==1) {

i+=4;

continue;

}

else if(\*(str+i+1)!='/') {

head=push(head, creatingTag(i,str,strTag));

i+=[strlen](http://www.opengroup.org/onlinepubs/009695399/functions/strlen.html)(head->value);

}

else {

if([strcmp](http://www.opengroup.org/onlinepubs/009695399/functions/strcmp.html)(head->value,creatingTag(i,str,strTag))==0) {

i+=[strlen](http://www.opengroup.org/onlinepubs/009695399/functions/strlen.html)(head->value); head=pop(head);

}

else {

[printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("wrong"); {

freeStack(head);

}

return;

}

}

}

else i++;

}

if (head==NULL) [printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("correct**\n**");

else {

[printf](http://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("wrong**\n**");

freeStack(head);

}

[free](http://www.opengroup.org/onlinepubs/009695399/functions/free.html)(strTag);

return;

}

7. Функция main, в которой мы считывали строку при помощи функции fgets и передавали введенную строку в функцию checkingStr

int main() {

char\* str=(char\*)[malloc](http://www.opengroup.org/onlinepubs/009695399/functions/malloc.html)(3000\*sizeof(char));

int i=0,k=0;

[fgets](http://www.opengroup.org/onlinepubs/009695399/functions/fgets.html)(str,3000,stdin);

(\*[strstr](http://www.opengroup.org/onlinepubs/009695399/functions/strstr.html)(str,"**\n**"))= 0;

checkingStr(str);

[free](http://www.opengroup.org/onlinepubs/009695399/functions/free.html)(str);

return 0;

}

# **Работа с репозиторием**

1. Копируем репозиторий группы 6304

*Git clone https://github.com/moevm/pr1-2016-6304.git*

1. От ветки master создаем локальную ветку с названием работы и именем:

*Git checkout –b “Timofeev\_kursework”*

1. В папку с клонированным репозиторием скопируем файлы с работой с помощью команды *cp –r*
2. Добавляем папку с файлом на ветку командой

*git add Kurswork*

1. Коммитим изменения:

*Git commit –m “Залил курсовую работу”*

1. Отправляем изменения в репозиторий командой

*Git push origin*

1. На сайте github создаем *pull request* в ветку *master.*

# Вывод

В ходе выполнения данной работы было закреплено написание программы, моделирующей работу стека, реализованного на базе списка.