**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 |  | Архипцев Евгений |
| Преподаватель |  | Чайка Константин |

**Цель работы:**  
 Требуется написать программу, получающую на вход строку, (без кириллических символов и не более 3000 символов) представляющую собой код "простой" [html](https://ru.wikipedia.org/wiki/HTML)-страницы и проверяющую ее на валидность. Программа должна вывести **correct** если страница валидна или **wrong**.

**Ход выполнения:**

**Flag** переменная, показывающая полноту стэка (Количество элементов)

**1.** Опишем функцию **push**  
Функция push() ставит значение на вершину стека.

Функция получает указатель на стэк, и элемент, который нужно поместить на вершину стэка.

void push (char \*\* stack, char \* element)

flag++;

strcpy(stack[flag],element);

}  
  
**2.** Опишем функцию **pop**  
Функция pop() удаляет значение с вершины, то есть удаляет последний элемент.

void pop(char \*\* stack){

flag--;

}

**3.** Опишем функцию **size\_s**  
Функция **size\_s**() возвращает количество элементов в стеке.

int size\_s(){

return flag;

}

**4.** Опишем функцию **empty\_s**  
Функция **empty\_s()** дает понять пуст стек или нет .

Возвращает 0, если стэк пуст

Возвращает 1, если стэк не пуст

int empty\_s(){

if (flag == -1){

return 0;

}

else{

return 1;

}

}

**5**. Считываем Тэги  
  
 **е**сли он открывающийся – помещаем в стэк  
 **е**сли он закрывающийся – проверяем, что он совпадает с предыдущим

**6**. Если стэк пуст по окончанию работы- выводим “**correct**”, иначе “**wrong**”.

**Исходный код программы:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int flag = -1;

void push(char \*\* stack, char \* element){ // Помещаем в СТЭК

flag++;

strcpy(stack[flag],element);

}

void pop(char \*\* stack){ // Удаляем последний элемент СТЭКа

flag--;

}

int size\_s(){ // Возвращает размер СТЭКа

return flag;

}

int empty\_s(){ // Возвращает 0 ,если стэк пуст

if (flag == -1){//Возвращает 1 ,если стэк не пуст

return 0;

}

else{

return 1;

}

}

int main()

{

char c,tag[80];

int i;

/\*Выделение памяти под стэк\*/

char \*\*stack;

stack = (char\*\*)malloc(100 \* sizeof(char\*));

for (i = 0; i<100; i++){

stack[i] = (char\*)malloc(80 \* sizeof(char));

}

/\*\*/

while ((c = getchar()) != EOF ){

if (c == '<') {

scanf("%79[^>]>",tag); //Считываем ИМЯ тэга

if (strcmp(tag,"hr") && strcmp(tag,"br")){

if (tag[0] != '/'){ // Если тэг - открывающийся : заносим в стэк

push(stack,tag);

}

else{

if (empty\_s() == 0){ // Если стэк пуст, то есть ЗАКРВАЮЩИЙСЯ тэг не был открыт, то выход

printf("wrong");

return 0;

}

else{

if (strcmp(stack[flag],tag+1) != 0){ // Если закрывающийся, не равен перед ним стоящему открывающемуся

printf("wrong");

return 0;

}

else{ // Если тэги совпадают

pop(stack);

}

}

}

}

}

}

if (empty\_s() == 0){ // Если стэк пуст, то страницы ВАЛИДНА

printf("correct");

}

else{

printf("wrong");

}

}

**Вывод:** Были усовершенствуемы знания в использовании динамических массивов. Ознакомились с таким абстрактным типом данным как Стек.

Изучены функции Pop,Push,Size,Empty для работы со Стеком.