## МИНОБРНАУКИ РОССИИ

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МОЭВМ**

## ОТЧЕТ **по лабораторной работе №3 по дисциплине «Алгоритмы и структуры данных»** ТЕМА: Стеки и очереди

|  |  |  |
| --- | --- | --- |
| Студент гр. 7303 |  | Юсковец А.В. |
| Преподаватель |  | Балтрашевич Т.А. |

Санкт-Петербург  
2018

**Оглавление**

[Цель работы 3](#__RefHeading___Toc1709_692322312)

[Задание 3](#__RefHeading___Toc8601_2844259154)

[Ход работы 3](#__RefHeading___Toc1711_692322312)

[Вывод 3](#__RefHeading___Toc1731_692322312)

[Приложение А. Стек 4](#__RefHeading___Toc8079_1876010945)

[Приложение Б. Функция 5](#__RefHeading___Toc7979_1876010945)

# Цель работы

Реализовать структуру данных: очередь или стек. Решить поставленное задание с использованием своей реализации нужной структуры данных.

# Задание

Правильная скобочная конструкция с тремя видами скобок определяется как

<текст> ::= <пусто> | <элемент> <текст>

<элемент> ::= <символ> | (<текст>) | [<текст>] | {<текст>},

где <символ> - любой символ, кроме (, ), [, ], {, }. Проверить, является ли текст правильной скобочной конструкцией; если нет, то указать номер ошибочной позиции.

# Ход работы

Для решения поставленной задачи используется стек, реализация которого приведена в приложении А.

Реализация требуемой функции приведена в приложении Б:

Функция check\_braces принимает строку (скобочную последовательность), которую нужно проверить на корректность. В случае ошибки возвращается порядковый номер скобки (начиная с 1), для которой не нашлось закрывающей или номер закрывающей скобки, не соответствующей открывающей.

На каждой итерации цикла функция считывает один символ:  
- если символ не скобка, то цикл переходит на следующую итерацию.   
- если символ — открывающая скобка, то в стек помещается пара (скобка, порядковый номер)  
- если символ — закрывающая скобка, то со стека выталкивается последняя скобка

# Вывод

Была написана программа проверяющая корректность скобочной последовательности. При написании программы были использованы все функции для работы со стеком. При компиляции и выполнении программы не возникает ошибок и предупреждений.

# Приложение А. Стек

#ifndef STACK\_H

#define STACK\_H

#include <vector>

#include <iostream>

#include <string>

template <typename T>

class **Stack** {

public:

void **push**(T i) {

\_data.push\_back(i);

}

void **pop**() {

if (!\_data.empty())

\_data.pop\_back();

}

T **top**() {

return \_data.back();

}

bool **empty**() {

return \_data.empty();

}

private:

std::vector<T> \_data;

};

#endif // STACK\_H

# Приложение Б. Функция

size\_t **check\_braces**(string input) {

Stack<pair<char, size\_t>> stack;

size\_t len = input.length();

for (size\_t i = 0; i != len; ++i) {

switch (input[i]) {

case '[':

case '{':

case '(':

stack.push(make\_pair(*input[i]*, i+1));

break;

case ']':

if (!stack.empty() && stack.top().first == '[')

stack.pop();

else

return i + 1;

break;

case '}':

if (!stack.empty() && stack.top().first == '{')

stack.pop();

else

return i + 1;

break;

case ')':

if (!stack.empty() && stack.top().first == '(')

stack.pop();

else

return i + 1;

break;

}

}

if (stack.empty())

return 0;

else

return stack.top().second;

}

# Приложение В. Интерфейс



