МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Алгоритмы и структуры данных»

Тема: Линейные структуры данных

Студент гр. 9384 _____ Соседков К.С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Получить практические навыки работы с линейными структурами данных.

Задание. (Вариант № 5)

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, содержащийся в заданном файле F, правильной скобочной конструкцией; если нет, то указать номер ошибочной позиции.

Выполнение работы.

Для выполнения работы был разработан класс *Stack* со следующими методами:

push — добавление элемента в стек

pop — вернуть последний добавленный элемент и удалить его из стека.

top — вернуть последний добавленный элемент

isEmpty — возвращает *true* если стек не пустой, иначе *false*

size — возвращает колличество элементов в стеке

Алгоритм выполнения программы:

Из введенной пользователем строки(или строки из файла) считываются символы.

Если текущий символ равен открывающейся скобке, то он кладется в стек.

Если скобка закрывающаяся и последний добавленный элемент в стек является парной скобкой, элемент удаляется из стека.

Если скобка не является парной или стек оказывается пустым, выдается ошибка и программа завершает свою работу.

Если после обхода всей строки стек оказывается пустым, то введенная строка является правильной скобочной последовательностью, иначе нет.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.		String is empty Error 1	Ошибка Ввод пустой строки
2.	([{({([a])})(b)}][{c}{}])	OK	Все в порядке
3.	)	ERROR at pos 0 (Stack is empty) Error 2	Ошибка Нет парной скобки „(“
4.	({()}{})[	ERROR: Stack is not empty Error 4	Ошибка Лишняя скобка на последней позиции
5.	(){}(((){}()	Unexpected character } at pos 7 Error 3	Ошибка Много лишних скобок

Выводы.

В ходе выполнения лабораторной работы была реализована линейная структура данных стек, и методы для работы с ней.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include "Stack.h"
#include <iostream>

Stack::Stack() {}

Stack::~Stack() {
    if(head) {
        Node* tmp = head;
        while (tmp){
            delete tmp;
            tmp = tmp->next;
        }
    }
}

void Stack::push(char value) {
    length++;
    Node* newNode = new Node;
    newNode->data = value;
    if(!head) {
        head = newNode;
        tail = head;

    }
    else {
        tail->next = newNode;
        newNode->prev = tail;
        tail = newNode;
    }

}
```

```
char Stack::pop() {
   char value = tail->data;
   if(this->isEmpty()) {
     delete this;
     throw std::runtime_error("Stack::pop() stack is empty");
   }
   if(this->size() == 1) {
      delete tail;
      length--;
      return value;
   }

   length--;
   tail->prev->next = nullptr;
   Node* tmp = tail->prev;
   delete tail;
   tail = tmp;
   return value;
}

bool Stack::isEmpty() {
   return length == 0;
}

int Stack::size() {
   return length;
}

char Stack::top() {
   return tail->data;
}

void Stack::print() {
```

```
    Node* tmp = tail;
    std::cout << "Stack--------------------\n";
    while(tmp) {
        std::cout << tmp->data << "\n";
        tmp = tmp->prev;
    }
    std::cout << "-------------------------\n";
}
```

Название файла: Stack.cpp

```
#include "Stack.h"
#include <iostream>

Stack::Stack() {}

Stack::~Stack() {
    if(head) {
        Node* tmp = head;
        while (tmp){
            delete tmp;
            tmp = tmp->next;
        }
    }
}

void Stack::push(char value) {
    length++;
    Node* newNode = new Node;
    newNode->data = value;
    if(!head) {
        head = newNode;
        tail = head;

    }
```

```
    else {
        tail->next = newNode;
        newNode->prev = tail;
        tail = newNode;
    }

}

char Stack::pop() {
    char value = tail->data;
    if(this->isEmpty()) {
      delete this;
      throw std::runtime_error("Stack::pop() stack is empty");
    }
    if(this->size() == 1) {
        delete tail;
        length--;
        return value;
    }

    length--;
    tail->prev->next = nullptr;
    Node* tmp = tail->prev;
    delete tail;
    tail = tmp;
    return value;
}

bool Stack::isEmpty() {
    return length == 0;
}

int Stack::size() {
    return length;
```

```
}

char Stack::top() {
    return tail->data;
}

void Stack::print() {
    Node* tmp = tail;
    std::cout << "Stack--------------------\n";
    while(tmp) {
        std::cout << tmp->data << "\n";
        tmp = tmp->prev;
    }
    std::cout << "-------------------------\n";
}
```

Название файла: Stack.h

```
#pragma once

class Stack
{
private:
    struct Node {
        Node* next = nullptr;
        Node* prev = nullptr;;
        char data;
    };

    Node* head = nullptr;
    Node* tail = nullptr;
    int length = 0;
public:
    Stack();
    ~Stack();
```

```
    void push(char value);
    char pop();
    bool isEmpty();
    int size();
    char top();
    void print();
};
```