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

ОТЧЕТ

по лабораторной работе №3
по дисциплине «Алгоритмы и структуры данных»
Тема: Стеки и очереди

Студент гр. 8382 _____ Нечепуренко Н.А.

Преподаватель _____ Балтрашевич В.Э.

Санкт-Петербург
2019

Цель работы.

Получить навыки работы со стеками и очередями. Написать программу, находящую пары индексов ближайших открывающихся и закрывающихся скобок.

Постановка задачи.

В заданном текстовом файле F записан текст, сбалансированный по круглым скобкам:

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

<элемент> ::= <символ> | (<текст>)

где <символ> - любой символ. Для каждой пары соответствующих открывающей и закрывающей скобок вывести номера их позиций в тексте, упорядочив пары в порядке возрастания номеров позиций закрывающих скобок.

Например, для текста A + (45 - F (X) * (B - C)) надо напечатать: 8 10; 12 16; 3 17.

В решении задачи использовать очередь и/или стек.

Выполнение работы.

Для решения задачи из условия можно использовать только стек, так как очередная пара добавляется, если встречается закрывающая скобка, что гарантирует возрастание индексов таких скобок.

Опишем шаблонный класс стека на базе массива в файле stack.cpp (см. Приложение А). Реализуем необходимый функционал – функции push, pop, top, empty, size.

Основной алгоритм описан в функции processing файла processing.cpp. Скриншот кода представлен на рисунке ниже (см. рис. 1).

```cpp
#include <iostream>
#include <string>
#include "stack.cpp"
using namespace std;
string processing(string s){
    Stack<int> ind_stack;
    string res = "";
    for (int i = 0; i < (int) s.length(); i++){
        if (s[i] == '(')
            ind_stack.push(i+1);
        if (!ind_stack.empty() && s[i] == ')'){
            int open_index = ind_stack.top(); ind_stack.pop();
            res += to_string(open_index)+" "+to_string(i+1)+"; ";
        }
    }
    return res;
}
```

Рисунок 1 – Файл processing.cpp

Создается стек, хранящий индексы открывающих скобок. Согласно заданию, скобки сбалансированы. Чтобы обеспечить единичную индексацию прибавляем единицу и формируем строку-ответ.

Пример работы программы представлен на рисунке ниже (см. рис. 2).

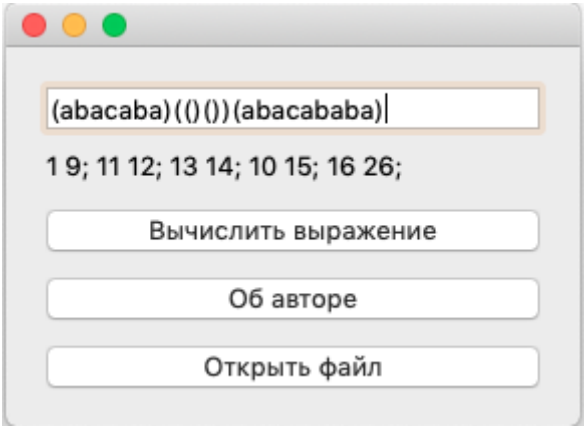

Рисунок 2 – Пример работы программы

Проведем тестирование программы. Результаты сведем в табл. 1.

Таблица 1 – Тестирование программы

Входные данные	Выходные данные
(f(x)+12*3)/(g(x)-1)	3 5; 1 15; 19 21; 17 24;
((((()))))	4 5; 3 6; 2 7; 1 8;
sin(x) = x - x^3/6 + o(x^3)	4 6; 23 27;

Выводы.

В ходе выполнения лабораторной работы были получены навыки работы со структурой данных стек. Был реализован стек с базовым интерфейсом, способный хранить в себе данные произвольного типа. Алгоритм нахождения индексов парных скобок успешно прошел все тесты. Для программы был реализован графический интерфейс (см. рис. 2).

ПРИЛОЖЕНИЕ А.
ФАЙЛ STACK.CPP

```
#include <cassert>
#include <algorithm>
template<class T>
class Stack {
    public:
        Stack(int size = 0){
            assert(size >= 0);
            _size = size;
            _capacity = (size+1) * 2;
            _stack_handler = new T[_capacity];
        }
        T top(){
            assert(_size >= 1);
            return _stack_handler[_size - 1];
        }

        void pop(){
            assert(_size >= 1);
            _size--;
        }

        void push(T param){
            if (_size + 1 == _capacity){
                _capacity *= 2;
                T* new_array = new T[_capacity];
                std::copy(_stack_handler,_stack_handler+_size, new_array);
                delete [] _stack_handler;
                _stack_handler = new_array;
            }
            _stack_handler[_size++] = param;
        }

        bool empty(){
            return _size ? false : true;
        }

        size_t size(){
            return _size;
        }
```

```
        ~Stack(){
            delete [] _stack_handler;
        }

    private:
        T* _stack_handler;
        size_t _size;
        size_t _capacity;
};
```