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

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Алгоритмы и структуры данных»
Тема: Рекурсивные алгоритмы

Студент гр. 7383 _____ Лосев М. Л.

Преподаватель _____ Размочаева Н. В.

Санкт-Петербург
2018

СОДЕРЖАНИЕ

1. ЦЕЛЬ РАБОТЫ

Цель работы: ознакомление с основными понятиями и приемами рекурсивного программирования, получение навыков программирования рекурсивных процедур и функций на языке программирования C++

Формулировка задачи: Построить синтаксический анализатор для понятия скобки.

скобки::=квадратные | круглые | фигурные

квадратные::=[круглые фигурные] | +

круглые::=(фигурные квадратные) | -

фигурные::={квадратные круглые} | 0

2. РЕАЛИЗАЦИЯ ЗАДАЧИ

Используемые функции:

bool Bracket(char *buf); – функция, получающая на вход строку (выражение) и возвращающая true, если выражение является скобками, или false, если не является

bool Round (char **buf, char s); – функция, получающая на вход строку (выражение), и возвращающая true, если выражение является круглыми скобками, или false, если не является.

bool Square (char **buf, char s); – функция, получающая на вход строку (выражение), и возвращающая true, если выражение является квадратными скобками, или false, если не является.

bool Figure (char **buf, char s); – функция, получающая на вход строку (выражение), и возвращающая true, если выражение является фигурными скобками, или false, если не является.

void Error (short k); – функция, получающая на вход номер ошибки и выводящая соответствующее сообщение об ошибке

Описание алгоритма:

void Load (char *filename, char *buf); **–** функция, получающая на вход имя файла и строку, и загружающая из этого файла выражение в строку.

void UserInterface (); **–** функция, выводящая пользовательское меню и обрабатывающая ввод (выбор опций меню).

Описаие алгоритма функции Bracket:

Функция Bracket проверяет первый символ полученной строки: если он является открывающейся скобкой или +, - или 0, то она вызывает функцию, проверяющую, является ли выражение соответствующими скобками. Потом она проверяет, не осталось ли в строке лишних символов, и возвращает true, если выражение является скобками, или false, если не является. Например, если входная строка - «(0+)», то будет вызвана функция Round (потому что

первый символ „(“ соответствует круглым скобкам), которая вернет true, потом будет проверено, что в строке нет лишних символов, и, так как их не осталось, будет возвращено true.

Описание алгоритма функции Round:

Функция Round проверяет первый символ полученной строки: если он „-“ то она вернет true (так как „-“ по определению является круглыми скобками). Если он „(“ , то она проверит, идут ли дальше фигурные скобки, вызывая Figure: если не идут, то Round тоже вернет false (т. к. внутри круглых скобок должны быть сначала фигурные, а потом квадратные), а если идут, то Round проверит, идут ли после фигурных скобок квадратные, вызывая Square: если не идут, то Round тоже вернет false (т. к. внутри круглых скобок после фигурных должны быть квадратные), а если идут, то Round проверит, идет ли дальше „)“: если да, то она вернет true, а если нет, то вызовет Error (которая выведет соответствующее сообщение), и вернет false. Если первый символ не „-“ и не „(“, то вызовется Error (которая выведет соответствующее сообщение). Если строка кончится раньше, чем встретится „)“, то функция вызовет Error (которая выведет соответствующее сообщение) и вернет false.

Например, если Round получит на вход строку «(0+)», то она проверит первый символ: он окажется „(“. Поэтому она проверит, идут ли дальше фигурные скобки с помощью функции Figure. Figure вернет true (потому что 0 – это фигурные скобки по определению). Поэтому дальше Round проверит, идут ли после фигурных скобок квадратные с помощью функции Square. Square вернет true (потому что „+“ – это квадратные скобки по определению). Поэтому дальше Round проверит, идет ли после квадратных скобок закрывающаяся круглая скобка, и вернет true, потому что закрывающаяся круглая действительно идет после „+“.

Описание алгоритма функций Square и Figure:

Функции Square и Figure полностью аналогичны функции Round.

Описание алгоритма функции Error:

Функция Error принимает целое число — код ошибки. Она выводит соответствующее сообщение об ошибке и не возвращает ничего.

3. ТЕСТИРОВАНИЕ

3.1 ПРОЦЕСС ТЕСТИРОВАНИЯ

Программа собрана в операционной системе Ubuntu 17.04 компилятором gcc. В других ОС тестирование не проводилось.

3.1 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Тестовые случаи в приложении Б. По результатам тестирования было показано, что задача выполнена.

4 . ВЫВОД

В ходе работы была написана программа на языке C++, решающая поставленную задачу с помощью сложной рекурсии. Был получен опыт в написании такого рода программ.

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

```
#define _CRT_SECURE_NO_WARNINGS

#include <iostream>
#include <fstream>
#include <iomanip>
#include <unistd.h>


using namespace std;

bool Bracket(char *buf);
bool Round (char **buf, char s);
bool Figure (char **buf, char s);
bool Square (char **buf, char s);
void Error (short k);

void Load ( char *filename, char *buf )
{
       ifstream infile (filename);
    if (!infile) cout << "Входной файл не открыт" << endl;
    else {
             cout << "Входной файл открыт" << endl;
             infile >> buf;
   }
   cout << "|" << buf << "|" << endl;
}

void UserInterface ()
{
    int key;
    char exp[100] = {0};
    while (true) {
        cout << "Выбери действие:" << endl;
        cout << "0. Завершить выполнение;" << endl;
        cout << "1. Загрузить выражение из пользовательского файла;" << endl;
        cout << "2. Загрузить выражение из файла по умолчанию;" << endl;
        cout << "3. Ввести выражение с клавиатуры;" << endl;
        cin >> key;
        switch (key) {
```

```
    case 0:
        return;
        break;
    case 1:
    {
                cout << "input file name: ";
        char filename[20];
        cin >> filename;
        Load(filename, exp);
        cout << "Анализатор скобок:" << endl;
                bool b = Bracket(exp);
                if (b) cout << endl << "ЭТО СКОБКИ!" << endl;
                else cout << "НЕТ, ЭТО НЕ СКОБКИ!" << endl;
    }
    break;
    case 2:
    {
        Load("in_seq5.txt", exp);
        cout << "Анализатор скобок:" << endl;
                bool b = Bracket(exp);
                if (b) cout << endl << "ЭТО СКОБКИ!" << endl;
                else cout << "НЕТ, ЭТО НЕ СКОБКИ!" << endl;
    }
    break;
    case 3:
    {
                cout << "input expression: ";
                cin >> exp;
                cout << "Анализатор скобок:" << endl;
                bool b = Bracket(exp);
                if (b) cout << endl << "ЭТО СКОБКИ!" << endl;
                else cout << "НЕТ, ЭТО НЕ СКОБКИ!" << endl;
    }
    break;

    //

    default : cout << "! - ...";
    break;
    // ?
};
```

```
    }
}


int main ( )
{
    UserInterface();
    return 0;

}

bool Square (char **buf, char s)
// квадратные::=[круглые фигурные] | +
// s — текущий символ входной строки
{   bool k;
    if (s == '+') { return true;}
    else if ( s == '[' )
        {   if (*((*buf)++))
            {   cout << **buf;
                k = Round (buf, **buf);
                if (k)
                {   if (*((*buf)++))
                    {   cout << **buf;
                        k = Figure (buf, **buf);}
                    else {Error (5); return false;} // квадр - пуст!
                }
                else return false; //первый квадр ошибочен
                if (k) // оба квадр правильны
                    if (*((*buf)++))
                    {   if (**buf != ']') {Error(3 ); return false;}
                        cout << **buf;
                        return true;
                    }
                    else {Error (3); return false;}
                else return false;
            }
            else { Error (5); return false;} // квадр — пуст!
        }
    else { Error(4); return false;} // не + и не [
}
// end of Square
```

```
bool Round (char **buf, char s)
// круглые::=(фигурные квадратные) | -
// s — текущий символ входной строки
{   bool k;
    if (s == '-') { return true;}
    else if ( s == '(' )
        {   if (*((*buf)++))
            {   cout << **buf;
                k = Figure (buf, **buf);
                if (k)
                {   if (*((*buf)++))
                    {   cout << **buf;
                        k = Square (buf, **buf);}
                    else {Error (8); return false;} // кругл - пуст!
                }
                else return false; //первый квадр ошибочен
                if (k) // оба кругл правильны
                    if (*((*buf)++))
                    {   if (**buf != ')') {Error(6); return false;}
                        cout << **buf;
                        return true;
                    }
                    else {Error (6); return false;}
                else return false;
            }
        else { Error (8); return false;} // кругл — пуст!
        }
    else { Error(7); return false;} // не – и не (
}
// end of Round


bool Figure (char **buf, char s)
// фигурные::={квадратные круглые} | 0
// s — текущий символ входной строки
{   bool k;
    if (s == '0') { return true;}
    else if ( s == '{' )
        {   if (*((*buf)++))
            {   cout << **buf;
```

```
                k = Square (buf, **buf);
                if (k)
                {   if (*((*buf)++))
                    {   cout << **buf;
                        k = Round (buf, **buf);}
                    else {Error (11); return false;} // фиг - пуст!
                }
                else return false; //первый квадр ошибочен
                if (k) // оба квадр правильны
                    if (*((*buf)++))
                    {   if (**buf != '}') {Error(9); return false;}
                        cout << **buf;
                        return true;
                    }
                    else {Error (9); return false;}
                else return false;
            }
            else { Error (11); return false;} // фиг — пуст!
        }
    else { Error(10); return false;} // не – и не ( }
}
// end of Square

bool Bracket(char *buf)
{   char s;
    bool b;
    b = false;
    s = *(buf);
    if (s)
    {   cout << s;
        if ((s == '+') || (s == '[')) b = Square (&buf, s);
        else if ((s == '-') || (s == '(')) b = Round (&buf, s);
            else if ((s == '0') || (s == '{')) b = Figure (&buf, s);
                else Error(2); //недопустимый начальный символ
        s = ((*buf)++);
        if (b && !(*buf)) Error(1); // лишние символы
        b = (b && *buf);
    }
    else Error (0); // пустая входная строка
    return b;
}
```

```
void Error (short k)
{
    cout << endl << "err#" << k << endl;
    switch (k) {
        case 0: cout << "! - Пустая входная строка" << endl; break;
        //{Bracket}
        case 1: cout << "! - Лишние символы во входной строке" << endl; break;
        //{Bracket}
        case 2: cout << "! - Недопустимый начальный символ" << endl; break;
        //{Bracket}
        case 3: cout << "! - Отсутствует ']'." << endl; break;
        //{Square}
        case 4: cout << "! - Отсутствует '+' или '['." << endl; break;
        //{Square}
        case 5: cout << "! - Очередной квадр — пуст." << endl; break;
        //{Round}
        case 6: cout << "! - Отсутствует ')'." << endl; break;
        //{Round}
        case 7: cout << "! - Отсутствует – или (." << endl; break;
        //{Round}
        case 8: cout << "! - Очередной кругл — пуст." << endl; break;
        //{Round}
         case 9: cout << "! - Отсутствует '}'." << endl; break;
        //{Figure}
        case 10: cout << "! - Отсутствует 0 или {." << endl; break;
        //{Figure}
        case 11: cout << "! - Очередной фиг — пуст." << endl; break;
        //{Figure}
        default : cout << "! - ...";break;
        // ?
    };
}
// end of Error
```

ПРИЛОЖЕНИЕ Б
ТЕСТОВЫЕ СЛУЧАИ

Входное выражение	**Вывод программы**	**Корректность выполнения**
(	Анализатор скобок: (err#10 ! - Отсутствует 0 или {. НЕТ, ЭТО НЕ СКОБКИ!	Да
(0	Анализатор скобок: (0 err#4 ! - Отсутствует '+' или '['. НЕТ, ЭТО НЕ СКОБКИ!	Да
(d	Анализатор скобок: (d err#10 ! - Отсутствует 0 или {. НЕТ, ЭТО НЕ СКОБКИ!	Да
abracadabre	Анализатор скобок: a err#2 ! - Недопустимый начальный символ НЕТ, ЭТО НЕ СКОБКИ!	Да
(0+)	Анализатор скобок: (0+) ЭТО СКОБКИ!	Да
(0asd	Анализатор скобок: (0a err#4 ! - Отсутствует '+' или '['. НЕТ, ЭТО НЕ СКОБКИ!	Да
{[(0[(0+)0]){+-}](0+)}	Анализатор скобок: {[(0[(0+)0]){+-}](0+)}	Да

	ЭТО СКОБКИ!	
(0[12])	Анализатор скобок: (0[1 err#7 ! - Отсутствует – или (. НЕТ, ЭТО НЕ СКОБКИ!	Да