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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Алгоритмы и структуры данных»

Тема: Рекурсия

Студент гр. 8304 _____ Бочаров Ф.Д.

Преподаватель _____ Фирсов М.А.

Санкт-Петербург

2019

Цель работы.

Изучить основы рекурсии и составления эффективных алгоритмов.

Постановка задачи.

1) Разработать программу, использующую рекурсию;
2) Сопоставить рекурсивное и итеративное решение задачи;
3) Сделать вывод о целесообразности и эффективности рекурсивного подхода для решения данной задачи.

Вариант 12

Построение синтаксического анализатора для понятия *скобки*.

скобки::=квадратные | круглые | фигурные

квадратные::=[круглые фигурные] | +

круглые::=(фигурные квадратные) | -

фигурные::={квадратные круглые} | 0

Описание алгоритма.

Для решения поставленной задачи была реализован набор рекурсивных функций, которые анализируют входное выражение. На каждом этапе программа анализирует данные и передает их в другую функцию на проверку. Таким образом самый первый вызов функции возвращает true or false.

Спецификация программы.

Программа предназначена для синтаксического анализа выражения методом рекурсии.

Программа написана на языке C++. Входными данными является анализируемая строка. Выходными данными являются промежуточные значения анализа скобок.

Описание функций.

1) Функция bool isbracketsequence(string); - Основная функция, реализующая описание определения.

bool issquare(string);

bool iscircle(string);

bool isfigure(string);

Данные функции

bool issquare(string);

Осуществляет проверку на то, удовлетворяет ли входная строка поставленной задаче. Функция принимает проверяемую строку, ссылку на поток вывода и имеет параметр cnt, который по умолчанию равен 0, он используется для подсчета символов. Программа по условию проверяет на количество символов. Циклом ищет ‘(‘ и закрывающую ‘)’, если условие выполняется то записывает содержимое в в подстроку и далее передает строку следующей функции анализатора.

Вывод.

Был получен опыт работы с рекурсией и с построением синтаксического анализатора. На мой взгляд, итеративное решение поставленной задачи более эффективно.

ПРИЛОЖЕНИЕ

1) ТЕСТИРОВАНИЕ:

Работа программы для строки

({[(0+){+-}]({+-}[-0])}[({+-}[-0]){[-0](0+)}])

Таблица результатов ввода/вывода тестирования программы

Входная строка	Вывод программы
+	True
-	True
0	True
2	True
[(0+){+-}]	True
{[-0](0+)}	True
({+-}[-0])	True
[({+-}[-0]){[-0](0+)}]	True
({[-0](0+)}[{0+}{+-}])	False
{[[0+]{+-}]({+-}[-0])}	False
({[(0+){+-}]({+-}[-0])}[({+-}[-0]){[-0](0+)}])	True

{[(0+)[+-]}	False

2) ИСХОДНЫЙ КОД:

```
#include <vector>
#include <fstream>
#include <iostream>
#include <string>
#include <iostream>
using namespace std;

bool isbracketsequence(string);
bool issquare(string);
bool iscircle(string);
bool isfigure(string);

bool isbracketsequence(string str)
{
    return issquare(str) || iscircle(str) || isfigure(str);
}

bool issquare(string str)
{
    if (str == "+")
        return true;

    if (str.size() < 4) return false; // По условию кв.ск. =
[ab], при этом не менее 4 символов

    int cnt = 0;

    for (int i = 1; i <= str.size() - 2; i++) // не учитываем
первую скобку и последнюю
    {
        if (str[i] == '(') //по условию круглые скобки
            cnt++;
        if (str[i] == ')')
            cnt--;
        if (!cnt)
        {
```

```
            cnt = i;
            break;
        }
    }

    if (cnt == str.size() - 2)
        return false; // в строке нет места для других скобок

    string substr1 = "";
    string substr2 = "";

    for (int i = 1; i <= cnt; i++)
        substr1 += str[i]; // записываю первые скобки
    for (int i = cnt + 1; i <= str.size() - 2; i++)
        substr2 += str[i]; // вторые скобки

    cout << "Analyzing issquare : " << str << std::endl;

    return ((str[0] == '[') && (str[str.size() - 1] == ']') &&
iscircle(substr1) && isfigure(substr2));
}

bool iscircle(string str)
{
    if (str == "-")
        return true;

    if (str.size() < 4) return false;

    int cnt = 0;

    for (int i = 1; i <= str.size() - 2; i++)
    {
        if (str[i] == '{')
            cnt++;
        if (str[i] == '}')
            cnt--;
        if (!cnt)
        {
            cnt = i;
            break;
        }
    }
```

```
    if (cnt == str.size() - 2)
        return false;

    string substr1 = "";
    string substr2 = "";

    for (int i = 1; i <= cnt; i++)
        substr1 += str[i];
    for (int i = cnt + 1; i <= str.size() - 2; i++)
        substr2 += str[i];


    cout << "Analyzing iscircle : " << str << std::endl;


    return  ((str[0] == '(') && (str[str.size() - 1] == ')')
&& isfigure(substr1) && issquare(substr2));
}
//
bool isfigure(string str) {

    if (str == "0")
        return true;

    if (str.size() < 4) return false;

    int cnt = 0;

    for (int i = 1; i <= str.size() - 2; i++)
    {
        if (str[i] == '[')
            cnt++;
        if (str[i] == ']')
            cnt--;
        if (!cnt)
        {
            cnt = i;
            break;
        }
    }

    if (cnt == str.size() - 2)
        return false;
```

```
    string substr1 = "";
    string substr2 = "";

    for (int i = 1; i <= cnt; i++)
        substr1 += str[i];
    for (int i = cnt + 1; i <= str.size() - 2; i++)
        substr2 += str[i];



    cout << "Analyzing isfigure: " << str << std::endl;


    return ((str[0] == '{') && (str[str.size() - 1] == '}') &&
issquare(substr1) && iscircle(substr2));
}


int main()
{
        std ::string str;
        std::ifstream input;
         char tmp;
         string proc_str = "";
             input.open("test.txt");
         if (!input)
        {
            std::cout << "Couldnt open file";
            }
            while(input.get(tmp)){
                if (tmp != '\n')
                {
                str +=tmp;

                }



                else
                {
                if (isbracketsequence(str)) cout << str << "
== Its bracket" <<endl <<endl;
                else cout << str << " == Its not bracket"
<<endl<<endl;
```

```
                str = "";
                }
            }

    }
```