## 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. 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)

[Функция main 3](#__RefHeading___Toc8605_2844259154)

[Функция check\_expr 3](#__RefHeading___Toc8607_2844259154)

[Функция check\_expr\_impl 3](#__RefHeading___Toc8609_2844259154)

[Вывод 4](#__RefHeading___Toc1731_692322312)

[Приложение A. Текст программы 5](#__RefHeading___Toc4028_623175150)

[Приложение Б. Тест 6](#__RefHeading___Toc4030_623175150)

# Цель работы

Ознакомиться с основными понятиями и приёмами рекурсивного программирования, получить навыки программирования рекурсивных процедур и функций.

# Задание

Построить синтаксический анализатор для понятия *простое выражение.*

*простое\_выражение::=простой\_идентификатор* |

(*простое\_выражение знак\_операции простое\_выражение*)

*простой\_идентификатор::= буква*

*знак\_операции:: = – | + | \**

# Ход работы

## **Функция main**

Программа принимает на вход путь к файлу со входными данными — выражениями. Создается два файловых потока: для считывания и для вывода, а также вектор строк, в котором будут хранится выражения. Инициализируется переменная spacing — длина самого большого выражения (нужно для форматированного вывода). Затем в цикле происходит вывод в файл соответствующих значения для каждого выражения.

## Функция check\_expr

Принимает строку — выражение, которое нужно проверить. В случае если выражение пустое или оно не начинается с открывающей скобки, возвращается false. Иначе создается строковый поток из переданного аргумента и затем из функции возвращается значение полученное из check\_expr\_impl.

## **Функция check\_expr\_impl**

Принимает два аргумента — строковый поток с выражением и флаг, который дает понять первый ли это рекурсивный вызов или нет. Далее логика данной функции заключается в следующем:

1.) После любой открывающей скобки должно следовать выражение → нужно вызвать функцию рекурсивно на этом выражении (из этого пункта сразу вытекает условие конца рекурсии — простейший случай выражение — литерал)

2.) После первого выражения должен следовать оператор.

3.) После оператора должно следовать второе выражение.

4.) После второго выражения должна следовать закрывающая скобка

\*5.) Если мы находимся в первом рекурсивном вызове, то после скобки также не должно ничего следовать.

# Вывод

Была написана программа анализирующая правильность синтаксического выражения с использованием рекурсии. При компиляции и выполнении программы не возникает ошибок и предупреждений.

# Приложение A. Текст программы

#include <iostream>

#include <cctype>

#include <fstream>

#include <sstream>

#include <iomanip>

#include <vector>

#define FIRST\_CALL true

using namespace std;

bool check\_expr\_impl(istringstream& istr, bool call = false) {

bool res;

char c;

istr >> c;

if (isalpha(c))

return true;

else if (c == '(')

res = check\_expr\_impl(istr);

else

return false;

istr >> c;

if (res && (c == '+' || c == '\*' || c == '-'))

res = check\_expr\_impl(istr);

else

return false;

istr >> c;

if (res && c == ')') {

// scratch

if (call == FIRST\_CALL) {

istr.peek();

if (istr.eof())

return true;

else

return false;

}

// scratch

return true;

} else

return false;

}

bool check\_expr(string const& expr) {

if (expr.size() == 1 && isalpha(expr.at(0)))

return true;

if (expr.empty() || expr.at(0) != '(')

return false;

istringstream str{expr};

return check\_expr\_impl(str, FIRST\_CALL);

}

int main(int argc, char\* argv[]) {

ifstream i\_file(argv[1]);

ofstream o\_file("lab1\_output.txt");

std::vector<string> exprs;

// for setw() to find spacing

size\_t spacing = 0;

string expr;

while(getline(i\_file, expr).good()) {

spacing = max(spacing, expr.size());

exprs.push\_back(expr);

}

o\_file << boolalpha;

for (auto& expr: exprs)

o\_file << setw(spacing) << left << expr <<" : " << check\_expr(expr) << endl;

i\_file.close();

o\_file.close();

return 0;

}

# Приложение Б. Тест

