**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: Рекурсия**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6383 |  | Спас А.А. |
| Преподаватель |  | Шолохова О.М. |

Санкт-Петербург

2017

**Цель работы**

Ознакомиться с основными понятиями и приёмами рекурсивного программирования, получить навыки программирования рекурсивных процедур и функций на языке программирования С++.

**Задание (вариант №20)**

Построить синтаксический анализатор понятия *список\_параметров.*

*список\_параметров::= параметр*|*параметр , список\_параметров*

*параметр::= имя=цифра цифра | имя=*(*список\_параметров*)

*имя::= буква буква буква*

**Основные теоретические положения**

Функция факториал натурального аргумента n обозначается как n! и определяется соотношением

n!=1·2·3...(n – 1)·n . (2.1)

Удобно доопределить 0!=1 и считать, что n – целое неотрицательное число.

Некоторым недостатком определения (2.1) является наличие в нём многоточия «...», передающего речевой оборот «и так далее» и имеющего интуитивно понятный читателю смысл. Можно дать точное, так называемое рекурсивное определение функции n!, лишенное этого недостатка, т. е. не апеллирующее к нашей интуиции. Определим:

а) 0! = 1, (2.2)

б) n! = (n – 1)!·n при n > 0.

Соотношения (2.2) можно рассматривать как свойства ранее определенной функции, а можно (как в данном случае) использовать их для определения этой функции.

Далее для функции n! Используется «функциональное» (префиксное) обозначение fact(n), указывая имя функции и за ним в скобках – аргумент. Тогда (2.2) можно записать в виде

⎧ 1, если *n* = 0;

fact (*n*) = ⎨ (2.3)

⎩ fact (*n* − 1) ⋅ *n*, если *n* > 0;

или в другой форме записи

fact (*n*) ≡ **if** *n* = 0 **then** 1 **else** fact (*n* − 1) ⋅ *n*, (2.4)

где использовано условное выражение **if** *b* **then** *e*1**else** *e*2, означающее, что в том месте, где оно записано, следует читать e1, если выполняется условие b, и следует читать e2, если условие b не выполняется.

Функция, определяемая таким образом, единственна. Действительно, пусть есть две функции, например: fact1 (n) и fact2 (n), удовлетворяющие со-отношениям (2.2) или их эквивалентам (2.3), (2.4). Рассмотрим разность dfact (n) = fact1 (n) − fact2 (n). Очевидно, что, во-первых, в силу соотношения «а» из (2.2) имеем dfact (0) = 0, а, во-вторых, для функции dfact (n) также справедливо соотношение «б». Действительно,

dfact (n) = fact1 (n) − fact2 (n) = fact1 (n − 1) · n − fact2 (n − 1) · n =

= (fact1 (n − 1) − fact2 (n − 1)) · n = dfact (n − 1) · n.

По индукции легко доказывается, что из соотношений dfact (0) = 0 и dfact (n) = dfact (n − 1) · n следует, что dfact (n) = 0 для любого n > 0.

**Спецификация программы**

*Назначение программы*.

Программа предназначена для определения понятия «список параметров».

*Описание программы*.

Программа написана на языке C++. Входными данными для программы являются символы, вводимые пользователем с клавиатуры. Проверка значений при некорректном вводе количества символов предусмотрена. Выходными данными является ответ на вопрос: «соответствует ли введенная комбинация понятию «список парметров»?».

*Пример диалога с пользователем*.

1. Введите комбинацию:

aaa=11

Является списком параметров

1. Введите комбинацию:

mad=(man=66)

Является списком парметров

1. Введите комбинацию:

god=(man=777)

Не является списком параметров

*Реализация*.

Функции:

1. bool imya (std::istream& in);
2. bool parametr (std::istream& in);
3. bool spisok (std::istream& in);

Эти три функции принимают на вход строку символов и обрабатывают эту введенную комбинацию для определения понятия «список парметров». Они взаимодействуют между собой с помощью рекурсий.

**Тестирование**

|  |  |
| --- | --- |
| **Комбинация** | **Результат** |
| aaa=11 | является списком параметров |
| AaA=19 | является списком параметров |
| saw | не является списком параметров |
| 45 | не является списком параметров |
| kra=23,sno=93,dar=23 | является списком параметров |
| Kra=(sik=93) | является списком параметров |
| Hel=77, ili=66 | не является списком параметров |
| tss=77,brr=(rrr=(aaa=11)) | является списком параметров |
| let=(sss=(get=(itt=99))) | является списком параметров |

**Вывод**

Мною была разработана программа, которая выясняет, задано ли определение «скобки» пользователем. В результате выполнения работы был получен опыт программирования на языке С++, а также навыки программирования рекурсивных функций.

**Приложение А. Исходный код.**

#include <iostream>

#include <Windows.h>

using namespace std;

/\*\*

\* список параметров: параметр| параметр, список параметров

\* параметр : имя = цифра цифра | имя=(список параметров)

\* имя : буква буква буква

\*/

bool spisok(std::istream& in);

bool imya(std::istream& in) {

for (int i = 0; i < 3; ++i) {

if (isalpha(in.peek())) in.get();

else return false;

}

return true;

}

bool parametr(std::istream& in) {

if (!imya(in))return false;

if (in.peek() != '=') {

return false;

}

in.get();

if (in.peek() == '(') {

in.get();

spisok(in);

if (in.peek() != ')')return false;

in.get();

return true;

}

else {

for (int i = 0; i < 2; ++i) {

if ('0' <= in.peek() <= '9') in.get();

else return false;

}

}

return true;

}

bool spisok(std::istream& in) {

if (!parametr(in)) return false;

if (in.peek() == '\n') return true;

if (in.peek() != ',')return false;

else {

in.get();

spisok(in);

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Введите комбинацию: ";

if (spisok(cin)) cout << "Является списком параметров\n";

else cout << "Не является списком параметров\n";

system("pause >nul");

return 0;

}