МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«24» 12 2017 г.

ИСПОЛЬЗОВАНИЕ ФУНКЦИОНАЛОВ В ПРОГРАММАХ НА ЯЗЫКЕ LISP

Методические указания по выполнению лабораторной работы по дисциплине «Функциональное и логическое программирование» для студентов направления подготовки 09.03.04 «Программная инженерия»

Курск 2017

УДК 004.65

Составители: В.Г. Белов, Т.М. Белова

Рецензент

Кандидат технических наук, доцент кафедры программной инженерии ЮЗГУ И.Н. Ефремова

Использование функционалов в программах на языке LISP: методические указания по выполнению лабораторной работы по дисциплине "Функциональное и логическое программирование" для студентов направления подготовки 09.03.04 "Программная инженерия" / Юго-Зап. гос. ун-т; сост.: В.Г. Белов, Т.М. Белова, – Курск, 2017. – 12 с.: ил. 2.

Рассмотрена методика использования функционалов в программах на языке Lisp на примере символьного дифференцирования.

Материал предназначен для студентов направления подготовки 09.03.04 «Программная инженерия» специализации «Разработка программно-информационных систем»

Подписано в печать 24.12.17. Формат 60x84 1/16.
Усл. печ. л. 0,6. Уч.-изд. л. 0,5. Тираж 100 экз. Заказ 4384. Бесплатно.
Юго-Западный государственный университет

305040, Курск, ул.50 лет Октября, 94.

Содержание

1 ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ 4
2 ПРЕОБРАЗОВАНИЕ АРИФМЕТИЧЕСКОГО ВЫРАЖЕНИЯ В ОПЕРАЦИОННОЕ ДЕРЕВО 5
3 РАБОТА АЛГОРИТМА ПОСТРОЕНИЯ ОПЕРАЦИОННОГО ДЕРЕВА.....8
4 ДИФФЕРЕНЦИРОВАНИЕ 9
5 ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ РАБОТЫ.....10
6 СОДЕРЖАНИЕ ОТЧЕТА 11
7 КОНТРОЛЬНЫЕ ВОПРОСЫ К ЗАЩИТЕ ЛАБОРАТОРНОЙ РАБОТЫ.....12

1 ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

Целью лабораторной работы является приобретение умения работы со списками свойств на примере решения задачи символьного дифференцирования.

2 ПРЕОБРАЗОВАНИЕ АРИФМЕТИЧЕСКОГО ВЫРАЖЕНИЯ В ОПЕРАЦИОННОЕ ДЕРЕВО

Для упрощения процедуры дифференцирования арифметического выражения его можно представить в виде операционного дерева. В операционном дереве имена арифметических операцийи функций являются именами функций, определенных в LISP-программах составной структуре данных, которой и является само операционное дерево.

Арифметическому выражению

A - B * C + D / K

соответствует операционное дерево

(+ (- (* B C)) (/ D K)) .

Список приоритетов функций и операций, используемых в операционном дереве, имеет вид

(EXP LN SIN COS TG MINUS **/ *- +),

где EXP имеет наивысший приоритет, а + наименьший.

Список операций, используемых в операционном дереве, имеет вид

(**/ * - +) .

Для построения операционного дерева на основании арифметического выражения можно использовать приведенный ниже алгоритм:

1. Если выражение пусто, то перейти к п.7., иначе – взять следующий элемент из выражения.

2. Если элемент является переменной или константой, то поместить его в начало дерева и перейти к п.1.

3. Если элемент является операцией или именем функции и стек пуст, то поместить элемент в стек и перейти к п.1.

4. Если приоритет операции или функции выше приоритета операции или функции, находящейся в вершине стека, то поместить ее в стек и перейти к п.1.

5. Если приоритет операции или функции меньше или равен приоритету операции или функции в вершине стека, то исключить операцию или функцию из вершины стека, преобразовать первые элементы дерева на основании исключенной операции или функции, перейти к п.3. для следующего сравнения выбранной ранее из выражения операции или функции.

6. Если элемент является выражением в круглых скобках, то получить из этого выражения поддерево по данному алгоритму, поместить его в начало формируемого дерева, перейти к п.1.

7. Если стек пуст, то дерево построено и алгоритм закончен, в противном случае исключить операцию или функцию из вершины стека,

преобразовать первые элементы дерева на основании исключенной операции или функции, перейти к п.7.

Для преобразования первых элементов дерева на основании операции или имени функции можно использовать приведенные ниже описания процедур:

1. Если выбранный из стека элемент является именем функции, то сформировывается список из имени функции и первого элемента непреобразованного дерева; удалить этот элемент из дерева; присоединить к началу дерева сформированный список.

В противном случае сформировать список из операции, второго элемента, первого элемента непреобразованного дерева; удалить первый и второй элементы из дерева; присоединить к началу дерева сформированный список.

2. Вызвать алгоритм построения операционного дерева на основании выражения для полученных состояний дерева, стека и выражения.

3 РАБОТА АЛГОРИТМА ПОСТРОЕНИЯ ОПЕРАЦИОННОГО ДЕРЕВА

На рисунке 1 приведена трассировка алгоритма построения операционного дерева.

N	Операционное дерево	Стек	Исходное выражение	Прим.
0	()	()	(A - B * C + D / K)	п.2.
1	(A)	()	(- B * C + D / K)	п.3.
2	(A)	(-)	(B * C + D / K)	п.2.
3	(B A)	(-)	(* C + D / K)	п.4.
4	(B A)	(* -)	(C + D / K)	п.2.
5	(C B A)	(* -)	(+ D / K)	п.5.
6	((* B C) A)	(-)	(+ D / K)	п.5.
7	((- A (* B C)))	()	(+ D / K)	п.3.
8	((- A (* B C)))	(+)	(D / K)	п.2.
9	(D (- A (* B C)))	(+)	(/ K)	п.4.
10	(D (- A (* B C)))	(/ +)	(K)	п.2.
11	(K D (- A (* B C)))	(/ +)	()	п.7.
12	((/ D K) (- A (* B C)))	(+)	()	п.7.
13	(+ (- A (* B C)) (/ D K))	()	()	п.7.

Рисунок 1

На рисунке 2 приведен один из примеров программы вычисления значения по операционному дереву.

```
(put 'op '+ '+op )
(defun +op(lst)
(+ (cont (car lst)) (cont (cadr lst))))
(put 'op '- '(lambda (lst)
(- (cont (car lst)) (cont (cadr lst)))))
(put 'op '*'(lambda (lst)
 (* (cont (car lst)) (cont (cadr lst)))))
(put 'op '/ '(lambda (lst)
 (/ (cont (car lst)) (cont (cadr lst)))))
(defun cont (lst)
( (numberp (car lst)) (car lst))
(apply (get 'op (car lst)) (cdr lst) nil ))
```

Рисунок 2

4 ДИФФЕРЕНЦИРОВАНИЕ

Программа дифференцирования должна быть основана на правиле дифференцирования сложной функции: производная сложной функци и равна производной этой функции по ее аргументу, умноженной на производную аргумента по его аргументу.

Терминальными случаями в этом рекурсивном определении являются те, в которых аргумент является переменной или константой.

5 ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ РАБОТЫ

1. Составить и отладить программу, реализующую преобразование арифметического выражения в операционное дерево.

2. Составить и отладить программу, реализующую дифференцирование операционного дерева.

3. Составить и отладить программу, реализующую упрощение операционного дерева, полученного в результате дифференцирования. Для этого необходимо удалить из него умножения на единицу, сложения с нулем, суммированием и умножением констант и привести подобные члены.

4. На основании пунктов 4.1.- 4.3. составить и отладить программу, реализующую дифференцирование арифметического выражения.

5. Разработать программу преобразования упрощенного операционного дерева в выражение в префиксной (исходной форме).

6 СОДЕРЖАНИЕ ОТЧЕТА

В отчет должны быть включены все программы из раздела 4. с результатами их трассировки на тестовых примерах.

7 КОНТРОЛЬНЫЕ ВОПРОСЫ К ЗАЩИТЕ ЛАБОРАТОРНОЙ РАБОТЫ

1. Какую структуру имеет список свойств?

2. С помощью какой функции осуществляется добавление и изменение списка свойств?

3. Назовите принципы построения операционного дерева по арифметическому выражению?

4. Как осуществляется преобразование первыхэлементов дерева на основании операции или имени функции?