Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа №8 по дисциплине «Функциональное и логическое программирование»

Тема apply и funcall

Студент Александров Э.И.

Группа ИУ7-53БВ

Преподаватель Строганов Ю.В.

Москва, 2024

Содержание

ВВЕДЕНИЕ

Цель данной работы — продемонстрировать различия между функциями apply и funcall в языке программирования Lisp, а также решить квадратное уравнение с использованием обеих функций. Задачи включают в себя реализацию функции для решения квадратного уравнения и её вызов с помощью apply и funcall.

1 Аналитическая часть

В языке Lisp функции apply и funcall используются для вызова других функций, но имеют разные подходы к передаче аргументов. funcall принимает функцию и её аргументы в виде отдельных параметров, тогда как apply принимает функцию и список аргументов, который будет распакован при вызове. Это различие позволяет использовать apply в ситуациях, когда количество аргументов заранее неизвестно или когда они хранятся в списке.

Вывод

Вывод: В аналитической части мы рассмотрели основные отличия между apply и funcall, что поможет лучше понять их применение в программировании на Lisp.

2 Конструкторская часть

Алгоритм решения квадратного уравнения имеет следующий вид:

1) Вычислить дискриминант:

$$D = b^2 - 4ac \tag{2.1}$$

2) Найти корни уравнения:

$$x_1 = \frac{-b + \sqrt{D}}{2a} \tag{2.2}$$

$$x_2 = \frac{-b - \sqrt{D}}{2a} \tag{2.3}$$

3) Вернуть корни в виде списка.

Архитектура программы состоит из одной функции solve-quadratic, которая принимает три аргумента (коэффициенты уравнения) и возвращает корни уравнения.

Вывод

В конструкторской части мы разработали алгоритм решения квадратного уравнения и описали его архитектуру.

3 Технологическая часть

Реализация была выполнена на языке программирования Lisp. Для написания и тестирования кода использовалась среда разработки VS Code с установленным пакетом Common Lisp, а также компилятор Steal Bank Common Lisp.

Коды алгоритмов:

Листинг 3.1 — Функция квадратного уравнения

```
(defun solve-quadratic (a b c)
  (let* ((d (- (* b b) (* 4 a c)))
         (sqrt-d (sqrt d))
         (x1 (/ (+ (- b) sqrt-d) (* 2 a)))
         (x2 (/ (- (- b) sqrt-d) (* 2 a))))
  (list x1 x2)))
```

Листинг 3.2 — Вызов функции с использованием funcall

```
(let ((a 1) (b -3) (c 2))
  (print (funcall 'solve-quadratic a b c)))
```

Листинг 3.3 — Вызов функции с использованием apply

```
(let ((a 1) (b -3) (c 2))
  (print (apply 'solve-quadratic (list a b c))))
```

Тестовые данные: Для коэффициентов a=1, b=-3, c=2 программа должна вернуть корни $x_1 = 2$ и $x_2 = 1$.

Листинг 3.4 — Результат

```
(2 1)
```

Все тесты пройдены успешно.

Вывод

В технологической части мы реализовали программу на Lisp, протестировали её и убедились в корректности работы.

ЗАКЛЮЧЕНИЕ

Цель работы заключалась в демонстрации различий между apply и funcall, а также в решении квадратного уравнения с использованием этих функций. В результате мы разработали функцию для решения квадратного уравнения, протестировали её, запустили с apply и funcall и получили корректные результаты.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Пол Грэм, ANSI Common Lisp - СПб.: Символ-Плюс, 2012. - 448 с.
2. Сайт "Применяющии функционалы Apply и Funcall"— [https://lisp2d.net/rus/teach/x.html]
3. Введение в язык Lisp. Apply и Funcall — [http://homelisp.ru/help/lisp.html#u25]

Приложение А

Код программы:

Листинг 3.5 — Функция квадратного уравнения

```
(defun solve-quadratic (a b c)
(let* ((d (- (* b b) (* 4 a c)))
(sqrt-d (sqrt d))
(x1 (/ (+ (- b) sqrt-d) (* 2 a)))
(x2 (/ (- (- b) sqrt-d) (* 2 a))))
(list x1 x2)))
```

Листинг 3.6 — Вызов функции с использованием funcall

```
(let ((a 1) (b -3) (c 2))
(print (funcall 'solve-quadratic a b c)))
```

Листинг 3.7 — Вызов функции с использованием apply

```
(let ((a 1) (b -3) (c 2))
(print (apply 'solve-quadratic (list a b c))))
```