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 №9 по дисциплине «Функциональное и логическое программирование»

Тема mapcar и reduce

Студент Александров Э.И.

Группа ИУ7-53БВ

Преподаватель Строганов Ю.В.

Москва, 2024

Содержание

ВВЕДЕНИЕ

Цель данной работы заключается в изучении и реализации основных операций с векторами и системами счисления с использованием языка программирования Lisp. Задачи, поставленные в рамках работы, включают реализацию векторного произведения, декартова произведения и перевода чисел из N-ричной системы счисления в десятичную с использованием функций mapcar и reduce.

1 Аналитическая часть

Векторное произведение — это операция, которая принимает два вектора в трехмерном пространстве и возвращает вектор, перпендикулярный обоим исходным векторам. Векторное произведение векторов

$a = (x_1, y_1, z_1)$ и $b = (x_2, y_2, z_2)$ вычисляется по формуле: $axb = (x = y_1z_2 - z_1y_2, y = z_1x_2 - x_1z_2, z = x_1y_2 - y_1x_2)$.

Декартово произведение — это операция, которая создает все возможные упорядоченные пары из двух множеств. Если A и B — два множества, то их декартово произведение A×B определяется как множество всех пар (a,b), где $a \in A$ и $b \in B$.

Перевод из N-ричной системы счисления в десятичную — это процесс, который позволяет преобразовать число, представленное в произвольной системе счисления, в десятичное представление. Для числа $d_k\, d_{k-1}\, \ldots\, d_1\, d_0$ в N-ричной системе, его десятичное значение вычисляется по формуле: $\sum_{i=0}^{k} d_i \cdot N^i$, где d_i - цифры числа, а N - основание системы счисления, i - номер разряда начиная с нуля.

Эти операции имеют широкое применение в математике, физике и компьютерных науках.

Вывод

В аналитической части работы были рассмотрены основные математические операции, которые будут реализованы в коде на языке Lisp.

2 Конструкторская часть

В данной части работы были разработаны алгоритмы для реализации векторного произведения, декартова произведения и перевода из N-ричной системы счисления в десятичную.

Для векторного произведения использовалась функция mapcar для обработки компонентов векторов.

Декартово произведение также реализовано с помощью mapcar, что позволяет создать все возможные комбинации элементов двух списков.

Перевод из N-ричной системы в десятичную выполнен с использованием функции reduce для суммирования значений, полученных из каждого разряда.

1) Векторное произведение:
   - — Используется функция mapcar для обработки компонентов векторов.
   - — Входные данные — два вектора, представленные в виде списков.
   - — Выходные данные — новый вектор, представляющий векторное произведение.
2) Декартово произведение:
   - — Реализовано с помощью вложенных вызовов mapcar, что позволяет создать все возможные комбинации элементов двух списков.
   - — Входные данные — два списка.
   - — Выходные данные — список всех упорядоченных пар.
3) Перевод из N-ричной системы в десятичную::
   - — Используется функция reduce для суммирования значений, полученных из каждого разряда.
   - — Входные данные — список цифр и основание системы счисления.
   - — Выходные данные — десятичное представление числа.

Вывод

В конструкторской части были разработаны алгоритмы для выполнения заданных операций, что позволило получить необходимые функции для работы с векторами и системами счисления. Эти алгоритмы обеспечивают корректное выполнение математических операций и могут быть использованы в дальнейшем.

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 — Векторное произведение

```
(defun vector-cross-product (v1 v2)
  (let* ((components (mapcar 'list v1 v2))
      (x1 (nth 0 (nth 0 components)))
      (y1 (nth 0 (nth 1 components)))
      (z1 (nth 0 (nth 2 components)))
      (x2 (nth 1 (nth 0 components)))
      (y2 (nth 1 (nth 1 components)))
      (z2 (nth 1 (nth 2 components))))
  (list (- (* y1 z2) (* z1 y2))
    (- (* z1 x2) (* x1 z2))
    (- (* x1 y2) (* y1 x2)))))
```

Листинг 3.2 — Тестирование векторного произведения

```
(print (vector-cross-product '(1 2 3) '(4 5 6)))
;Result => (-3 6 -3)
```

Листинг 3.3 — Декартово произведение

```
(defun cartesian-product (list1 list2)
  (apply 'append
      (mapcar (lambda (x)
          (mapcar (lambda (y) (list x y)) list2))
        list1)))
```

Листинг 3.4 — Тестирование декартого произведения

```
(print (cartesian-product '(1 2) '(3 4))) ;
;Result => ((1 3) (1 4) (2 3) (2 4))
```

Листинг 3.5 — Перевод из N-ричной системы счисления в 10-чную (1 способ)

```
(defun n-ary-to-decimal (digits base)
  (reduce #'+
    (mapcar (lambda (digit index)
        (* digit (expt base index)))
        (reverse digits)
        (loop for i from 0 below (length digits) collect i))))
```

Листинг 3.6 — Тестирование перевода из N-ричной системы счисления в 10-чную (1 способ)

```
(print (n-ary-to-decimal '(1 0 1) 2))
;Result => 5

(print (n-ary-to-decimal '(15 0 1) 16))
;Result => 3841
```

Листинг 3.7 — Перевод из N-ричной системы счисления в 10-чную (2 способ)

```
(defun n-ary-to-decimal (digits base)
  (reduce #'+
    (mapcar (lambda (digit index)
        (let ((value (case digit
                   (#\0 0) (#\1 1) (#\2 2) (#\3 3)
                   (#\4 4) (#\5 5) (#\6 6) (#\7 7)
                   (#\8 8) (#\9 9) (#\A 10) (#\B 11)
                   (#\C 12) (#\D 13) (#\E 14) (#\F 15)
                   (t (error "Character ~A is not supported." digit)))))
        (* value (expt base index))))
      (reverse digits) ; Reverse to get the correct powers
      (loop for i from 0 below (length digits) collect i))))
```

Листинг 3.8 — Тестирование перевода из N-ричной системы счисления в 10-чную (2 способ)

```
(print (n-ary-to-decimal '(#\A #\F) 16))
;Result => 175

(print (n-ary-to-decimal '(#\1 #\0) 2))
;Result => 2
```

Все тесты пройдены успешно, что подтверждает корректность реализации.

Вывод

В технологической части была реализована функциональность для выполнения заданных операций, что подтвердилось успешным прохождением всех тестов. Реализованные функции позволяют эффективно работать с векторами и системами счисления, что может быть полезно в различных приложениях.

ЗАКЛЮЧЕНИЕ

В ходе работы была достигнута цель — реализация функций для выполнения векторного произведения, декартова произведения и перевода из N-ричной системы счисления в десятичную с использованием фугнкций mapcar и reduce. Все поставленные задачи были успешно выполнены, и полученные результаты подтверждают корректность реализованных алгоритмов. Работа над проектом позволила углубить знания в области функционального программирования и применения языка Lisp для решения математических задач.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Пол Грэм, ANSI Common Lisp - СПб.: Символ-Плюс, 2012. - 448 с.
2. Сайт "The mapcar function"— [https://www.gnu.org/software/emacs/manual/html_node/eintr/mapcar.html]
3. Сайт "Введение в язык Lisp. Mapcar"— [http://homelisp.ru/help/lisp.html#u25]
4. Сайт "The reduce function"— [https://lisp-lang.org/learn/lists]
5. Сайт "Reduce function"— [https://exercism.org/tracks/common-lisp/concepts/reducing]
6. Сайт "Mapping, Filtering, and Reducing"— [https://piembsystech.com/map-filter-and-reduce-data-in-lisp-programming-language/]

Приложение А

Код программы:

Листинг 3.9 — Векторное произведение

```
(defun vector-cross-product (v1 v2)
  (let* ((components (mapcar 'list v1 v2))
      (x1 (nth 0 (nth 0 components)))
      (y1 (nth 0 (nth 1 components)))
      (z1 (nth 0 (nth 2 components)))
      (x2 (nth 1 (nth 0 components)))
      (y2 (nth 1 (nth 1 components)))
      (z2 (nth 1 (nth 2 components))))
  (list (- (* y1 z2) (* z1 y2))
      (- (* z1 x2) (* x1 z2))
      (- (* x1 y2) (* y1 x2)))))
```

Листинг 3.10 — Вызов функции векторного произведения

```
(print (vector-cross-product '(1 2 3) '(4 5 6)))
;Result => (-3 6 -3)
```

Листинг 3.11 — Декартово произведение

```
(defun cartesian-product (list1 list2)
  (apply 'append
      (mapcar (lambda (x)
        (mapcar (lambda (y) (list x y)) list2))
        list1)))
```

Листинг 3.12 — Вызов функции декартого произведения

```
(print (cartesian-product '(1 2) '(3 4))) ;
;Result => ((1 3) (1 4) (2 3) (2 4))
```

Листинг 3.13 — Перевод из N-ричной системы счисления в 10-чную (1 способ)

```
(defun n-ary-to-decimal (digits base)
  (reduce #'+
      (mapcar (lambda (digit index)
          (* digit (expt base index)))
          (reverse digits)
          (loop for i from 0 below (length digits) collect i))))
```

Листинг 3.14 — Вызов функции перевода из N-ричной системы счисления в 10-чную (1 способ)

```
(print (n-ary-to-decimal '(1 0 1) 2))
;Result => 5
```

```
(print (n-ary-to-decimal '(15 0 1) 16))
;Result => 3841
```

Листинг 3.15 — Перевод из N-ричной системы счисления в 10-чную (2 способ)

```
(defun n-ary-to-decimal (digits base)
  (reduce #'+
    (mapcar (lambda (digit index)
      (let ((value (case digit
              (#\0 0) (#\1 1) (#\2 2) (#\3 3)
              (#\4 4) (#\5 5) (#\6 6) (#\7 7)
              (#\8 8) (#\9 9) (#\A 10) (#\B 11)
              (#\C 12) (#\D 13) (#\E 14) (#\F 15)
              (t (error "Character ~A is not supported." digit)))))
      (* value (expt base index))))
      (reverse digits) ; Reverse to get the correct powers
      (loop for i from 0 below (length digits) collect i))))
```

Листинг 3.16 — Вызов функции перевода из N-ричной системы счисления в 10-чную (2 способ)

```
(print (n-ary-to-decimal '(#\A #\F) 16))
;Result => 175

(print (n-ary-to-decimal '(#\1 #\0) 2))
;Result => 2
```