Отчет по лабораторной работе № 1 по курсу «Функциональное программирование»

Студент группы М8О-306 МАИ *Дубинин Артем*, №5 по списку
Контакты: rusartdub@gmail.com
Работа выполнена: 08.03.2019

Преподаватель: Иванов Дмитрий Анатольевич, доц. каф. 806
Отчет сдан:
Итоговая оценка:
Подпись преподавателя:

1. Тема работы

Примитивные функции и особые операторы Common Lisp.

2. Цель работы

Научиться вводить S-выражения в Лисп-систему, определять переменные и функции, работать с условными операторами, работать с числами, использую схему линейной и древовидной рекурсии.

3. Задание (вариант №1.20)

Запрограммируйте на языке Коммон Лисп функцию-предикат, проверяющую, является ли целое число палиндромом, т.е. перевёртышем.
Примеры:

- (palindrome-p 4334) => T
- (palindrome-p 4343) => NIL
- (palindrome-p 434) => T

4. Оборудование студента

Ноутбук Dell Vostro 5568, процессор Intel Core i5-7200U @ 4x 3.1GHz, память: 8Gb, разрядность системы: 64.

5. Программное обеспечение

ОС Ubuntu 18.04 bionic, компилятор sbcl, текстовый редактор Atom.

6. Идея, метод, алгоритм

Если число меньше 10, то число уже палиндром, так как состоит из одной цифры. Иначе, проверяем, чтобы левая цифра числа была равна правой цифре числа, если это не так, то выводим nil, если они равны, то рекурсивно запускаем заново наш алгоритм, с тем условием, что мы отрезаем левую и правую цифру числа.
Например:
Число 1321, оно не меньше 10, следовательно проверяем левую и правую цифру на равенство 1 = 1, следовательно запускаем наш алгоритм для числа 32. Число 32 не меньше 10, проверяем левую и правую цифру, 3 != 2, следовательно возвращаем nil.

7. Сценарий выполнения работы

8. Распечатка программы и её результаты

8.1. Исходный код

```
(defun take-left (n)
  (if (< n 10)
    n
    (take-left (truncate n 10))))

(defun take-right (n)
  (rem n 10))

(defun cut-right (n)
  (truncate n 10))

(defun cut-left (n z m)
  (if (< n 10)
    z
    (cut-left (truncate n 10)
      (+ z
        (*
          (expt 10 m) (rem n 10)))
      (1+ m))))

(defun palindrome-p (n)
  (if (< n 10)
    t
    (if
      (and
```

```
  (eq (take-left n) (take-right n))
  (palindrome-p (cut-left (cut-right n) 0 0 )))
t
nil)))
(defun take-left (n)
  (if (< n 10)
    n
    (take-left (truncate n 10))))

(defun take-right (n)
  (rem n 10))

(defun cut-right (n)
  (truncate n 10))

(defun cut-left (n z m)
  (if (< n 10)
    z
    (cut-left (truncate n 10)
      (+ z
        (*
          (expt 10 m) (rem n 10)))
      (1+ m))))
```

8.2. Результаты работы

```
* (palindrome-p 4334)
T
* (palindrome-p 4343)
NIL
* (palindrome-p 434)
T
* (palindrome-p 1)
T
* (palindrome-p 22)
T
* (palindrome-p 12)
NIL
```

9. Дневник отладки

Дата	Событие	Действие по исправлению	Примечание
09.03.2019	Warning по тому, что главная функция была обьявлена первее функций, которые используются в основной функции ; caught STYLE-WARNING: ; undefined function: *	Главная функция перенесенна в конец программы	

10. Замечания автора по существу работы

Чтобы выполнить работу, нужно было хорошо разобраться с синтаксисом common lisp и с основами функционального программирования.

11. Выводы

При выполнении работы возникли проблемы со средой программирования, современных ide для common lisp не существует, а старые методы программирования emacs + slime, vim + slime, сложны если не хорошо знаком с данными текстовыми редакторами. Сама работа заставила подумать о том, как написать алгоритм в функциональной парадигме.