Отчет по лабораторной работе № 4 по курсу «Функциональное программирование»

Студент группы М8О-306 МАИ *Дубинин Артем*, №5 по списку
Контакты: rusartdub@gmail.com
Работа выполнена: 14.04.2020

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

1. Тема работы

Знаки и строки.

2. Цель работы

Научиться работать с литерами (знаками) и строками при помощи функций обработки строк и общих функций работы с последовательностями.

3. Задание (вариант №3.36)

Запрограммировать на языке Коммон Лисп функцию, принимающую один аргумент - текст.

Если в тексте нет малых букв, функция должна вернуть этот текст без изменения. В противном случае функция должна вернуть копию текста, в котором после всех слов, содержащих хотя бы одну малую букву, вставлен знак пунктуации , (запятая).

Функция должна работать как для малых латинских, так и малых русских букв.

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

Создаем новый текст, изначально пустой, туда мы будем записывать предложения измененные или нет. Сначала пробегаемся по тексту беря каждый элемент списка(строку). Создаем новую строку в которую мы будем копировать слова и другие символы. Далее пробегаемся по оригинальной строке, беря каждый символ и записывая в локальную переменную word, и если встретится пробел, табуляция или перевод строки, то проверяем наше слово. Проверяем мы наше слово на то, встретилась ли в нем lower-case буква, если встретилась, то добавляем к слову запятую, иначе оставляем слово таким, какое оно есть. Добавляем слово в новую строку, так как же, когда мы бежим по оригинальной строке, мы добавляем все символы не являющиеся словами. Измененную новую строку добавляем в новый текст.

7. Сценарий выполнения работы

8. Распечатка программы и её результаты

8.1. Исходный код

```
(defun russian-lower-case-p (char)
  (position char абвгдеёжзийклмнопрстуфхцчшщъыьэюя""))

(defun whitespace-char-p (char)
  (member char '(#\Space #\Tab #\Newline)))

(defun check-word (word)
  (let ((lower nil))
  (loop for c across word do (if (or (lower-case-p c)
   (russian-lower-case-p c))
    (setf lower T)))
  lower))

(defun word-transform (word)
  (if (check-word word)
            (concatenate 'string word ",")
            word))

(defun sentence-traverse (sentence)
  (let ((new-sentence "") (word ""))
    (loop for c across sentence do
      (if (whitespace-char-p c)
```

```
        (progn
          (setf new-sentence (concatenate 'string new-sentence
   (word-transform word)))
          (setf new-sentence (concatenate 'string new-sentence
   (string c)))
          (setf word "")
        )
        (setf word (concatenate 'string word (string c))))
    )
  (setf new-sentence (concatenate 'string new-sentence
   (word-transform word)))
  new-sentence))

(defun text-traverse (text)
  (let ((new-text '()))
    (dolist (sentence text)
      (setf new-text
        (append new-text
          (list
            (sentence-traverse sentence)))))
    new-text))


;(print (text-traverse '("HELLO      world." ПРИВЕТ" мир")))
```

8.2. Результаты работы

```
* (text-traverse '("HELLO world.ПРИВЕТ мир"))
("HELLO    world.,ПРИВЕТ мир,")
```

9. Замечания автора по существу работы

Чтобы выполнить работу, нужно было хорошо разобраться с строками и символами в common lisp.

10. Выводы

При выполнении работы я столкнулся с проблеми, что в common lisp есть разные типы массивов строк, т.е. просто строк, массивам можно задать разные параметры – fill-pointer, adjustable и поначалу можно сильно запутаться.