Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет: “Информационные технологии и прикладная математика”
Кафедра: 806 “Вычислительная математика и программирование”

Отчет по лабораторной работе №4

по курсу «Функциональное программирование»

Студент: Полей-Добронравова Амелия Вадимовна

Группа: М8О-307Б, № по списку 16.

Преподаватель: Иванов Д. А., доц. каф. 806

Дата: 08.05.2021

Итоговая оценка:

Подпись преподавателя:

Москва, 2021

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 № 4.43)

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

Функция должна удалить все слова, в которых встречается не более двух различных букв, и вернуть новый текст. Знаки пунктуации считаются частью слова. Сравнение как латинских букв, так и русских должно быть регистро-независимым.

4. Оборудование студента

MacBook Pro (15-inch, Mid 2012), процессор 2,3 GHz Intel Core i7, память 8 ГБ.

5. Программное обеспечение

macOS Mojave 10.14.6

6. Идея, метод, алгоритм

Возьмем из лекций функции для преобразования строк с русскими буквами в верхний и нижний регистр. Нужно будет преобразовать все буквы к одному регистру для сравнения, потому что буквы в верхнем и нижнем регистре считаются разными.

Далее в главной функции remove-two-char-words пропишем итерацию по предложениям, и в них - по словам. Будем образовывать новый текст, собирая списки. Для анализа слов убираем из них знаки препинания ((let ((string (string-right-trim ",.;:?!" (russian-string-downcase word))))))), чтобы они не посчитались за буквы, но в выходной список положим со знаками препинания исходное word. Чтобы во внешнем списке лежал список именно предложений, а не слов, в конце каждого внутреннего цикла проканкатенируем элементы. И при этом на момент формирования списка слов между ними положим по пробелу. Последний пробел перед конкатенацией нужно будет удалить с помощью функции delete_end.

Для проверки того, подходит ли слово, чтобы добавить его в итоговый текст, вызывается функция is_two-char-word, возвращающая 0, если в слове использовано больше двух букв, и 1 иначе.

В функции is_two-char-word вызывается collect-letter-counts, образующая хеш-таблицу, хранящая количество встречаемости каждой буквы в слове, и проверяется размер хеш-таблицы. В зависимости от размера возвращается нужное значение.

7. Сценарий выполнения работы

Изучить строки в Common Lisp, продумать логику и написать программу.

8. Распечатка программы и её результаты

Программа

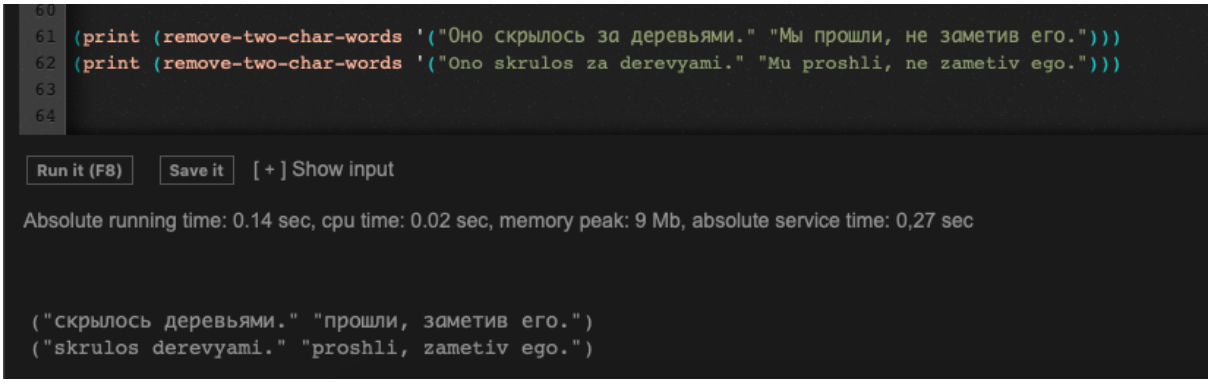
```
(defun russian-upper-case-p (char)
  (position char "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"))


(defun russian-char-downcase (char)
  (let ((i (russian-upper-case-p char)))
    (if i
        (char "абвгдеёжзийклмнопрстуфхцчшщъыьэюя" i)
        (char-downcase char))))


(defun russian-string-downcase (string)
  ;; Преобразовать и латинские, и русские буквы строки в строчные
  (map 'string #'russian-char-downcase string))


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


(defun word-list (string)
  ;; Разбить строки на слова, разделённые знаками whitespace
  ;; A la (split-seq-if #'whitespace-char-p string)
```

```
  (loop with len = (length string)
        for left = 0 then (1+ right)
        for right = (or (position-if #'whitespace-char-p string
                                     :start left)
                        len)
        unless (= right left)  ; исключить пустые слова
          collect (subseq string left right)
        while (< right len)))

(defun collect-letter-counts (word)
  (let ((ht (make-hash-table :test #'equal)))
      (loop for i upfrom 0 below (length word) do
      (incf (gethash (char word i) ht 0)))
    ht))

(defun delete_end(list)
  (cond ((null (cdr list))
     nil)
    (t
     (cons (car list)
           (delete_end (cdr list))))))

(defun is_two-char-word (word)
      (let ((answer 0))
            (if (>= 2 (hash-table-count (collect-letter-counts word)))
                  (setf answer 1))
      answer))

(defun remove-two-char-words(text)
      (loop for sentence in text
```

```
      collect (let ((predloz ()))

      (dolist (word (word-list sentence))

            (let ((string (string-right-trim ",.;:?!"
(russian-string-downcase word))))

                        (when (< 0 (length string))

                              (when (= 0 (is_two-char-word
(russian-string-downcase string)))

                                    (setq predloz (append predloz (list
word)))

                                    (setq predloz (append predloz (list "
"))))))))

            (setq predloz (delete_end predloz)) ;;убрать последний пробел

            (apply #'concatenate 'string predloz))))
```

```
(print (remove-two-char-words '("Оно скрылось за деревьями." "Мы прошли, не заметив его.")))

(print (remove-two-char-words '("Ono skrulos za derevyami." "Mu proshli, ne zametiv ego.")))
```

Результаты



9. Дневник отладки

№ Дата, время Событие Действие по исправлению Примечание

10. Замечания автора по существу работы

Основная сложность на мой взгляд была в сохранении обработанных значений, а не самой обработке.

11. Выводы

В Common Lisp очень удобна работа с переводом разных типов данных в строку и наоборот. При этом встроенных функций обработки довольно мало, например разделения строки по символу нет, хотя это одна из самых нужных функций при работе с естественным языком.