

ЛАБОРАТОРНАЯ РАБОТА 5

ФУНКЦИОНАЛЫ

1. Цель и задачи.

Целью лабораторной работы является изучение отображающих и применяющих функционалов.

2. Краткие теоретические сведения.

2.1 Применяющие функционалы.

В Лиспе определено три применяющих функционала¹:

- (FUNCALL <функциональный аргумент> аргументы) – имеется в Common Lisp и muLISP;
- (APPLY <функциональный аргумент> список аргументов) – в Common Lisp, muLISP и newLISP;
- (EVAL <любое лисповское выражение>) – в Common Lisp, muLISP и newLISP.

В качестве функционального аргумента можно использовать имя функции, lambda-вызов или лисповское выражение, значением которого является имя функции или lambda-вызов.

2.2 Отображающие функционалы.

В Common Lisp и muLISP определено шесть отображающих функционалов²:

- (MAPCAR <функциональный аргумент> списки);
- (MAPLIST <функциональный аргумент> списки);
- (MAPCAN <функциональный аргумент> списки);
- (MAPCON <функциональный аргумент> списки);
- (MAPC <функциональный аргумент> списки);
- (MAPL <функциональный аргумент> списки).

Основным назначением этих функционалов является отображение аргументов в новую последовательность. Перечисленные функционалы можно разделить на две группы: в первую группу включаем функционалы MAPCAR, MAPCAN, MAPC и во вторую группу MAPLIST, MAPCON и MAPL. Первый вид отображающих функционалов отображает функциональный аргумент отдельно на каждый элемент списка. Второй – на последовательность, состоящую из списков, каждый последующий список представляет собой хвост предыдущего. Результатом повторяющихся вычислений будет список, содержащий результаты отображений.

Примеры.

(MAPCAR '* '(1 2 3 4 5) '(10 20 30 40 50)) – результатом будет: (10 40 90 160 250).

¹ См. [1, С. 239–248; 8; 9].

² См. [1, С. 239–244, 249–257].

Другой пример (Common Lisp): (MAPCAR (lambda (y)(list (print y))) '(tom anny mary)).

В результате вызова MAPCAR здесь, во-первых, в текущий выходной поток будут выведены значения:

```
tom
anny
mary,
```

во-вторых, в качестве значения возвращается список ((tom)(anny)(mary)).

```
(MAPCAR (lambda (var)(if (atom var) var (car var))) list)
```

Результат такого вызова зависит от вида аргумента list: если list представляет собой список атомов, то этот же список и будет возвращен. Если list является списком списков, то возвращается список, содержащий головы подсписков. Если, к примеру, list представляет собой '((first second)(lisp ll) 7 8) то возвращается (first lisp 7 8).

В следующем примере воспользуемся функцией sum:

```
(defun sum (lst)
  (cond
    ((null lst) 0)
    (t (+ (car lst)(sum (cdr lst))))
  )
)
```

```
(MAPLIST 'sum '(10 20 30 40 50)).
```

В результате вызова отображающего функционала MAPLIST получим (150 140 120 90 50) т.е. возвращается список, составленный из результатов применения функции sum сначала ко всему списку, затем к хвосту списка, затем к хвосту хвоста и т.д. до пустого списка.

Количество списков в каждом вызове определяется функциональным аргументом: если это функция одного аргумента, то в вызове функционала должен присутствовать один список, если двух, то два и т.д.

```
(MAPCAR 'LIST '(A S D F)) – возвращает ((A)(S)(D)(F));
```

```
(MAPCAN 'LIST '(A S D F)) – возвращает (A S D F);
```

```
(MAPCAN
  (lambda (var)
    (if (atom var) nil
        (list (list (car var)(caddr var))))
  )
  '(((a)(2) (+ a 2))((s)(3)(- s 1))))
```

Вызов такого функционала возвращает список следующего вида:

```
((a) (+ a 2)) ((s) (- s 1)),
```

в котором элементы попарно объединены в списки.

В newLISP'е определен отображающий функционал map. Он отображает аргументы-списки в новый список применением к одинаково расположенным элементам этих списков функции, представленной первым аргументом.

Примеры:

(map + '(1 2 3) '(50 60 70)) возвращает '(51 62 73)

(map if '(true nil true nil true) '(1 2 3 4 5) '(6 7 8 9 10)) возвращает '(1 7 3 9 5)

3. Задание на лабораторную работу.

Задание 1.

Написать программу обработки текста естественного языка с использованием отображающих функционалов в соответствии с заданием из таблицы. Текст рекомендуется представлять списком списков: каждое предложение – список слов, весь текст – список предложений.

Таблица 1. Вариант задания 1.

Вариант.	Задание.
1,13.	Дан текст. Сделать заглавной первую букву первого слова каждого предложения. Предполагается, что первое слово предложения может как начинаться, так и не начинаться с заглавной буквы.
2,14.	Дан текст. Сделать заглавной каждую букву каждого слова, начинающегося с заглавной буквы.
3,15.	Дан текст. В каждом слове текста заменить заданную букву заданной буквой (сочетанием букв). Пример. Заменяемая буква: “б”, заменяющее сочетание букв: “ку”, слово: “абракадабра”, результат : “акуракадакура”.
4,16.	В каждом слове удалить букву, стоящую между двумя заданными.
5,17.	Сформировать список, информирующий о вхождении заданной буквы в текст в текст в виде ((<0 1 5 2 0>) (<3 0 1 5 2 0 1 0>)...). Цифры указывают количество вхождений буквы в каждое слово предложения.
6,18.	Дан текст. Заменить в каждом предложении все вхождения заданного слова на заданное новое слово.
7,19.	Дан текст. Удалить из каждого слова в каждом предложении все повторяющиеся буквы.
8,20.	Дан текст. В каждом слове каждого предложения для повторяющихся букв произвести следующую замену: повторные вхождения букв удалить, к первому вхождению буквы приписать число вхождений буквы в слово. Пример: '((aaabb cccddd)(eeefggg hhkl)) преобразуется в '((a3b2 c4d3)(e3fg3 h2kl)).
9,21.	Дан текст. В каждом слове вставить после заданного 3-буквенного сочетания заданное 2-буквенное.
10,22.	Дан текст. Вставить заданное новое слово после каждого вхождения другого заданного слова.
11,23.	Дан текст. Записать каждое предложение текста в порядке возрастания количества гласных букв в слове.
12,24.	Дан текст. Переписать каждое предложение, расположив слова в обратном алфавитном порядке.
25.	Написать программу, которая в каждом слове исходного текста меняет местами первую и последнюю буквы.

Задание 2.

Дана фраза русского языка. Написать программу, которая разбивает каждое слово фразы на слоги³. Для выполнения этого и последующего задания в `tuLISP`'е рекомендуется воспользоваться версией интерпретатора `mulisp_2.com`.

Задание 3.

“Язык сплетника”. Есть ключевое слово, например, “сплетня”. Слово переводится на язык сплетника⁴ путем отделения первого слога в переводимом и ключевом слове (например, *сло-во* и *спле-тня*) с последующей перестановкой по определенным правилам:

‘(слово сплетня) преобразуется в ‘(сплево слотня).

Каждое слово преобразуется в пару слов. Первое слово есть конкатенация первого слога ключевого слова и части переводимого слова, оставшейся после отделения от него первого слога. Второе слово есть конкатенация первого слога переводимого слова и части ключевого слова, оставшейся после отделения от него первого слога.

Написать программу перевода предложения русского языка на заданный таким образом “тайный” язык.

Задание 4.

Написать программу в соответствии с заданием из Таблицы 2.

Таблица 2. Варианты задания 3.

Вариант.	Задание.
1,12,14,20.	“Тайные языки”. Используя разработанную по заданию 3 программу, построить программу перевода предложения русского языка на так называемый цыганский жаргон ⁵ , в котором ключевым словом всегда является следующее слово. Если последнее слово остается без пары, то его можно переводить или в себя, или с некоторым заданным вспомогательным ключевым словом, например, “сплетня”.
2,7,15,21.	Написать программу, которая заменяет слова исходного текста номерами их семантических эквивалентов ⁶ по словарю в зависимости от значения трехбуквенного конца слова ⁷ . Если слово содержит менее трех букв, то замену не производить.
3,8,16,22.	Частотный словарь ⁸ . Написать программу, которая по заданному тексту строит список пар: (<слово> <частота встречаемости в тексте>).
4,9,17,23.	Написать программу, исключающую в исходном тексте из каждого слова его окончание по словарю. Словарь окончаний представлять списком строк.
5,6,10,18,24.	Написать программу, которая в исходном тексте заменяет слова, являющиеся значениями Лексических Функций (ЛФ) [6] от других слов того же текста, списками вида: {<символ ЛФ по словарю> <ключевое слово>}. Словарь-справочник ЛФ представлять в виде списка троек: (<ключевое слово> <символьное обозначение ЛФ> <значение ЛФ для ключевого слова>). Пример: (“дождь” “Magn” “ливень”).
11,13,19,25.	Написать программу, которая кодирует исходный текст по методу Юлия Цезаря: каждая буква в каждом слове заменяется на следующую.

³ См. [7].

⁴ См. [2, С. 203–214].

⁵ См. [2, С. 214–216].

⁶ См. [3, С. 176–185].

⁷ См. [4, С. 121–129, 158–168, 240–252].

⁸ См. [5, С. 203].

4. Содержание отчета по лабораторной работе.

Отчет по лабораторной работе должен содержать :

- формулировку цели и задач;
- описание процесса разработки программ;
- выводы по проделанной реализации.

Рекомендуемая литература

1. Хювёнен Э. Мир Лиспа. В 2-х т. Т.1: Введение в язык Лисп и функциональное программирование: пер. с финск. [Электронный ресурс] / Э. Хювёнен, Й. Сеппянен. Режим доступа: <http://www.novsu.ru/file/1004773> (дата обращения: 01.02.2017).
2. Хювёнен Э. Мир Лиспа. В 2-х т. Т.2: Методы и системы программирования: пер. с финск. [Электронный ресурс] / Э. Хювёнен, Й. Сеппянен. Режим доступа: <http://www.novsu.ru/file/1004777> (дата обращения: 01.02.2017).
3. Белоногов Г.Г. Автоматизированные информационные системы [Электронный ресурс] / Г.Г. Белоногов, В.И. Богатырёв. Режим доступа: <http://www.novsu.ru/file/1261440> (дата обращения: 06.02.2017).
4. Белоногов Г.Г. Автоматизация процессов накопления, поиска и обобщения информации [Электронный ресурс] / Г.Г. Белоногов, А.П. Новосёлов. Режим доступа: <http://www.novsu.ru/file/1261441> (дата обращения: 06.02.2017).
5. Вирт Н. Алгоритмы + структуры данных = программы: пер. с англ. [Электронный ресурс] / Н. Вирт. Режим доступа: <http://www.novsu.ru/file/1261442> (дата обращения: 06.02.2017).
6. Национальный корпус русского языка: лексические функции [Электронный ресурс]. Режим доступа: <http://www.ruscorpora.ru/instruction-syntax.html#Лексические> (дата обращения 03.02.2017).
7. Весь школьный курс русского языка – Слог. Слогораздел [Электронный ресурс]. Режим доступа: <http://www.pshelp.narod.ru/lib/applicant/0002-007.html> (дата обращения: 02.02.2017).
8. newLISP™ for Mac OS X, GNU Linux, Unix and Windows [Электронный ресурс]. Режим доступа: <http://www.newlisp.org> (дата обращения 03.02.2017).
9. GNU CLISP – an ANSI Common Lisp Implementation [Электронный ресурс]. Режим доступа: <http://www.clisp.org> (дата обращения 03.02.2017).