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»

Отчет по лабораторной работе № 5 по курсу «Функциональное и логическое программирование»

Тема Использование функционалов

Студент Виноградов А. О.

Группа ИУ7-66Б

Оценка (баллы)

Преподаватели Толпинская Н. Б., Строганов Ю. В.

Москва — 2023 г.

Введение

Цель работы: приобрести навыки использования функционалов.

Задачи работы: изучить работу и методы использования применяющих и отображающих функционалов: apply, funcall, mapcar, maplist.

1 Практические задания

1.1 Напишите функцию, которая уменьшает на 10 все числа из списка-аргумента этой функции, проходя по верхнему уровню списковых ячеек. (* Список смешанный структурированный)

```
(defun f1 (lst)
  (
    mapcar #'(lambda (elem)
        (cond ((numberp elem) (- elem 10))
            (T elem))
        ) lst
  )
)
```

1.2 Написать функцию которая получает как аргумент список чисел, а возвращает список квадратов этих чисел в том же порядке.

```
(defun f2 (lst)
  (
    mapcar #'(lambda (elem)
        (* elem elem)
        ) lst
  )
)
```

1.3 Напишите функцию, которая умножает на заданное число-аргумент все числа из заданного списка-аргумента

а) все элементы списка — числа, б) элементы списка – любые объекты.

```
(defun f3a (num lst)
  (
    mapcar #'(lambda (elem)
      (* num elem)
    ) lst
  )
)
```

```
(defun f3b (num lst)
  (
    mapcar #'(lambda (elem)
      (cond
        ((numberp elem) (* num elem))
        (t elem)
      )
    ) lst
  )
)
```

1.4 Написать функцию, которая по своему списку-аргументу lst определяет является ли он палиндромом (то есть равны ли lst и (reverse lst)), для одноуровнего смешанного списка.

```
(defun f4 (lst)
  (if
    (
      find-if #'evenp
          (mapcar #'(lambda (elem reverse-elem)
              (cond ((eql elem reverse-elem) 1)
                  (T 0))
            ) lst (reverse lst)
          )
    )
    Nil
    T
  )
)
```

1.5 Используя функционалы, написать предикат set-equal, который возвращает t, если два его множества-аргумента (одноуровневые списки) содержат одни и те же элементы, порядок которых не имеет значения.

```
(defun set-equal (set1 set2)
  (
    and (eq (length set1) (length set2))
        (every #'(lambda (elem)
            (member elem set2 :test #'equal)
        ) set1)
        (every #'(lambda (elem)
            (member elem set1 :test #'equal)
        ) set2)
  )
)
```

1.6 Напишите функцию, select-between, которая из списка-аргумента, содержащего только числа, выбирает только те, которые расположены между двумя указанными числами - границами-аргументами и возвращает их в виде списка.

```
(defun select-between (lst num1 num2)
  (
    mapcan #'(lambda (elem)
      (if (or (< num1 elem num2) (> num1 elem num2))
          (list elem)
          ()
      )
    ) lst
  )
)
```

1.7 Написать функцию, вычисляющую декартово произведение двух своих списков аргументов.

Напомним, что A x B это множество всевозможных пар (a b), где a принадлежит A, принадлежит B.

```
(defun f7 (lst1 lst2)
  (
    mapcan #'(lambda (elem1)
      (mapcar #'(lambda (elem2)
        (list elem1 elem2)
      ) lst2)
    ) lst1
  )
)
```

1.8 Почему так реализовано reduce, в чем причина?

```
1 (reduce #'+ ()) -> 0
2 (reduce #'* ()) -> 1
```

Если список пуст и не задано начальное значение, вызывается функция без аргументов.

(+) -> 0

(*) -> 1

1.9 Пусть list-of-list список, состоящий из списков. Написать функцию, которая вычисляет сумму длин всех элементов list-of-list (количество атомов)

Например для аргумента ((1 2) (3 4)) -> 4

```
1  (defun f9 (lst)
2    (
3      reduce #'+
4        (
5          mapcar #'(lambda (elem)
6            (if (atom elem) 1 (f9 elem))
7          )lst
8        )
9    )
10 )
```