Государственное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н. Э. Баумана»

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчет по лабораторной работе №10

по курсу ФиЛП

Студент _____ Якубаускайте М.А.
(Подпись, дата)

Преподаватель _____ Толпинская Н.Б.
(Подпись, дата)

Москва 2019

Практическая часть

6.7

Пусть list-of-list список, состоящий из списков. Написать функцию, которая вычисляет сумму длин всех элементов list-of-list, т.е. например для аргумента ((1 2) (3 4)) -> 4

```
(defun sum_lengths (list-of-lists);
    (reduce #'+
    (mapcar (lambda (x)
        (if (listp x) (sum_lengths x) 1)
        ) list-of-lists))
)
```

6.8

Написать рекурсивную версию (с именем rec-add) вычисления суммы чисел заданного списка. Например: (rec-add (2 4 6)) -> 12

```
(defun rec_add_inner (lst sum)
    (let (
        (head (car lst))
            (tail (cdr lst)))
                (cond ((null lst) sum)
                ((listp head) (rec_add_inner tail (rec_add_inner head
                    sum)) )
                ((numberp head) (rec_add_inner tail (+ sum head)) )
            (t (rec_add_inner tail sum))
        )
    )
)
(defun rec_add (lst)
    (if (eq lst nil)
    nil
    (rec_add_inner lst 0))
)
```

6.9

Написать рекурсивную версию с именем rec-nth функции nth.

```
(defun rec_nth_inner (elem curr target)
    (cond ((= curr target) (car elem))
        ((eq elem nil) nil)
        (t (rec_nth_inner (cdr elem) (+ curr 1) target))))
(defun rec_nth (num lst)(rec_nth_inner lst 0 num))
```

6.10

Написать рекурсивную функцию alloddr, которая возвращает t , когда все элементы списка нечетные.

```
(defun alloddr (lst);
    (let ((head (car lst))
        (tail (cdr lst))
        )
            (cond ((null lst) t)
            ((listp head)
            (and (alloddr head) (alloddr tail))
            )
        ((not (numberp head)) nil)
        ((evenp head) nil)
        (t (alloddr tail))
        )
    )
)
```

6.11

Написать рекурсивную функцию, относящуюся к хвостовой рекурсии с одним тестом завершения, которая возвращает последний элемент списка-аргумента.

```
(defun mylast (curr)
    (if (eq (cdr curr) nil)
    (car curr)
    (mylast (cdr curr))
    )
)
```

6.12

Написать рекурсивную функцию, относящуюся к дополняемой рекурсии с одним тестом завершения, которая вычисляет сумму всех чисел от 0 до n-аргумента функции от n-аргумента функции до последнего >= 0

```
(defun get_n_sum (curr n)
    (if (or (eq curr nil) (= n 0))
    0
    (+ (car curr) (get_n_sum (cdr curr) (- n 1)))))
```

6.13

Написать рекурсивную функцию, которая возвращает последнее нечетное число из числового списка, возможно создавая некоторые вспомогательные функции.

```
(defun get_last_odd_inner (curr value)
    (cond ((eq curr nil) value)
        ((oddp (car curr)) (get_last_odd_inner (cdr curr) (car curr)))
        (t (get_last_odd_inner (cdr curr) value))
    )
)
(defun get_last_odd (lst)
    (get_last_odd_inner lst nil)
)
```

6.14

Используя cons-дополняемуюрекурсию с одним тестом завершения, написать функцию которая получает как аргумент список чисел, а возвращает список квадратов этих чисел в том же порядке.

```
(defun square_all (lst);
    (mapcar #'(lambda (x)
        (cond ((numberp x)  x x))((listp x) (square_allx))(tx)))lst))
```

6.15

Написать функцию с именем select-odd, которая из заданного списка выбирает все нечетные числа.

```
(defun select_odd_inner (lst result)
    (mapcar #'(lambda (x)
        (cond ((listp x) (select_odd_inner x result))
            ((and (numberp x) (oddp x))
            (nconc result (cons x nil))
            )
            )
        )
    lst
    )
    (cdr result)
    )
(defun select_odd (lst)
    (select_odd_inner lst (cons nil nil))
)
```