|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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» (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 10 |

**Дисциплина:** Функциональное и логическое программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-62Б |  |  | Е.В. Брянская |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Н.Б.Толпинская |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2021

**Задание 8**

Написать рекурсивную версию (с именем rec-add) вычисления суммы чисел заданного списка

(defun sum-list (lst sum)

(cond ((null lst) sum)

((numberp (car lst))(sum-list (cdr lst) (+ sum (car lst))))

((listp (car lst))(sum-list (cdr lst) (sum-list (car lst) sum)))

(t (sum-list (cdr lst) sum))

)

)

(defun rec-add(lst)

(if (null lst)

nil

(sum-list lst 0))

)

**Задание 9**

Написать рекурсивную версию с именем rec-nth функции nth.

(defun rec-nth (n lst)

(cond ((< n 0) nil)

((= n 0)(car lst))

(t (rec-nth (- n 1) (cdr lst)))

)

)

**Задание 10**

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

(defun alloddr (lst)

(cond ((null lst) t)

((listp (car lst))(and (alloddr (car lst)) (alloddr (cdr lst))))

((not (numberp (car lst))) nil)

((evenp (car lst)) nil)

(t (alloddr (cdr lst))))

)

**Задание 11**

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

(defun my-last (lst)

(cond ((null (cdr lst)) lst)

(t (my-last (cdr lst)))

)

)

**Задание 12**

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

(defun sum-n (lst n)

(cond ((or (null lst) (< n 0)) 0)

(t (+ (car lst)

(sum-n (cdr lst) (- n 1)))))

)

Вариант:

1. от n-ого аргумента функции до последнего >= 0

(defun sum-n-last (lst n)

(cond ((or (null lst) (< (car lst) 0)) 0)

((> n 0) (sum-n-last (cdr lst) (- n 1)))

(t (+ (car lst) (sum-n-last (cdr lst) (- n 1)))))

)

1. от n-ого аргумента функции до m-ого аргумента с шагом d

(defun sum-to-from (lst n m d)

(cond ((or (null lst) (> n m)) 0)

((> n 0) (sum-to-from (cdr lst) (- n 1) (- m 1) d))

(t (+ (car lst) (sum-to-from (cdr lst) (- d 1) (- m 1) d))))

)

**Задание 13**

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

(defun last-odd (lst elem)

(cond ((null lst) elem)

((oddp (car lst)) (last-odd (cdr lst) (car lst)))

(t (last-odd (cdr lst) elem)))

)

(defun last-odd-elem (lst)

(last-odd lst nil)

)

**Задание 14**

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

(defun make-square (lst)

(cond ((null lst) nil)

(t (cons (\* (car lst) (car lst)) (make-square (cdr lst)))))

)

**Задание 15**

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

(defun select-odd (lst)

(cond ((null lst) nil)

((oddp (car lst))(cons (car lst) (select-odd (cdr lst))))

(t (select-odd (cdr lst))))

)

(Вариант 1: select-even

(defun select-even (lst)

(cond ((null lst) nil)

((evenp (car lst))(cons (car lst) (select-odd (cdr lst))))

(t (select-odd (cdr lst))))

)

Вариант 2: вычисляет сумму всех нечетных чисел (sum-all-odd) или сумму всех четных чисел (sum-all-even) из заданного списка)

(defun sum-all-odd (lst)

(cond ((null lst) 0)

((oddp (car lst))(+ (car lst) (sum-all-odd (cdr lst))))

(t (sum-all-odd (cdr lst))))

)

**Дополнительное задание**

Создать и обработать смешанный структурированный список с информацией: ФИО, зарплата, возраст, категория (квалификация). Изменить зарплату, в зависимости от заданного условия, и подсчитать суммарную зарплату. Использовать композиции функций.