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»

Отчёт
к лабораторной работе № 10
По курсу: «Функциональное и логическое программирование»

Студент Прохорова Л. А.

Группа ИУ7-63Б

Оценка (баллы) _____

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

Москва.
2021 г.

Задание 15. Написать функцию с именем select-odd, которая из заданного списка выбирает все нечетные числа.
Вариант 1: select-even,
вариант 2: вычисляет сумму всех нечетных чисел(sum-all-odd) или сумму всех четных чисел (sum-all-even) из заданного списка.

Выбирает все нечётные числа из заданного списка

С использованием функционалов

```
(defun select-odd-2(lst)
   (reduce
      #'(lambda (x y) (if (oddp x) (cons x y) y))
      lst :initial-value Nil :from-end t
   )
)
```

(select-odd-2 '(1 2 3))->(1 3)
(select-odd-2 '(2 4 6))->NIL

С использованием рекурсии

```
(defun my_reverse(lst)
   (reduce
      #'(lambda (res tmp)
         (cons tmp res)
      ) lst :initial-value nil
   )
)

(defun select-odd-rec(lst result)
   (cond
      ((null lst) (my_reverse result))
      (t (select-odd-rec (cdr lst) (if (oddp (car lst)) (cons (car lst) result) result))
      )
   )
)
```

```
(defun select-odd-3(lst) (select-odd-rec lst Nil))
```

(select-odd-3 '(1 2 4 3 5))->(1 3 5)
(select-odd-3 ‘(4 4 2))->Nil

Посчитать сумму всех нечётных элементов

```
(defun sum-all-odd-h(lst res)
   ( cond
      ((null lst) res)
      (
         (and (numberp (car lst)) (oddp (car lst)))
         (sum-all-odd-h (cdr lst) (+ res (car lst)))
      )

      (
         (listp (car lst))
         (sum-all-odd-h (cdr lst)(sum-all-odd-h (car lst) res))
      )
      (t (sum-all-odd-h (cdr lst) res))
   )
)
```

Тестирование
(sum-all-odd-h '(1 2 3 4 5) 0)->9
(sum-all-odd-h '(2 4 6 8) 0)->0

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.

```
(defun create-data (lst surname name patronymic salary age skill)
   (cons (list surname name patronymic salary age skill) lst)
)

(defun change-salary-all(lst param)
   (mapcar
      #'(lambda (x)
         (setf (cadddr x)(* param (cadddr x)))
      ) lst
   )
)

(defun change-salary-person (lst surname name patronymic param)
   (mapcar
      #'(lambda (x)
         (if
            (and
               (equal surname (car x))
               (equal name (cadr x))
               (equal patronymic (caddr x))
            )
            (setf (caddr x) (* param (cadddr x)))
         )
      ) lst

   )
)


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