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 №3

по курсу «Функциональное и логическое программирование»

на тему: «Работа интерпретатора Lisp»

Студент ИУ7-61Б _____ Савинова М. Г.
(Группа) (Подпись, дата) (Фамилия И. О.)

Преподаватель _____ Толпинская Н. Б.
(Подпись, дата) (Фамилия И. О.)

Преподаватель _____ Строганов Ю.В.
(Подпись, дата) (Фамилия И. О.)

2024 г.

СОДЕРЖАНИЕ

1 Практические задания

1.1 Задание 1

Написать функцию, которая принимает целое число и возвращает первое четное число, не меньшее аргумента.

```
(defun f (x)
    (cond ((oddp x) (+ x 1))
          (t x              ) ) )

(print (f 1)) ;; 2
(print (f 4)) ;; 4
```

1.2 Задание 2

Написать функцию, которая принимает число и возвращает число того же знака, но с модулем на 1 больше модуля аргумента.

```
(defun f (x)
    (cond ((> x 0) (+ 1 x))
          (t (+ x -1)     ) ) )

(print (f -1)) ;; -2
(print (f 1))  ;; 2
```

1.3 Задание 3

Написать функцию, которая принимает два числа и возвращает список из этих чисел, расположенных по возрастанию.

```
(defun f (x y)
    (cond ((< x y) (cons x (cons y Nil)))
          (t (cons y (cons x Nil))      ) ) )

(print (f 1 2)) ;; 1 2
(print (f 2 1)) ;; 2 1
```

1.4 Задание 4

Написать функцию, которая принимает три числа и возвращает T только тогда, когда первое число расположено между вторым и третьим.

```
(defun f (x y z)
    (cond ((> x y) (< x z))
          ((< z x) (< x y)) ) )

(print (f 0 -1 1)) ;; T
(print (f 0 1 -1)) ;; T
(print (f 0 0 0))  ;; Nil
(print (f 1 1 0))  ;; Nil
(print (f 0 1 0))  ;; Nil
```

1.5 Задание 5

Каков результат вычисления следующих выражений?

```
(print (and 'fee 'fie 'foe)         ) ;; FOE
(print (or nil 'fie 'foe)           ) ;; FIE
(print (and (equal 'abc 'abc) 'yes)) ;; YES

(print (or 'fee 'fie 'foe)          ) ;; FEE
(print (and nil 'fie 'foe)          ) ;; Nil
(print (or (equal 'abc 'abc) 'yes) ) ;; T
```

1.6 Задание 6

Написать предикат, который принимает два числа-аргумента и возвращает T, если первое число не меньше второго.

```
(defun f (a b)
    (cond ((< a b) Nil)
          (t t         ) ) )

(print (f 1 2)) ;; Nil
(print (f 2 2)) ;; T
(print (f 3 2)) ;; T
```

1.7 Задание 7

Какой вариант из следующих двух вариантов предиката ошибочен и почему?

```
(defun pred1 (x)
    (and (numberp x) (plusp x)) )

(defun pred2 (x)
    (and (plusp x) (numberp x)) )

(print (pred1 1))       ;; T
(print (pred2 1))       ;; T

(print (pred1 "1"))     ;; Nil
(print (pred2 "1"))     ;; ошибка
```

Ошибочным является `pred2`, так как сначала происходит проверка на положительность аргумента, а только потом является ли его аргумент числовым атомом.

1.8 Задание 8

Решить задачу 4, используя для ее решения конструкции: только IF, только COND, только AND/OR.

```
(defun f_cond (x y z)
    (cond ((> x y) (< x z))
          ((> x z) (< x y)) ) )

(defun f_if (x y z)
    (if (< y x)
        (if (< x z) t Nil)
    (if (< x y)
        (if (< z x) t Nil)) ) )

(defun f_and_or (x y z)
    (or (and (< y x) (< x z))
        (and (< z x) (< x y))) )
```

1.9 Задание 9

Переписать функцию `how-alike`, приведенную в лекции и использующую COND, используя только конструкции IF, AND/OR.

```
;; (defun how_alike_old (x y)
;;          (cond ((or (= x y) (equal x y)) 'the_same)
;;                ((and (oddp x) (oddp y)) 'both_odd)
;;                ((and (evenp x) (evenp y)) 'both_even)
;;                (t 'difference) ) )

(defun how_alike (x y)
        (if (or (= x y) (equal x y))
            'the_same
        (if (and (oddp x) (oddp y))
            'both_odd
        (if (and (evenp x) (evenp y))
            'both_even
        'difference) ) ) )

(print (how_alike 1 3)) ;; BOTH_ODD
(print (how_alike 2 3)) ;; DIFFERENCE
(print (how_alike 2 4)) ;; BOTH_EVEN
(print (how_alike 2 2)) ;; THE_SAME
```