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 дисциплине
"Функциональное и логическое программирование"

Тема Работа интерпретатора Lisp

Студент Малышев И. А.

Группа ИУ7-61Б

Оценка (баллы) _____

Преподаватель: Толпинская Н. Б.

Москва — 2022 г.

Теоретические вопросы

1. Базис Lisp

Базис языка представлен:

- структурами и атомами;
- функциями;

Функция – правило, по которому каждому значению одного или нескольких аргументов ставится в соответствие конкретное значение результата.

Функции, входящие в базис языка: atom, eq, cons, car, cdr, cond, quote, lambda, eval, apply, funcall.

2. Классификация функций

- "чистая"функция – соответствует математической функции;
- специальные функции или формы – могут принимать несколько аргументов;
- псевдофункции – создают «эффект» – отображение на экране процесса обработки данных и т.п.;
- функции с вариантами значения;
- функционалы – принимают в качестве аргумента функцию или возвращают функцию;
- базисные функции (cons, car, cdr, atom, cond, eq, quote, eval, lambda, apply, funcall).

3. Способы создание функций

Определение функций пользователя в Lisp-е возможно двумя способами:

- с использованием Лямбда-нотаци (функции без имени): (lambda (<аргументы>) (<тело>));
- с использованием макро определения DEFUN: (defun <имя> (<аргументы>) (<тело>)).

4. Работа функций Cond, if, and/or

Сигнатура функции **cond**:
(cond (предикат-1 действие-1))
(предикат-2 действие-2)
...
(предикат-n действие-n)

Работа функции **cond**:
сначала просматриваются все предикаты в порядке следования, и если хоть один из них истинный, то cond возвращает результат, связанный с этим предикатом. Если ни один предикат не был истинным, то она вернет Nil.

Сигнатура функции **if**:
(if условие выражение-1 выражение-2)

Работа функции **if**:
если условие истинно (T), то выполняется выражение-1, иначе (Nil) – выражение-2

Сигнатура функции **and**:
(and выражение-1 выражение-2 ... выражение-n)

Работа функции **and**:
результат функции будет истинным, если все ее выражения истинны. В таком случае в качестве результата вернется значение выражения-n. В случае, если хотя бы одно выражение ложно (Nil), вычисление последующих выражений не производится и результатом функции является Nil.

Сигнатура функции **or**:
(or выражение-1 выражение-2 ... выражение-n)

Работа функции **or**:
результат функции будет ложным (Nil), если все ее выражения ложны. В случае, если хотя бы одно выражение истинно, вычисление последующих выражений не производится и результатом функции является значения выражения, которое первым в списке аргументов дало в результате истину.

Практические задания

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

Листинг 1: Решение задания №1

```
1 (defun first-even (x) (if (evenp x) x (+ x 1)))
```

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

Листинг 2: Решение задания №2

```
1 (defun abs-plus-one (x) (if (> x 0) (+ x 1) (- x 1)))
```

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

Листинг 3: Решение задания №3

```
1 (defun sorted-pair-list (fst snd) (if (> fst snd) (list fst snd) (list
     snd fst)))
```

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

Листинг 4: Решение задания №4

```
1 (defun between (a b c) (if (and (> a b) (< a c)) T Nil))
```

5. Каков результат вычисления следующих выражений?

(and 'fee 'fie 'foe) -> foe
(or 'fee 'fie 'foe) -> fee

(or nil 'fie 'foe) -> fie
(and nil 'fie 'foe) -> NIL
(and (equal 'abc 'abc) 'yes) -> yes
(or (equal 'abc 'abc) 'yes) -> T

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

Листинг 5: Решение задания №6

```
1 (defun ge (a b) (if (>= a b) T Nil))
```

7. Какой из следующих двух вариантов предиката ошибочен и почему?

(defun pred1 (x) (and (numberp x) (plusp x))) – ok
(defun pred2 (x) (and (plusp x)(numberp x))) – runtime error

Второй вариант ошибочен, т.к. если в функцию будет передано не число и на него будет применена функция **plusp** (которая работает только с числовыми значениями), интерпретатор выдаст ошибку.

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

Листинг 6: Решение задания №8

```
1 (defun between-if (a b c) (if (and (> a b) (< a c)) T Nil))
2
3 (defun between-cond (a b c) (cond ((and (> a b) (< a c)) T) (T Nil)))
4
5 (defun between-and (a b c) (and (> a b) (< a c)))
```

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

Листинг 7: Решение задания №9

```
1 (defun how-alike-cond (x y)
2   (cond ((or (= x y) (equal x y)) 'the_same)
3     ((and (oddp x) (oddp y)) 'both_odd)
4     ((and (evenp x) (evenp y)) 'both_even)
5     (T 'difference)))
6
7
8
```

```
(defun how-alike-if (x y)
  (if (if (= x y) (equal x y)) 'the_same
  (if (if (oddp x) (oddp y)) 'both_odd
  (if (if (evenp x) (evenp y)) 'both_even
  'difference))))

(defun how-alike-andor (x y)
  (or (and (or (= x y) (equal x y)) 'the_same)
    (and (oddp x) (oddp y) 'both_odd)
    (and (evenp x) (evenp y) 'both_even)
    'difference))
```