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 №5, 6, 7, 8

По курсу «Функциональное и логическое программирование»

Студент: Киселев А.М.

Группа: ИУ7-66

Преподаватель: Толпинская Н.Б.

Москва 2019

Содержание

1 Выполнение работы

1.1 Построенная база знаний

Листинг 1.1 — Получение элементов списка с помощью команд car и cdr.

```

predicates
    max2(integer, integer, integer)
    max3(integer, integer, integer, integer)

clauses
    max2(X, Y, X) :- X >= Y, !.
    max2(\_, Y, Y).

    max3(X, Y, Z, X) :- X >= Y, X >= Z, !.
    max3(\_, Y, Z, Y) :- Y >= Z, !.
    max3(\_, \_, Z, Z).

goal
    %max2(1, 3, Z).
    max3(4, 3, 2, P).
```

Листинг 1.2 — Получение элементов списка с помощью команд car и cdr.

```
predicates
    factorial(integer)
    factorial(integer, integer)

    fibonacci(integer)
    fibonacci(integer, integer)

clauses
    factorial(1, X) :-
        X = 1.
    factorial(N, X) :-
        N\_1 = N - 1,
        factorial(N\_1, X1),
        X = X1 * N.
    factorial(N) :-
        factorial(N, X),
        write(X).
```

```
18
19      fibonacci(1, 1) :-
20          !.
21      fibonacci(2, 1) :-
22          !.
23      fibonacci(N, X) :-
24          N\_1 = N - 1,
25          N\_2 = N - 2,
26          fibonacci(N\_1, I1),
27          fibonacci(N\_2, I2),
28          X = I1 + I2.
29      fibonacci(N) :-
30          fibonacci(N, X),
31          write(X).
32
33  goal
34      fibonacci(5).
```

Листинг 1.3 — Получение элементов списка с помощью команд car и cdr.

```
1
2   predicates
3   domains
4       Number = integer
5       NList = Number*
6   predicates
7       len(NList, Number)
8
9       length(NList, Number)
10      length(NList, Number, Number)
11
12      listSum(NList, integer)
13      deleteEl(NList, integer, NList)
14      deleteEls(NList, integer, NList)
15
16      /* Bubble sort */
17      permutation(NList, NList)
18      bubble(NList, NList)
19      /* Bubble sort engds*/
20      makeSet(NList, NList)
21      makeSet(NList, integer, NList)
```

```
22
23      makeListGreaterThanEl(NList, integer, NList)
24
25      even(integer)
26      makeListWithEvenPos(NList, NList).
27      makeListWithEvenPos(NList, integer, NList).
28
29      mergeLists(NList, NList, NList)
30      merge(NList, Nlist, NList)
31
32  clauses
33      len([], 0) :-
34          !.
35      len([\_|Tail], X) :-
36          len(Tail, X1),
37          X = X1 + 1,
38          !.
39
40      length(List, X) :-
41          length(List, 0, X),
42          !.
43      length([], Count, Count) :-
44          !.
45      length([\_|Tail], Count, X) :-
46          NewCount = Count + 1,
47          length(Tail, NewCount, X).
48
49
50      listSum([Head|[]], Head) :-
51          !.
52      listSum([Head|Tail], X) :-
53          listSum(Tail, X1),
54          X = Head + X1,
55          !.
56
57
58      deleteEl([], \_, []) :-
59          !.
60      deleteEl([El|Tail], El, Tail) :-
61          !.
62      deleteEl([Head|Tail], El, [Head|X]) :-
```

```
63          deleteEl(Tail, El, X).
64
65
66      deleteEls([], \_, []) :-
67          !.
68      deleteEls([El|Tail], El, X1) :-
69          deleteEls(Tail, El, X1),
70          !.
71      deleteEls([Head|Tail], El, [Head|X]) :-
72          deleteEls(Tail, El, X).
73
74
75      permutation([X,Y|T],[Y,X|T]) :-
76          X > Y,
77          !.
78      permutation([X|T],[X|T1]) :-
79          permutation(T,T1).
80      bubble(L,L1) :-
81          permutation(L,LL),
82          !,
83          bubble(LL,L1).
84      bubble(L,L).
85
86
87      makeSet([], []) :-
88          !.
89      makeSet(List, X) :-
90          bubble(List, Sorted),
91          Sorted = [Head|Tail],
92          makeSet(Tail, Head, X1),
93          X = [Head|X1],
94          !.
95      makeSet([], \_, []) :-
96          !.
97      makeSet([Head|Tail], Head, X) :-
98          makeSet(Tail, Head, X),
99          !.
100     makeSet([Head|Tail], \_, [Head|X]) :-
101         makeSet(Tail, Head, X),
102         !.
103
```

```
104
105     makeListGreaterThanEl([], \_, []) :-
106         !.
107     makeListGreaterThanEl([Head|Tail], El, X) :-
108         Head > El,
109         makeListGreaterThanEl(Tail, El, X1),
110         X = [Head|X1],
111         !.
112     makeListGreaterThanEl([\_|Tail], El, X) :-
113         makeListGreaterThanEl(Tail, El, X),
114         !.
115
116
117     even(N) :-
118         N mod 2 = 0.
119     makeListWithEvenPos([Head|Tail], [Head|X]) :-
120         Index = 1,
121         makeListWithEvenPos(Tail, Index, X),
122         !.
123     makeListWithEvenPos([], \_, []) :-
124         !.
125     makeListWithEvenPos([Head|Tail], Index, X) :-
126         even(Index),
127         Index1 = Index + 1,
128         makeListWithEvenPos(Tail, Index1, X1),
129         X = [Head|X1],
130         !.
131     makeListWithEvenPos([\_|Tail], Index, X) :-
132         Index1 = Index + 1,
133         makeListWithEvenPos(Tail, Index1, X),
134         !.
135
136     mergeLists(L1, L2, X) :-
137         length(L1, Len1),
138         length(L2, Len2),
139         Len1 < Len2,
140         merge(L1, L2, X),
141         !.
142     mergeLists(L1, L2, X) :-
143         merge(L2, L1, X),
144         !.
```

```
145
146     merge([Head|[]], L2, [Head|L2]) :-
147         !.
148     merge([Head|Tail], L2, [Head|X]) :-
149         merge(Tail, L2, X),
150         !.
151
152 goal
153     %len([1, 2, 3, 4, 5, 6], Z).
154     %length([1, 2, 3, 4, 5, 6, 7], Z).
155     %listSum([2, 2, 2, 8, 2, 2], Z).
156     %deleteEl([1, 2, 2, 3, 4, 3, 5, 6], 3, Z).
157     %deleteEls([3, 1, 2, 2, 3, 4, 3, 5 ,6, 3], 3, Z).
158     %makeSet([5, 5, 6, 3, 3, 3, 9, 10, 1, 1, 0, 5, 10], Set).
159     %makeListGreaterThanEl([5, 3, 6, 99, 7, 9, 2, 0, 5, 3], 3, Z).
160     %makeListWithEvenPos([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10], Z).
161     mergeLists([9, 8, 7, 6],  [1, 2, 3], Z).
```