



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего 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»

Лабораторная работа № 20

Дисциплина	Функциональное и логическое программирование.
Тема	Формирование и модификация списков на Prolog.
Студент	Степанов А. О.
Группа	ИУ7-63Б
Оценка (баллы)	
Преподаватель	Толпинская Н.Б.

Москва, 2020 г.

ЗАДАНИЕ

Используя хвостовую рекурсию, разработать, комментируя аргументы, эффективную программу, позволяющую:

1. Сформировать список из элементов числового списка, больших заданного значения;
2. Сформировать список из элементов, стоящих на нечетных позициях исходного списка (нумерация от 0);
3. Удалить заданный элемент из списка (один или все вхождения);
4. Преобразовать список в множество (можно использовать ранее разработанные процедуры).

Листинг 1: Текст программы

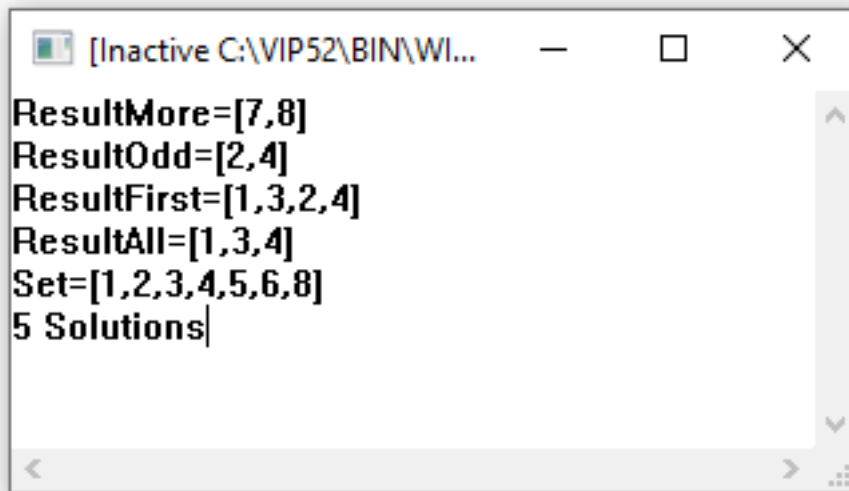
```
1 domains
2     list = integer*.
3     index = integer.
4
5 predicates
6     moreThat(list , integer , list ).
7     listOdd(list , list ).
8     listOdd(list , list , index).
9     removeFirst(list , integer , list ).
10    removeAll(list , integer , list ).
11    set(list , list ).
12
13 clauses
14    moreThat([], _, []) :- !.
15    moreThat([Head|Tail], Number, Result) :-
16        Head > Number,
17        moreThat(Tail, Number, TailResult),
18        Result = [Head|TailResult].
19    moreThat([Head|Tail], Number, Result) :-
20        Head <= Number,
21        moreThat(Tail, Number, Result).
22
23    listOdd(List, Result) :- listOdd(List, Result, 0).
24    listOdd([], [], _) :- !.
```

```

25 listOdd([_|Tail], Result, Index) :-
26     Index mod 2 = 0,
27     NextIndex = Index + 1,
28     listOdd(Tail, Result, NextIndex).
29 listOdd([Head|Tail], Result, Index) :-
30     Index mod 2 = 1,
31     NextIndex = Index + 1,
32     listOdd(Tail, TailResult, NextIndex),
33     Result = [Head|TailResult].
34
35 removeFirst([], _, []) :- !.
36 removeFirst([Head|Tail], Number, Result) :-
37     Head = Number,
38     Result = Tail, !.
39 removeFirst([Head|Tail], Number, Result) :-
40     Head <> Number,
41     removeFirst(Tail, Number, TailResult),
42     Result = [Head|TailResult].
43
44 removeAll([], _, []) :- !.
45 removeAll([Head|Tail], Number, Result) :-
46     Head = Number,
47     removeAll(Tail, Number, Result).
48 removeAll([Head|Tail], Number, Result) :-
49     Head <> Number,
50     removeAll(Tail, Number, TailResult),
51     Result = [Head|TailResult].
52
53 set([], []) :- !.
54 set([Head|Tail], Set) :-
55     removeAll(Tail, Head, NextTail),
56     set(NextTail, TailSet),
57     Set = [Head|TailSet].
58
59 goal
60     moreThat([4, 7, 1, 2, 8], 5, ResultMore);
61     listOdd([1, 2, 3, 4, 5], ResultOdd);
62     removeFirst([1, 2, 3, 2, 4], 2, ResultFirst);
63     removeAll([1, 2, 3, 2, 4], 2, ResultAll);
64     set([1, 2, 3, 4, 1, 2, 5, 2, 1, 2, 6, 8, 2, 6, 8], Set).

```

РЕЗУЛЬТАТ РАБОТЫ



A screenshot of a Windows command window. The title bar shows the file path "C:\VIP52\BIN\WI..." and standard window controls. The output text is as follows:

```
ResultMore=[7,8]
ResultOdd=[2,4]
ResultFirst=[1,3,2,4]
ResultAll=[1,3,4]
Set=[1,2,3,4,5,6,8]
5 Solutions|
```

Рис. 1: Результат работы программы

ФОРМИРОВАНИЕ ОТВЕТА

Для одного из вариантов ВОПРОСА и одного из заданий составить таблицу, отражающую конкретный порядок работы системы: Т.к. резольвента хранится в виде стека, то состояние резольвенты требуется отображать в столбик: вершина – сверху! Новый шаг надо начинать с нового состояния резольвенты! Для каждого запуска алгоритма унификации, требуется указать № выбранного правила и дальнейшие действия – и почему.

Таблица 1: moreThat([4, 7, 1, 2, 8], 5, ResultMore)

№ ша- га	Состояние резольвенты, и вы- вод: дальнейшие действия (по- чему?)	Для каких термов запускается алгоритм унификации: T1=T2 и каков результат (и подстанов- ка)	дальнейшие дей- ствия: прямой ход или откат (почему и к чему приводит?)
1	moreThat([4, 7, 1, 2, 8], 5, ResultMore)	Подстановка: Head = 4, Tail = [7, 1, 2, 8], Number = 5, Result = ResultMore moreThat([4, 7, 1, 2, 8], 5, ResultMore) moreThat([Head Tail], Number, Result)	Прямой ход
2	Head > Number moreThat(Tail, Number, TailResult) Result = [Head TailResult]	4 > 5	Обратный ход
3	moreThat([4, 7, 1, 2, 8], 5, ResultMore)	Подстановка: Head = 4, Tail = [7, 1, 2, 8], Number = 5, Result = ResultMore moreThat([4, 7, 1, 2, 8], 5, ResultMore) moreThat([Head Tail], Number, Result)	Прямой ход
4	Head <= Number moreThat(Tail, Number, Result)	4 <= 5	Прямой ход
5	moreThat(Tail, Number, Result)	Подстановка: Tail = [7, 1, 2, 8], Result = Result moreThat([7, 1, 2, 8], 5, ResultMore) moreThat([Head Tail], Number, Result)	Прямой ход
6	Head > Number	7 > 5	Прямой ход

	moreThat(Tail, Number, TailResult) Result = [Head TailResult]		
7	moreThat(Tail, Number, TailResult) Result = [Head TailResult]	Подстановка: Tail = [1, 2, 8], Number = 5, Result = TailResult	Прямой ход
8	Head > Number moreThat(Tail, Number, TailResult) Result = [Head TailResult] Result = [Head TailResult]	1 > 5	Обратный ход
9	moreThat(Tail, Number, TailResult) Result = [Head TailResult]	Подстановка: Tail = [1, 2, 8], Number = 5, Result = TailResult	Прямой ход
10	Head <= Number moreThat(Tail, Number, Result) Result = [Head TailResult]	1 <= 5	Прямой ход
11	moreThat(Tail, Number, Result) Result = [Head TailResult]	Подстановка: Tail = [2, 8], Number = 5, Result = Result	Прямой ход
12	Head > Number moreThat(Tail, Number, TailResult) Result = [Head TailResult] Result = [Head TailResult]	2 > 5	Обратный ход
13	moreThat(Tail, Number, Result) Result = [Head TailResult]	Подстановка: Tail = [2, 8], Number = 5, Result = Result	Прямой ход
14	Head <= Number moreThat(Tail, Number, Result) Result = [Head TailResult]	2 <= 5	Прямой ход
15	moreThat(Tail, Number, Result) Result = [Head TailResult]	Подстановка: Tail = [8], Number = 5, Result = Result	Прямой ход
16	Head > Number moreThat(Tail, Number, TailResult) Result = [Head TailResult] Result = [Head TailResult]	8 > 5	Прямой ход
17	moreThat(Tail, Number, TailResult) Result = [Head TailResult]	Подстановка: Tail = [], Number = 5, Result = TailResult	Прямой ход

	Result = [Head TailResult]		
18	! Result = [Head TailResult] Result = [Head TailResult]	Отсечение	Прямой ход
19	Result = [Head TailResult] Result = [Head TailResult]	Подстановка: Result = [8]	Прямой ход
20	Result = [Head TailResult]	Подстановка: Result = [7, 8]	
21	Пустая	Результат: MoreResult = [7, 8]	Обратный ход
22	moreThat(Tail, Number, Result)	Подстановка: Tail = [8], Result = Result moreThat([8], 5, ResultMore) moreThat([Head Tail], Number, Result)	Прямой ход
23	Head <= Number moreThat(Tail, Number, Result)	8 <= 5	Обратный ход
24	moreThat(Tail, Number, Result)	Подстановка: Tail = [7, 1, 2, 8], Result = Result moreThat([7, 1, 2, 8], 5, ResultMore) moreThat([Head Tail], Number, Result)	Прямой ход
25	Head <= Number moreThat(Tail, Number, Result)	7 <= 5	Обратный ход

ВЫВОДЫ

Эффективность достигается за счет использования отсечения (!), которое останавливает поиски следующих фактов и правил. Так же за счет использования конструкции [Head|Tail] можно эффективно разделить голову списка от хвоста.

ВОПРОСЫ

1. Как организуется хвостовая рекурсия в Prolog?

В языке Prolog рекурсия организуется при помощи правила, в котором есть обращение к тому же правилу.

2. Какое первое состояние резольвенты?

Первое состояние резольвенты – вопрос.

3. Каким способом можно разделить список на части, какие, требования к частям?

В Prolog используется специальный символ для деления списка на голову и хвост – вертикальная черта `|`. Вертикальную черту можно использовать не только для отделения головы списка, но и для отделения произвольного числа начальных элементов списка.

4. Как выделить за один шаг первые два подряд идущих элемента списка? Как выделить 1-й и 3-й элемент за один шаг?

- `[First, Second|_]` (First – первый элемент списка, Second – второй)
- `[First, _, Third|_]` (First – первый элемент списка, Third – третий)

5. Как формируется новое состояние резольвенты?

Преобразование резольвенты выполняется с помощью редукции.

Редукция – замена цели телом того правила, заголовок которого унифицируется с целью. Новая резольвента получается в два этапа:

- (a) В текущей резольвенте выбирается одна из целей и для неё выполняется редукция \Rightarrow получаем новую конъюнкцию целей (новую резольвенту)
- (b) К полученной новой резольвенте применяется подстановка, как наибольший общий унификатор цели и заголовка правила, сопоставимого с этой целью.

6. Когда останавливается работа системы? Как это определяется на формальном уровне?

Работа системы останавливается в двух случаях:

- Когда встретился символ отсечения (!);
- Когда резольвента осталась пустой (формально не осталось подходящих фактов и правил).