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

по лабораторной работе № 20

Название: Формирование и модификация списков на Prolog

Дисциплина: Функциональное и логическое программирование

Студент ИУ7-65Б _____ Д.В. Сусликов

(Группа) (Подпись, дата) (И.О. Фамилия)

Преподаватель _____ Н.Б. Толпинская

(Подпись, дата) (И.О. Фамилия)

Москва, 2021

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).

Убедиться в правильности результатов.

Для одного из вариантов вопроса и 1-ого задания составить таблицу, отражающую конкретный порядок работы системы

Листинг:

```
domains
    list = integer*

predicates
    more_than(list, integer, list)
    odd_list(list, list)
    del_num_list(list, integer, list)
    get_set(list, list)

clauses
    more_than([], _, []).
    more_than([H|T], Num, [H|ResT]):-
            H > Num,
            more_than(T, Num, ResT), !.
    more_than([_|T], Num, ResT):-
        more_than(T, Num, ResT).
```

```
odd_list([], []).
odd_list([_,H|T], [H|ResT]):-
        odd_list(T, ResT).

del_num_list([], _, []).
del_num_list([H|T], Num, [H|ResT]):-
        H <> Num,
        del_num_list(T, Num, ResT), !.
del_num_list([_|T], Num, ResT):-
        del_num_list(T, Num, ResT).

get_set([], []).
get_set([H|T], [H|ResT]):-
        del_num_list(T, H, TempRes),
        get_set(TempRes, ResT).
```

```
goal
    %more_than([1,10,2,12,3,13], 9, Res).
    %odd_list([1,10,2,12,3,13], Rest).
    %del_num_list([1,2,1,4,1,5,6,1,32,1,23], 1, Res).
    get_set([1,1,2,2,3,3,4,4,5,5,6,6,1,3], Res).
```

Результаты работы:

```
Res=[10,12,13]
1 Solution
```

Пример more_than

```
Rest=[10,12,13]
1 Solution
```

Пример odd_list

```
Res=[2,4,5,6,32,23]
1 Solution
```

Пример del_num_list

```
Res=[1,2,3,4,5,6]
1 Solution
```

Пример get_set

Приведем таблицу для составления нового списка чисел, больше введенного.

more/than([1],9, Res)

Текст процедуры:

```
1:      more_than([], _, []).
2:      more_than([H|T], Num, [H|ResT]):-
                H > Num,
                more_than(T, Num, ResT), !.
3:      more_than([_|T], Num, ResT):-
                more_than(T, Num, ResT).
```

№	Текущая резольвента - ТР	ТЦ, выбираемые правила: сравниваемые термы, подстановка	Дальнейшие действия с комментариями
1	more_than([1], 9, Res)	ТЦ: more_than([1], 9, Res)	Поиск знания с начала БЗ
	more_than([1], 9, Res)	ПР1: [] = [1] _ = 9 [] = Res Неудача	Переход к следующему заголовку БЗ
	more_than([1], 9, Res)	ПР2: [H|T] = [1] Num = 9 [H1|ResT] = Res Успех H = 1 T = [] Num = 9 Res = [1|ResT]	Тело ПР2 заменяют цель в резольвенте
2	1 >9, more_than([], 9, Res), !	Сравнение: 1 >9 Неудача	Откат. Переход к следующему заголовку БЗ
3	more_than([1], 9, Res)	ПР3: [_|T] = [1] Num = 9 Res = Res Успех T = [] Num = 9 Res = ResT	Тело ПР3 заменяет цель в резольвенте

4	more_than([], 9, Res)	ТЦ:more_than([], 9, Res)	Поиск знания с начала БЗ
	more_than([], 9, Res)	ПР1: [] = [] _ = 9 [] = ResT Успех ResT = []	Пустое тело заменяет цель в резольвенте
	Пусто		Успех Res = ResT = [] Откат
5	more_than([], 9, Res)	ПР2: [H|T] = [] Num = 9 [H|ResT] = ResT Неудача	Переход к следующему заголовку БЗ
	more_than([], 9, Res)	ПР3: [_|T] = [] Num = 9 ResT = Res Неудача	Должен включиться откат, но метки в конце процедур, т.е. других альтернатив нет. Завершение работы

Вывод

Эффективность работы системы может быть достигнута за счет хвостовой рекурсии и использования отсечения в тех случаях, где заведомо известна единственность ответа на вопрос.

Ответы на вопросы

1. Как организуется хвостовая рекурсия в Prolog?

   Хвостовая рекурсия: Для ее осуществления рекурсивный вызов определяемого предиката должен быть последней подцелью в теле рекурсивного правила и к моменту рекурсивного вызова не должно остаться точек возврата (непроверенных альтернатив).

2. Какое первое состояние резольвенты?

   Вопрос.

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

   В Prolog существует более общий способ доступа к элементам списка. Для этого используется метод разбиения списка на начало и остаток. Для этого используется вертикальная черта (|) за последним элементом начала.

   Начало списка – это группа первых элементов, не менее одного. Остаток списка – обязательно список (может быть пустой), всегда один.

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

   Два подряд идущих:

   ```
   [First, Second|Tail]
   ```

   1-й и 3-й:

   ```
   [First, _, Third|Tail]
   ```

5. Как формируется новое состояние резольвенты?

   Резольвента - текущая цель, существующая на любой стадии вычислений. Резольвенты порождаются целью и каким-либо правилом или фактом, которые просматриваются последовательно сверху вниз. Если резольвента существует при наиболее общей унификации, она вычисляется. Если пустая резольвента с помощью такой стратегии не найдена, то ответ на вопрос отрицателен.

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

   Когда стек пуст.