|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Лабораторная работа №20*

*По предмету: «Функциональное и логическое программирование»*

Преподаватель: Строганов Ю.В.

Студент: Гасанзаде М.А.,

Группа: ИУ7-66Б

Москва, 2020 г.

# ЗАДАНИЕ

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

* Сформировать список из элементов числового списка, больших заданного значения;
* Сформировать список из элементов, стоящих на нечетных позициях исходного списка (нумерация от 0);
* Удалить заданный элемент из списка (один или все вхождения);
* 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:

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 соответствующий вывод: успех или нет –и почему.

# ВОПРОСЫ

**Как организуется хвостовая рекурсия в Prolog?**

* Рекурсивный вызов единственен и расположен в конце тела правила
* Не должно быть возможности сделать откат до вычисления рекурсивного вызова

**Как организовать выход из рекурсии в Prolog?**

С помощью отсечения.

**Какое первое состояние резольвенты?**

Заданный вопрос (goal).

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

Получить голову или хвост списка можно при унификации списка с [H|T], H – голова списка, T – хвост списка (является списком).

**Как выделить за один шаг первые два подряд идущих элемента списка?**

[H1|[H2|\_]]

**Как выделить 1-й и 3-й элемент за один шаг?**

[H1|[\_|[H3|\_]]]

**Как формируется новое состояние резольвенты?**

Преобразования резольвенты выполняются с помощью редукции. Редукцией цели G с помощью программы P называется замена цели G телом того правила из P, заголовок которого унифицируется с целью. Новая резольвента образуется в два этапа:

1. в текущей резольвенте выбирается одна из подцелей и для неё выполняется редукция;
2. к полученной конъюнкции целей применяется подстановка, полученная как наибольший общий унификатор цели и заголовка сопоставленного с ней правила.

**Когда останавливается работа системы?**

Работа системы останавливается, когда найдены все возможные ответы на вопрос.

**Как это определяется на формальном уровне?**

Когда в резольвенте находится исходный вопрос, для которого пройдена вся БЗ.

# ЛИСТИНГ

domains

list = **integer**\*.

predicates

biggerThan(list, **integer**, list). % список, число, результат

oddElements(list, list). % список, результат

deleteElement(list, **integer**, list).% список, число, результат

createSet(list, list). %исходный список, результат

clauses

biggerThan([], \_, []).

biggerThan([H|T], Min, [H|ResTail]) :-

H > Min,

biggerThan(T, Min, ResTail), !.

biggerThan([\_|T], Min, ResTail) :-

biggerThan(T, Min, ResTail).

oddElements([], []).

oddElements([\_], []).

oddElements([\_|[H|T]],[H|ResTail]) :-

oddElements(T, ResTail).

deleteElement([], \_, []).

deleteElement([El|T], El, ResTail) :-

deleteElement(T, El, ResTail), !.

deleteElement([H|T], El, [H|ResTail]) :-

deleteElement(T, El, ResTail).

createSet([], []).

createSet([H|T], [H|ResTail]) :-

deleteElement(T, H, TmpRes),

createSet(TmpRes, ResTail).

goal

%biggerThan([1, 2, 3, 4, 2, 1], 5, Res).

%oddElements([1, 2, 3, 4], Res).

%oddElements([1, 2, 3], Res).

%deleteElement([1, 2, 3, 3, 1, 3, 4], 3, Res).

%createSet([1, 2, 3, 3, 1, 3, 4], Res).

%createSet([1, 1, 1], Res).

Эффективность достигнута за счет использования хвостовой рекурсии и использования отсечения.

# ТАБЛИЦА

**Текст процедуры**

1: biggerThan([], \_, []).

2: biggerThan([H|T], Min, [H|ResTail]) :-

H > Min,

biggerThan(T, Min, ResTail), !.

3: biggerThan([\_|T], Min, ResTail) :-

biggerThan(T, Min, ResTail).

**Вопрос: biggerThan([1], 2, Res)**

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Текущая резольвента – ТР | ТЦ, выбираемые правила: сравниваемые термы,  подстановка | Дальнейшие действия с комментариями |
| 1 | biggerThan([1], 2, Res) | ТЦ: biggerThan([1], 2, Res) | Поиск знания с начала БЗ |
|  | biggerThan([1], 2, Res) | ПР1:  [] = [1] \_ = 2 [] = Res  Неудача | Метка переносится ниже |
|  | biggerThan([1], 2, Res) | ПР2:  [H1|T1] = [1] Min1 = 2 [H1|ResTail1] = Res  Успех  H1 = 1 T1 = [] Min1 = 2 Res = [1|ResTail1] | Тело ПР2 заменяет цель в резольвенте |
| 2 | 1 > 2 biggerThan([], 2, ResTail1) ! | Сравнение:  1 > 2  Ложь | Откат к 1.  Метка переносится ниже. |
| 3 | biggerThan([1], 2, Res) | ПР3:  [\_|T3] = [1] Min3 = 2 ResTail3 = Res  Успех  T3 = [] Min3 = 2 Res = ResTail3 | Тело ПР3 заменяет цель в резольвенте |
| 4 | biggerThan([], 2, ResTail3) | ТЦ: biggerThan([], 2, ResTail3) | Поиск знания с начала БЗ |
|  | biggerThan([], 2, ResTail3) | ПР1: [] = [] \_ = 2 [] = ResTail3  Успех  ResTail3 = 0 | Пустое тело заменяет цель в резольвенте |
|  | Пусто |  | Успех.  Res = ResTail3 = []  Возврат к предыдущему состоянию резольвенты |
| 5 | biggerThan([], 2, ResTail3) | ПР2: [H5|T5] = [] Min5 = 2 [H5|ResTail5] = ResTail3  Неудача | Метка переносится ниже. |
|  | biggerThan([], 2, ResTail3) | ПР3:  [\_|T3] = [] Min5 = 2 ResTail5 = Res  Неудача | Необходимо включить откат, но метки в конце процедур – других альтернатив нет.  Система завершает работу. |