|  |  |
| --- | --- |
| 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» (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 19 |
|  |  |

**Дисциплина:** Функциональное и логическое программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-62Б |  |  | Е.В. Брянская |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Н.Б.Толпинская  Ю.В.Строганов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2021

**Задание**

Используя хвостовую рекурсию, разработать эффективную программу (комментируя назначение аргументов), позволяющую:

1. Найти длину списка (по верхнему уровню)
2. Найти сумму элементов числового списка
3. Найти сумму элементов числового списка, стоящих на нечетных позициях исходного списка (нумерация от 0)

Убедиться в правильности результатов.

Для одного из вариантов вопроса и одного из заданий составить таблицу, отражающую конкретный порядок работы системы.

Так как резольвента хранится в виде стека, то состояние резольвенты следует 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.

|  |
| --- |
| domains  lst = integer\*.    predicates  len(lst, integer).  len(lst, integer, integer).    sum(lst, integer).  sum(lst, integer, integer).    sum\_odd\_pos(lst, integer).  sum\_odd\_pos(lst, integer, integer).  clauses  % list's length  len([\_|T], Len\_temp, Len) :- % T - tail  Temp = Len\_temp + 1, % Len\_temp - temporary length  len(T, Temp, Len), % Len - length  !. % Temp - temporary var  len([], Len, Len) :- !. % Lst - currect list  len(Lst, Len) :- len(Lst, 0, Len), !.    % sum of elements  sum([X|T], Sum\_temp, Sum) :- % X - 1st element  Temp = Sum\_temp + X, % T - tail  sum(T, Temp, Sum). % Sum\_temp - temporary sum, Sum - sum  sum([], Sum, Sum) :- !. % Temp - temporary var  sum(Lst, Sum) :- sum(Lst, 0, Sum), !. % Lst - currect list    % sum of elements in odd positions  sum\_odd\_pos([\_, X|T], Sum\_temp, Sum) :- % X - 2nd element from head = 1st index  Temp = Sum\_temp + X, % T - tail  sum\_odd\_pos(T, Temp, Sum). % Sum\_temp - temporary sum, Sum - sum  sum\_odd\_pos([\_], Sum, Sum) :- !. % Temp - temporary var  sum\_odd\_pos([], Sum, Sum) :- !. % Lst - currect list, result  sum\_odd\_pos(Lst, Sum) :- sum\_odd\_pos(Lst, 0, Sum), !.    goal  %len([5], Len).  %sum([-5, 0, 5, -9], Sum).  %sum\_odd\_pos([-5, 1, 3], Sum). |

**Текст процедуры:**

|  |
| --- |
| len([\_|T], Len\_temp, Len) :- % T - tail  Temp = Len\_temp + 1, % Len\_temp - temporary length  len(T, Temp, Len), % Len - length  !. % Temp - temporary var  len([], Len, Len) :- !. % Lst - currect list  len(Lst, Len) :- len(Lst, 0, Len), !. |

**Вопрос**: len([5, -2, 0], Len).

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Текущая резольвента - ТР** | **ТЦ, выбираемые правила: сравниваемые термы, подстановка** | **Дальнейшие действия** |
| 0. | len([5, -2, 0], Len) |  |  |
|  | len([5, -2, 0], Len) | len([5, -2, 0], Len)  =  len([\_|T], Len\_temp, Len)  (1)  **Неудача**  (разная арность) | Прямой ход, переход к следующему правилу. |
|  | len([5, -2, 0], Len) | len([5, -2, 0], Len)  =  len([], Len, Len)  (2)  **Неудача**  (разная арность) | Прямой ход, переход к следующему правилу. |
|  | len([5, -2, 0], 0, Len\_1),  ! | len([5, -2, 0], Len)  =  len(Lst\_1, Len\_1)  (3)  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | Temp\_2 = 0 + 1,  len([-2,0], Temp\_2, Len\_2),  !,  ! | len([5, -2, 0], 0, Len\_1)  =  len([\_|T\_2], Len\_temp\_2, Len\_2)  (1)  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | len([-2,0], 1, Len\_2),  !,  ! | Temp\_2 = 0 + 1  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | Temp\_4 = 1 + 1,  len([0], Temp\_4, Len\_4),  !,  !,  ! | len([-2,0], 1, Len\_2)  =  len([\_|T\_4], Len\_temp\_4, Len\_4)  (1)  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | len([0], 2, Len\_4),  !,  !,  ! | Temp\_4 = 1 + 1  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | Temp\_6 = 2 + 1,  len([], Temp\_6, Len\_6),  !,  !,  !,  ! | len([0], 2, Len\_4)  =  len([\_|T\_6], Len\_temp\_6, Len\_6)  (1)  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2,  T\_6=[], Len\_temp\_6=2, Len\_4=Len\_6} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | len([], 3, Len\_6),  !,  !,  !,  ! | Temp\_6 = 2 + 1  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2,  T\_6=[], Len\_temp\_6=2, Len\_4=Len\_6,  Temp\_6=3} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | len([], 3, Len\_6),  !,  !,  !,  ! | len([], 3, Len)  =  len([\_|T], Len\_temp, Len)  (1)  **Неудача**  (пустой список) | Прямой ход, переход к следующему правилу. |
|  | !,  !,  !,  !,  ! | len([], 3, Len\_6)  =  len([], Len\_8, Len\_8)  (2)  **Удача**  **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2,  T\_6=[], Len\_temp\_6=2, Len\_4=Len\_6,  Temp\_6=3,  Len\_8=3, Len\_6=Len\_8} | Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | !,  !,  !,  ! | !  **Удача**  **Подстановка:** без изменений | **Отсечение** (системный предикат отсечения)  Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | !,  !,  ! | !  **Удача**  **Подстановка:** без изменений | **Отсечение** (системный предикат отсечения)  Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | !,  ! | !  **Удача**  **Подстановка:** без изменений | **Отсечение** (системный предикат отсечения)  Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | ! | !  **Удача**  **Подстановка:** без изменений | **Отсечение** (системный предикат отсечения)  Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки |
|  | **Резольвента пуста** | !  **Удача**  **Подстановка:** без изменений | **Отсечение** (системный предикат отсечения)  Прямой ход  **Изменение резольвенты**:   1. применение редукции 2. применение подстановки   **Вывод:**  Len = 3  **Откат**  (пустая резольвента) |
|  | len([], 3, Len\_6),  !,  !,  !,  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2,  T\_6=[], Len\_temp\_6=2, Len\_4=Len\_6,  Temp\_6=3} | **Откат**  (отсечение) |
|  | Temp\_6 = 2 + 1,  len([], Temp\_6, Len\_6),  !,  !,  !,  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2,  T\_6=[], Len\_temp\_6=2, Len\_4=Len\_6} | **Откат**  (унификация с константой) |
|  | len([0], 2, Len\_4),  !,  !,  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4,  Temp\_4=2} | **Откат**  (отсечение) |
|  | Temp\_4 = 1 + 1,  len([0], Temp\_4, Len\_4),  !,  !,  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1,  T\_4=[0], Len\_temp\_4=1, Len\_2=Len\_4} | **Откат**  (унификация с константой) |
|  | len([-2,0], 1, Len\_2),  !,  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2,  Temp\_2=1} | **Откат**  (отсечение) |
|  | Temp\_2 = 0 + 1,  len([-2,0], Temp\_2, Len\_2),  !,  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1,  T\_2=[-2,0], Len\_temp\_2=0, Len\_1=Len\_2} | **Откат**  (унификация с константой) |
|  | len([5, -2, 0], 0, Len\_1),  ! | **Подстановка:**  {Lst\_1=[5, -2, 0], Len=Len\_1} | **Откат**  (отсечение) |
|  | len([5, -2, 0], Len) | **Подстановка:**  {} | Завершение работы |

**Вопросы**

1. Что такое рекурсия? Как организуется хвостовая рекурсия в Prolog? Как можно организовать выход из рекурсии в Prolog?

Рекурсия – это ссылка на описываемый объект в процессе его описания. При хвостовой рекурсии все действия сделаны до момента выхода из неё, вызов единственен. Выход из рекурсии организуется с помощью отсечения.

1. Какое первое состояние резольвенты?

Начальное состояние резольвенты – вопрос.

1. В каких пределах программы переменные уникальны?

Именованные переменные уникальны в пределах предложения. Анонимные переменные уникальны всегда.

1. В какой момент, и каким способом системе удаётся получить доступ к голове списка?

Получить голову можно при унификации списка ([H|T], где H – голова, T – хвост).

1. Каково назначение использования алгоритма унификации?

Алгоритм унификации используется для доказательства очередной цели.

1. Каков результат работы алгоритма унификации?

Алгоритм унификации делает вывод о том, унифицируемы два терма или нет, и если да, то строит наиболее общий унификатор.

1. Как формируется новое состояние резольвенты?

Резольвента меняется в два этапа:

1. В текущей резольвенте выбирается одна из целей, для неё выполняется редукция
2. Затем к резольвенте применяется подстановка, полученная, как наибольший общий унификатор цели и заголовка сопоставимого с ней правила.
3. Как применяется подстановка, полученная с помощью алгоритма унификации, как глубоко?

Подстановка - это множество пар вида {Xi = ti}. Применить подстановку, значит, найти все вхождения в резольвенте и результирующей ячейке Xi и заменить на соответствующее значение ti.Применяется на любую вложенность.

1. В каких случаях запускается механизм отката?

Механизм отката запускается в случаях, если резольвента оказалась пустой (то есть, будет воспроизведена попытка найти следующее подходящее знание), либо возникла тупиковая ситуация (просмотрена вся БЗ). В обоих случаях происходит откат к предыдущему состоянию резольвенты.

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

На формальном уровне это определяется тем, что в резольвенте находится исходный вопрос, для которого вся БЗ просмотрена. То есть система завершает работу в случае, когда все возможные ответы рассмотрены.