|  |  |
| --- | --- |
| **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»

**ОТЧЕТ**

*к лабораторной работе №17*

*По курсу: «Функциональное и логическое программирование»*

**Тема: «**Обработка списков на Prolog»

Студент: Зайцева А. А.

Группа: ИУ7-62Б

Преподаватели: Толпинская Н. Б.,

Строганов Ю. В.

Москва, 2022 г.

# Практическая часть

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:

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.

|  |
| --- |
| domains  list = integer\*.  predicates  length(list, integer).  lengthInner(list, integer, integer).  sum(list, integer).  sumInner(list, integer, integer).  oddSum(list, integer).  oddSumInner(list, integer, integer).  clauses  lengthInner([], FinalAnswer, CurAnswer) :-  FinalAnswer = CurAnswer,  !.    lengthInner([\_|Tail], FinalAnswer, CurAnswer) :-  NextAnswer = CurAnswer + 1,  lengthInner(Tail, FinalAnswer, NextAnswer).    length(List, Length) :-  lengthInner(List, Length, 0).  sumInner([], FinalSum, CurSum) :-  FinalSum = CurSum,  !.    sumInner([Head|Tail], FinalSum, CurSum) :-  NextSum = CurSum + Head,  sumInner(Tail, FinalSum, NextSum).    sum(List, Sum) :-  sumInner(List, Sum, 0).  % if list has even lenght  oddSumInner([], FinalSum, CurSum) :-  FinalSum = CurSum,  !.  % if list has odd lenght  oddSumInner([\_|[]], FinalSum, CurSum) :-  FinalSum = CurSum,  !.    oddSumInner([\_|[NextHead|NextTail]], FinalSum, CurSum) :-  NextSum = CurSum + NextHead,  oddSumInner(NextTail, FinalSum, NextSum),  !.    oddSum(List, Sum) :-  oddSumInner(List, Sum, 0).  goal  % length([], Length\_).  % Length\_=0  % 1 Solution  % length([1, 2, 3], Length\_).  % Length\_=3  % 1 Solution    % sum([], Sum\_).  % Sum\_=0  % 1 Solution  % sum([-1, 1, 2], Sum\_).  % Sum\_=2  % 1 Solution    % oddSum([], Sum\_).  % Sum\_=0  % 1 Solution    % oddSum([0], Sum\_).  % Sum\_=0  % 1 Solution    % oddSum([0, 1], Sum\_).  % Sum\_=1  % 1 Solution    oddSum([0, 1, 2], Sum\_).  % Sum\_=1  % 1 Solution    % oddSum([0, 1, 2, 3], Sum\_).  % Sum\_=4  % 1 Solution    % oddSum([0, 1, 2, 3, 4], Sum\_).  % Sum\_=4  % 1 Solution |

Вопрос: oddSum([0, 1, 2], Sum\_).

Текст процедуры:

oddSum(List, Sum) :- oddSumInner(List, Sum, 0).

Текст вспомогательной процедуры:

% if list has even lenght

oddSumInner([], FinalSum, CurSum) :-

FinalSum = CurSum,

!.

% if list has odd lenght

oddSumInner([\_|[]], FinalSum, CurSum) :-

FinalSum = CurSum,

!.

oddSumInner([\_|[NextHead|NextTail]], FinalSum, CurSum) :-

NextSum = CurSum + NextHead,

oddSumInner(NextTail, FinalSum, NextSum),

!.

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Текущая резольвента–ТР | ТЦ, выбираемые правила: сравниваемые термы, подстановка | Дальнейшие действия с  комментариями |
| 0 | oddSum([0, 1, 2], Sum\_) |  | Запуск алгоритма унификации для вопроса, с начала БЗ |
| 1 | oddSum([0, 1, 2], Sum\_) | oddSum([0, 1, 2], Sum\_)= lengthInner([], FinalAnswer, CurAnswer)  Сравнение главных функторов:  oddSum = lengthInner  Унификация неуспешна | Резольвента не менятеся (неуспешная унификацияя с заголовком)  Прямой ход, переход к следующему предложению БЗ |
| … |  |  |  |
| 10 | oddSum([0, 1, 2], Sum\_) | oddSum([0, 1, 2], Sum\_)=  oddSum(List, Sum)  Унификация успешна  Подстановка: { List=[0, 1, 2]} | Верхняя подцель успешно унифицировалась с заголовком правила. Образование новой резольвенты:  1. Редукция верхней подцели: замена oddSum([0, 1, 2], Sum\_) телом найденного правила:  oddSumInner(List, Sum, 0)  2. Применение подстановки к полученной  конъюнкции целей. Новое состояние резольвенты:  oddSumInner([0, 1, 2], Sum, 0)  Резольвента непуста, запуск алгоритма унификации для верхней подцели c начала БЗ |
| … |  |  |  |
| 17 | oddSumInner([0, 1, 2], Sum, 0) | oddSumInner([0, 1, 2], Sum, 0)=  oddSumInner([], FinalSum, CurSum)  Унификация неуспешна | Прямой ход, переход к следующему предложению БЗ |
| 18 | oddSumInner([0, 1, 2], Sum, 0) | oddSumInner([0, 1, 2], Sum, 0)=  oddSumInner([\_|[]], FinalSum, CurSum)  Унификация неуспешна | Прямой ход, переход к следующему предложению БЗ |
| 19 | oddSumInner([0, 1, 2], Sum, 0) | oddSumInner([0, 1, 2], Sum, 0)=  oddSumInner([\_|[NextHead|NextTail]], FinalSum, CurSum)  Унификация успешна  Подстановка: { NextHead=1, NextTail=[2], CurSum=0} | Верхняя подцель успешно унифицировалась с заголовком правила. Образование новой резольвенты:  1. Редукция верхней подцели: замена oddSumInner([0, 1, 2], Sum, 0) телом найденного правила:  NextSum = CurSum + NextHead,  oddSumInner(NextTail, FinalSum, NextSum),  !.  2. Применение подстановки к полученной  конъюнкции целей. Новое состояние резольвенты:  NextSum = 0 + 1,  oddSumInner([2], FinalSum, NextSum),  !.  Резольвента непуста, запуск алгоритма унификации для верхней подцели |
| 20 | NextSum = 0 + 1,  oddSumInner([2], FinalSum, NextSum),  !. | NextSum = 0 + 1  Унификация успешна  Подстановка: { NextSum=1} | Образование новой резольвенты:  1. Редукция верхней подцели: удаление  NextSum = 0 + 1  2. Применение подстановки к полученной  конъюнкции целей. Новое состояние резольвенты:  oddSumInner([2], FinalSum, 1),  !.  Резольвента непуста, запуск алгоритма унификации для верхней подцели c начала БЗ |
| … |  |  |  |
| 27 | oddSumInner([2], FinalSum, 1),  !. | oddSumInner([2], Fi-nalSum, 1)=  oddSumInner([], FinalSum, CurSum)  Унификация неуспешна | Прямой ход, переход к следующему предложению БЗ |
| 28 | oddSumInner([2], Fi-nalSum, 1),  !. | oddSumInner([2], FinalSum, 1)=  oddSumInner([\_|[]], FinalSum, CurSum)  Унификация успешна  Подстановка: { CurSum=1 } | Верхняя подцель успешно унифицировалась с заголовком правила. Образование новой резольвенты:  1. Редукция верхней подцели: замена oddSumInner([2], Fi-nalSum, 1) телом найденного правила:  FinalSum = CurSum,  !  2. Применение подстановки к полученной  конъюнкции целей. Новое состояние резольвенты:  FinalSum = 1,  !,  !.  Резольвента непуста, запуск алгоритма унификации для верхней подцели |
| 29 | FinalSum = 1,  !,  !. | FinalSum = 1  Унификация успешна  Подстановка: { FinalSum = 1} | Образование новой резольвенты:  1. Редукция верхней подцели: удаление  FinalSum = 1  2. Применение подстановки к полученной  конъюнкции целей. Новое состояние резольвенты:  !,  !.  Резольвента непуста, запуск алгоритма унификации для верхней подцели |
| 30 | !,  !. | !  Истина | Встречен системный предикат отсечения, который на обратном ходе запрещает использование других правил процдуры oddSumInner. Редукция верхней подцели |
| 31 | ! | !  Истина | Встречен системный предикат отсечения, который на обратном ходе запрещает использование других правил процдуры oddSumInner. Редукция верхней подцели |
| 32 |  |  | Резольвента пуста. Решение найдено: формируется подстановка { Sum\_=1} в качестве побочного эффекта.  Восстановление предыдущего состояния резольты, дважды ! запрещает использование других правил процдуры oddSumInner, новое состояние резольвенты oddSum([0, 1, 2], Sum\_). Конец БЗ. Завершение работы. |