|  |  |
| --- | --- |
| 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 №18-20*

*По курсу: «Функциональное и логическое программирование»*

**Темы:** «Рекурсия на Prolog», «Обработка списков на Prolog», «Формирование и модификация списков на Prolog»

Студент: Якуба Д.В.

Группа: ИУ7-63Б

Преподаватели: Толпинская Н. Б.,

Строганов Ю. В.

Москва, 2021 г.

# Практическая часть

# Лабораторная работа 18.

Задание. **Используя хвостовую рекурсию, разработать программу, позволяющую найти:**

1. **n!**,
2. **n-e** число Фибоначчи.

Убедиться в правильности результатов.

Для одного из вариантов ВОПРОСА и каждого 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!

|  |
| --- |
| predicates  factorial(integer, integer).  factorialInner(integer, integer, integer).  fibbo(integer, integer).  fibboInner(integer, integer, integer, integer).  clauses  factorialInner(Number, Answer, NewAnswer) :- Number <= 1, Answer = NewAnswer, !.  factorialInner(Number, Answer, NewAnswer) :-  TempAnswer = NewAnswer \* Number,  TempNumber = Number - 1,  factorialInner(TempNumber, Answer, TempAnswer).  factorial(Number, Answer) :- factorialInner(Number, Answer, 1).  fibboInner(Number, Answer, \_, PrevAnswer) :- Number <= 2, Answer = PrevAnswer, !.  fibboInner(Number, Answer, PrevPrevAnswer, PrevAnswer) :-  TempNumber = Number - 1,  NewPrev = PrevPrevAnswer + PrevAnswer,  fibboInner(TempNumber, Answer, PrevAnswer, NewPrev).  fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1).  goal  factorial(1, Answer\_); % 1  factorial(2, Answer\_); % 2  factorial(3, Answer\_); % 6  factorial(4, Answer\_); % 24  factorial(5, Answer\_); % 120  factorial(6, Answer\_); % 720  factorial(7, Answer\_); % 5040  factorial(8, Answer\_); % 40320  fibbo(1, Answer\_); % 1  fibbo(2, Answer\_); % 1  fibbo(3, Answer\_); % 2  fibbo(4, Answer\_); % 3  fibbo(5, Answer\_); % 5  fibbo(6, Answer\_); % 8  fibbo(7, Answer\_); % 13  fibbo(8, Answer\_) % 21  . |

Вопрос: factorial(3, Answer\_)

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Состояние резольвенты, и вывод: дальнейшие действия (почему?) | Для каких термов запускается алгоритм унификации: Т1=Т2 и каков результат (и подстановка) | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
| 1 | factorial(3, Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorial(3, Answer\_) = factorialInner(Number, Answer, NewAnswer) :- Number <= 0, Answer = NewAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 2 | … | … | … |
| 3 | factorial(3, Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorial(3, Answer\_) = factorial(Number, Answer) :- factorialInner(Number, Answer, 1)  Унификация успешна  Подстановка: {Number = 3, Answer = Answer\_} | Формируется новое состояние резольвенты. Прямой ход |
| 4 | factorialInner(3, Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(3, Answer\_, 1) = factorialInner(Number, Answer, NewAnswer) :- Number <= 1, Answer = NewAnswer, !  Унификация успешна  Подстановка:  {Number = 3, Answer = Answer\_, NewAnswer = 1} | Формируется новое состояние резольвенты. Прямой ход |
| 5 | 3 <= 1,  Answer\_ = 1,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Сравнение констант:  3 <= 1  Результат: ложь | Откат к предыдущему состоянию резольвенты  Новая подстановка: {Number = 3, Answer = Answer\_} |
| 6 | factorialInner(3, Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(3, Answer\_, 1) = factorialInner(Number, Answer, NewAnswer) :- TempAnswer = NewAnswer \* Number, TempNumber = Number - 1, factorialInner(TempNumber, Answer, TempAnswer)  Унификация успешна  Подстановка: {Number = 3, Answer = Answer\_, NewAnswer = 1} | Формируется новое состояние резольвенты. Прямой ход |
| 7 | TempAnswer = 1 \* 3,  TempNumber = 3 - 1,  factorialInner(TempNumber, Answer\_, TempAnswer)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempAnswer = 1 \* 3  Успех. Подстановка: {TempAnswer = 1 \* 3} | Формируется новое состояние резольвенты. Прямой ход |
| 8 | TempNumber = 3 - 1,  factorialInner(TempNumber, Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempNumber = 3 – 1  Успех. Подстановка: {TempNumber = 2} | Формируется новое состояние резольвенты. Прямой ход |
| 9 | factorialInner(2, Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(2, Answer\_, 3) = factorialInner(Number, Answer, NewAnswer) :- Number <= 1, Answer = NewAnswer, !.  Унификация успешна  Подстановка: {Number = 2, Answer = Answer\_, NewAnswer = 3} | Формируется новое состояние резольвенты. Прямой ход |
| 10 | 2 <= 1,  Answer\_ = 3,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Сравнение констант:  2 <= 1  Результат: ложь | Откат к предыдущему состоянию резольвенты  Новая подстановка: Подстановка: {Number = 2, Answer = Answer\_, NewAnswer = 3} |
| 11 | factorialInner(2, Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(2, Answer\_, 3) = factorialInner(Number, Answer, NewAnswer) :- TempAnswer = NewAnswer \* Number, TempNumber = Number - 1, factorialInner(TempNumber, Answer, TempAnswer)  Унификация успешна  Подстановка: {Number = 2, Answer = Answer\_, NewAnswer = 3} | Формируется новое состояние резольвенты. Прямой ход |
| 12 | TempAnswer = 3 \* 2,  TempNumber = 2 - 1,  factorialInner(TempNumber, Answer, TempAnswer)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempAnswer = 3 \* 2  Успех. Подстановка: {TempAnswer = 6} | Формируется новое состояние резольвенты. Прямой ход |
| 13 | TempNumber = 2 - 1,  factorialInner(TempNumber, Answer, 6)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempNumber = 2 – 1  Успех. Подстановка: {TempNumber = 1} | Формируется новое состояние резольвенты. Прямой ход |
| 14 | factorialInner(1, Answer\_, 6)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(1, Answer\_, 6) = factorialInner(Number, Answer, NewAnswer) :- Number <= 1, Answer = NewAnswer, !.  Унификация успешна  Подстановка: {Number = 1, Answer = Answer\_, NewAnswer = 6} | Формируется новое состояние резольвенты. Прямой ход |
| 15 | 1 ≤ 1,  Answer\_ = 6,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Сравнение констант:  1 <= 1  Результат: истина | Формируется новое состояние резольвенты. Прямой ход |
| 16 | Answer\_ = 6,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  Answer\_ = 6  Подстановка: {Answer\_ = 6} | Формируется новое состояние резольвенты. Прямой ход |
| 17 | !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Встречен системный предикат отсечения | Резольвента пуста.  Вывод: Answer\_ = 6  Откат с отсечением остаточных предложений процедуры относительно шага 14  Новая подстановка: {Number = 2, Answer = Answer\_, NewAnswer = 3} |
| 18 | factorialInner(1, Answer\_, 6)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(1, Answer\_, 6) = factorial(Number, Answer) :- factorialInner(Number, Answer, 1).  Унификация неуспешна (несовпадение функторов) | Откат, следующее предложение |
| 19-20 | … | … | … |
| 21 | factorialInner(1, Answer\_, 6)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(1, Answer\_, 6) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, следующее предложение относительно пункта 11  Новая подстановка: {Number = 3, Answer = Answer\_, NewAnswer = 1} |
| 22 | factorialInner(2, Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(2, Answer\_, 3) = factorial(Number, Answer) :- factorialInner(Number, Answer, 1)  Унификация неуспешна (несовпадение функторов) | Откат, следующее предложение |
| 23-24 | … | … | … |
| 25 | factorialInner(2, Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(2, Answer\_, 3) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, следующее предложение относительно пункта 6  Новая подстановка: {Number = 3, Answer = Answer\_, NewAnswer = 1} |
| 26 | factorialInner(3, Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(3, Answer\_, 1) = factorial(Number, Answer) :- factorialInner(Number, Answer, 1)  Унификация неуспешна (несовпадение функторов) | Откат, следующее предложение |
| 27-28 | … | … | … |
| 29 | factorialInner(3, Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorialInner(3, Answer\_, 1) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, следующее предложение относительно пункта 3  Подстановки более нет |
| 30 | factorial(3, Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorial(3, Answer\_) = fibboInner(Number, Answer, PrevAnswer, \_) :- Number < 2, Answer = PrevAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, следующее предложение |
| 31 | … | … | … |
| 32 | factorial(3, Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | factorial(3, Answer\_) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут, завершение работы системы. |

Вопрос: fibbo(3, Answer)

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Состояние резольвенты, и вывод: дальнейшие действия (почему?) | Для каких термов запускается алгоритм унификации: Т1=Т2 и каков результат (и подстановка) | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
| 1 | fibbo(3, Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibbo(3, Answer\_) = factorialInner(Number, Answer, NewAnswer) :- Number <= 0, Answer = NewAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 2-5 | … | … | … |
| 6 | fibbo(3, Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibbo(3, Answer\_) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1).  Унификация успешна  Подстановка: {Number = 3, Answer = Answer\_} | Формируется новое состояние резольвенты, прямой ход |
| 7 | fibboInner(3, Answer\_, 1, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(3, Answer\_, 1, 1) = factorialInner(Number, Answer, NewAnswer) :- Number <= 0, Answer = NewAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 8-9 | … | … | … |
| 10 | fibboInner(3, Answer\_, 1, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(3, Answer\_, 1, 1) = fibboInner(Number, Answer, \_, PrevAnswer) :- Number <= 2, Answer = PrevAnswer, !.  Унификация успешна  Подстановка: {Number = 3, Answer = Answer\_, PrevAnswer = 1} | Формируется новое состояние резольвенты, прямой ход |
| 11 | 3 <= 2,  Answer = 1,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Сравнение констант:  3 <= 2  Результат: ложь | Откат к предыдущему состоянию резольвенты  Новая подстановка: Подстановка: {Number = 3, Answer = Answer\_} |
| 12 | fibboInner(3, Answer\_, 1, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(3, Answer\_, 1, 1) = fibboInner(Number, Answer, PrevPrevAnswer, PrevAnswer) :- TempNumber = Number - 1, NewPrev = PrevPrevAnswer + PrevAnswer, fibboInner(TempNumber, Answer, PrevAnswer, NewPrev).  Унификация успешна  Подстановка: {Number = 3, Answer = Answer\_, PrevPrevAnswer = 1, PrevAnswer = 1} | Формируется новое состояние резольвенты |
| 13 | TempNumber = 3 - 1,  NewPrev = 1 + 1, fibboInner(TempNumber, Answer\_, 1, NewPrev)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempNumber = 3 – 1  Успех. Подстановка: {TempNumber = 2} | Формируется новое состояние резольвенты |
| 14 | NewPrev = 1 + 1, fibboInner(TempNumber, Answer\_, 1, NewPrev)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  NewPrev = 1 + 1  Успех. Подстановка: {NewPrev = 2} | Формируется новое состояние резольвенты |
| 15 | fibboInner(2, Answer\_, 1, 2)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(2, Answer\_, 1, 1) = factorialInner(Number, Answer, NewAnswer) :- Number <= 0, Answer = NewAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, следующее предложение |
| 16-17 | … | … | … |
| 18 | fibboInner(2, Answer\_, 1, 2)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(2, Answer\_, 1, 2) = fibboInner(Number, Answer, \_, PrevAnswer) :- Number <= 2, Answer = PrevAnswer, !.  Унификация успешна  Подстановка: {Number = 2, Answer = Answer\_, PrevAnswer = 2} | Формируется новое состояние резольвенты |
| 19 | 2 <= 2,  Answer\_ = PrevAnswer,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Сравнение констант:  2 <= 2  Результат: истина | Формируется новое состояние резольвенты |
| 20 | Answer\_ = 2,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  Answer\_ = 2  Успех. Подстановка: {Answer\_ = 2} | Формируется новое состояние резольвенты |
| 21 | !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Встречен системный предикат отсечения | Резольвента пуста.  Вывод: Answer\_ = 2  Откат с отсечением остаточных предложений процедуры относительно шага 18  Новая подстановка: {Number = 2, Answer = Answer\_} |
| 22 | fibboInner(2, Answer\_, 1, 2)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(2, Answer\_, 1, 2) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, следующее предложение относительно шага 12  Подстановка: {Number = 3, Answer = Answer\_} |
| 23 | fibboInner(3, Answer\_, 1, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | fibboInner(3, Answer\_, 1, 1) = fibbo(Number, Answer) :- fibboInner(Number, Answer, 1, 1)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, следующее предложение относительно шага 6  Подстановки более нет |
| 24 | fibbo(3, Answer\_) | - | Конец БЗ достигнут. Завершение работы системы |

# Лабораторная работа 19.

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  listLen(list, integer).  listLenInner(list, integer, integer).  listSum(list, integer).  listSumInner(list, integer, integer).  listOddSum(list, integer).  listOddSumInner(list, integer, integer).  clauses  listLenInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !.  listLenInner([\_|T], OutAnswer, CurAnswer) :-  TempAnswer = CurAnswer + 1,  listLenInner(T, OutAnswer, TempAnswer).    listLen(List, Answ) :- listLenInner(List, Answ, 0).  listSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !.  listSumInner([H|T], OutAnswer, CurAnswer) :-  TempAnswer = CurAnswer + H,  listSumInner(T, OutAnswer, TempAnswer).  listSum(List, Answ) :- listSumInner(List, Answ, 0).    listOddSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !.  listOddSumInner([H|[\_|TT]], OutAnswer, CurAnswer) :-  TempAnswer = CurAnswer + H,  listOddSumInner(TT, OutAnswer, TempAnswer), !.  listOddSumInner([H|\_], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer + H.  listOddSum(List, Answ) :- listOddSumInner(List, Answ, 0).  goal  % listLen([], Answer\_);  % listLen([1, 2, 3], Answer\_);    % listSum([], Answer\_);  % listSum([1, 2, 4], Answer\_);  % listSum([1, 2, 3, 4, 5, -5], Answer\_)    listOddSum([], Answer\_);  listOddSum([1, 2, 3, 4, 5, 6, 7], Answer\_)  . |

Текст процедуры:

listSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !.

listSumInner([H|T], OutAnswer, CurAnswer) :-

TempAnswer = CurAnswer + H,

listSumInner(T, OutAnswer, TempAnswer).

listSum(List, Answ) :- listSumInner(List, Answ, 0).

Вопрос: listSum([1, 2, 4], Answer\_)

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Состояние резольвенты, и вывод: дальнейшие действия (почему?) | Для каких термов запускается алгоритм унификации: Т1=Т2 и каков результат (и подстановка) | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
| 1 | listSum([1, 2, 4], Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSum([1, 2, 4], Answer\_) = listLenInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 2-5 | … | … | … |
| 4 | listSum([1, 2, 4], Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSum([1, 2, 4], Answer\_) = listSum(List, Answ) :- listSumInner(List, Answ, 0)  Унификация успешна  Подстановка: {List = [1, 2, 4], Answ = Answer\_} | Формируется новое состояние резольвенты |
| 5 | listSumInner([1, 2, 4], Answer\_, 0)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([1, 2, 4], Answer\_, 0) = listLenInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 6-7 | … | … | … |
| 8 | listSumInner([1, 2, 4], Answer\_, 0)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([1, 2, 4], Answer\_, 0) = listSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (первые аргументы не унифицируемы) | Откат, переход к следующему предложению |
| 9 | listSumInner([1, 2, 4], Answer\_, 0)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([1, 2, 4], Answer\_, 0) = listSumInner([H|T], OutAnswer, CurAnswer) :-  TempAnswer = CurAnswer + H,  listSumInner(T, OutAnswer, TempAnswer)  Унификация успешна  Подстановка: {H = 1, T = [2, 4], OutAnswer = Answer\_, CurAnswer = 0} | Формируется новое состояние резольвенты, прямой ход |
| 10 | TempAnswer = 0 + 1,  listSumInner([2, 4], Answer\_, TempAnswer)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempAnswer = 0 + 1  Успех. Подстановка: {TempAnswer = 1} | Формируется новое состояние резольвенты, прямой ход |
| 11 | listSumInner([2, 4], Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([2, 4], Answer\_, 1) = listLenInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 12-13 | … | … | … |
| 14 | listSumInner([2, 4], Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([2, 4], Answer\_, 1) = listSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (первые аргументы не унифицируемы) | Откат, переход к следующему предложению |
| 15 | listSumInner([2, 4], Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([2, 4], Answer\_, 1) = listSumInner([H|T], OutAnswer, CurAnswer) :-  TempAnswer = CurAnswer + H,  listSumInner(T, OutAnswer, TempAnswer)  Унификация успешна  Подстановка: {H = 2, T = [4], OutAnswer = Answer\_, CurAnswer = 1} | Формируется новое состояние резольвенты, прямой ход |
| 16 | TempAnswer = 1 + 2,  listSumInner([4], Answer\_, TempAnswer)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempAnswer = 1 + 2  Успех. Подстановка: {TempAnswer = 3} | Формируется новое состояние резольвенты, прямой ход |
| 17 | listSumInner([4], Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([4], Answer\_, 3) = listLenInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 18-19 | … | … | … |
| 20 | listSumInner([4], Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([4], Answer\_, 3) = listSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (первые аргументы не унифицируемы) | Откат, переход к следующему предложению |
| 21 | listSumInner([4], Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([4], Answer\_, 3) = listSumInner([H|T], OutAnswer, CurAnswer) :-  TempAnswer = CurAnswer + H,  listSumInner(T, OutAnswer, TempAnswer)  Унификация успешна  Подстановка: {H = 4, T = [], OutAnswer = Answer\_, CurAnswer = 3} | Формируется новое состояние резольвенты, прямой ход |
| 22 | TempAnswer = 3 + 4,  listSumInner([], Answer\_, TempAnswer)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  TempAnswer = 3 + 4  Успех. Подстановка: {TempAnswer = 7} | Формируется новое состояние резольвенты, прямой ход |
| 23 | listSumInner([], Answer\_, 7)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([], Answer\_, 7) = listLenInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 24-25 | … | … | … |
| 26 | listSumInner([], Answer\_, 7)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([], Answer\_, 7) = listSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация успешна  Подстановка: {OutAnswer = Answer\_, CurAnswer = 7} | Формируется новое состояние резольвенты, прямой ход |
| 27 | Answer\_ = 7,  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  Answer\_ = 7  Успех. Подстановка: {Answer\_ = 7} | Формируется новое состояние резольвенты, прямой ход |
| 28 | !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Встречен системный предикат отсечения | Резольвента пуста.  Вывод: Answer\_ = 7  Откат с отсечением остаточных предложений процедуры относительно шага 26 |
| 29 | listSumInner([], Answer\_, 7)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([], Answer\_, 7) = listOddSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 30-31 | … | … | … |
| 32 | listSumInner([], Answer\_, 7)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([], Answer\_, 7) = listOddSum(List, Answ) :- listOddSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, переход к следующему предложению относительно шага 21 |
| 33 | listSumInner([4], Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([4], Answer\_, 3) = listSum(List, Answ) :- listSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 34-36 | … | … | … |
| 37 | listSumInner([4], Answer\_, 3)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([4], Answer\_, 3) = listOddSum(List, Answ) :- listOddSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, переход к следующему предложению относительно шага 15 |
| 38 | listSumInner([2, 4], Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([2, 4], Answer\_, 1) = listSum(List, Answ) :- listSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 39-41 | … | … | … |
| 42 | listSumInner([2, 4], Answer\_, 1)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([2, 4], Answer\_, 1) = listOddSum(List, Answ) :- listOddSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, переход к следующему предложению относительно шага 9 |
| 43 | listSumInner([1, 2, 4], Answer\_, 0)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([1, 2, 4], Answer\_, 0) = listSum(List, Answ) :- listSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 44-46 | … | … | … |
| 47 | listSumInner([1, 2, 4], Answer\_, 0)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSumInner([1, 2, 4], Answer\_, 0) = listOddSum(List, Answ) :- listOddSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат, переход к следующему предложению относительно шага 4 |
| 48 | listSum([1, 2, 4], Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSum([1, 2, 4], Answer\_) = listOddSumInner([], OutAnswer, CurAnswer) :- OutAnswer = CurAnswer, !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 49-50 | … | … | … |
| 51 | listSum([1, 2, 4], Answer\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listSum([1, 2, 4], Answer\_) = listOddSum(List, Answ) :- listOddSumInner(List, Answ, 0)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Резольвента пуста, завершение работы системы |

# Лабораторная работа 20.

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-го 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 хапуска алгоритма унификации, требуется указать № выбранного правила и соответствующий вывод: успех или нет – и почему.

|  |
| --- |
| domains  list = integer\*.  predicates  append(list, list, list).  listWithUpper(list, integer, list).  oddPosList(list, list).  deleteOneElement(list, integer, list).  deleteAllElement(list, integer, list).  includes(list, integer).  makeSet(list, list).  clauses  % SList - второй добавляемый список  % AddTail - хвост списка, уходящий для собственного определения в глубину,  % пока первый список не станет пустым (что станет и с AddTail)  append([H|T], SList, [H|AddTail]) :-  append(T, SList, AddTail), % look for new value of AddTail  !.  append([], SList, SList). % Just append last elements to a list  % Num - число, больше которого должны быть элементы списка  % AddTail - хвост, уходящий для собственного определения в глубину  listWithUpper([H|T], Num, [H|AddTail]) :-  H > Num,  !,  listWithUpper(T, Num, AddTail).  listWithUpper([\_|T], Num, OutList) :- listWithUpper(T, Num, OutList), !.  listWithUpper([], \_, []).  % OutList - см. AddTail предыдущих реализаций  oddPosList([H|[\_|T]], [H|OutList]) :-  oddPosList(T, OutList),  !.  oddPosList(List, List).  % El - элемент, подлежащий удалению  % AddTail - см. предыдущие реализации  deleteOneElement([H|T], El, [H|AddTail]) :-  El <> H,  !,  deleteOneElement(T, El, AddTail).  deleteOneElement([H|T], H, T) :- !.  deleteOneElement(List, \_, List).  % El - см. предыдущую реализацию  % AddTail - см. предыдущую реализацию  deleteAllElement([H|T], El, AddTail) :-  El = H,  !,  deleteAllElement(T, El, AddTail).  deleteAllElement([H|T], El, [H|AddTail]) :-  deleteAllElement(T, El, AddTail),  !.  deleteAllElement(List, \_, List).  % El - элемент, включение которого требуется определить  includes([\_|T], El) :-  includes(T, El),  !.  includes([H|\_], H).  % AddTail - см. предыдущие реализации  makeSet([H|T], AddTail) :-  includes(T, H),  !,  makeSet(T, AddTail).  makeSet([H|T], [H|AddTail]) :- makeSet(T, AddTail), !.  makeSet(List, List).  goal  makeSet([1, 2, 3, 2, 4, 1, 5, 6, 6, 6], OutList\_).  % deleteOneElement([1, 2, 1, 3, 1, 4], 1, OutList\_).  % deleteAllElement([1, 2, 1, 3, 1, 4], 1, OutList\_).  % oddPosList([1, 2, 3, 4, 5, 6], OutList\_).  % append([1, 2, 3], [4, 5, 6], OutList\_).  % listWithUpper([1, 2, 3, 4, 5, 6, 7, 8], 4, OutList\_). |

Текст процедуры:

listWithUpper([H|T], Num, [H|AddTail]) :-

H > Num,

!,

listWithUpper(T, Num, AddTail).

listWithUpper([\_|T], Num, OutList) :- listWithUpper(T, Num, OutList).

listWithUpper([], \_, []) :- !.

Вопрос: listWithUpper([1, 4, 5], 4, OutList\_)

|  |  |  |  |
| --- | --- | --- | --- |
| № шага | Состояние резольвенты, и вывод: дальнейшие действия (почему?) | Для каких термов запускается алгоритм унификации: Т1=Т2 и каков результат (и подстановка) | Дальнейшие действия: прямой ход или откат (к чему приводит?) |
| 1 | listWithUpper([1, 4, 5], 4, OutList\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([1, 4, 5], 4, OutList\_) = append([H|T], SList, [H|AddTail]) :- append(T, SList, AddTail), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 2 | … | … | … |
| 3 | listWithUpper([1, 4, 5], 4, OutList\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([1, 4, 5], 4, OutList\_) = listWithUpper([H|T], Num, [H|AddTail]) :- H > Num, !, listWithUpper(T, Num, AddTail)  Унификация успешна  Подстановка: {H = 1, T = [4, 5], Num = 4, [H|AddTail] = OutList\_} | Формируется новое состояние резольвенты, прямой ход |
| 4 | 1 > 4,  !,  listWithUpper([4, 5], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  1 > 4  Результат: ложь | Откат, возврат к предыдущему состоянию резольвенты, переход к следующему предложению |
| 5 | listWithUpper([1, 4, 5], 4, OutList\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([1, 4, 5], 4, OutList\_) = listWithUpper([\_|T], Num, OutList) :- listWithUpper(T, Num, OutList), !  Унификация успешна  Подстановка: {T = [4, 5], Num = 4, OutList = OutList\_} | Формируется новое состояние резольвенты, прямой ход |
| 6 | listWithUpper([4, 5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([4, 5], 4, OutList\_) = append([H|T], SList, [H|AddTail]) :- append(T, SList, AddTail), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 7 | … | … | … |
| 8 | listWithUpper([4, 5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([4, 5], 4, OutList\_) = listWithUpper([H|T], Num, [H|AddTail]) :- H > Num, !, listWithUpper(T, Num, AddTail)  Унификация успешна  Подстановка: {H = 4, T = [5], Num = 4, [4|AddTail] = OutList\_} | Формируется новое состояние резольвенты, прямой ход |
| 9 | 4 > 4,  !,  listWithUpper([5], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  1 < 4  Результат: ложь | Откат, возврат к предыдущему состоянию резольвенты, переход к следующему предложению |
| 10 | listWithUpper([4, 5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([4, 5], 4, OutList\_) = listWithUpper([\_|T], Num, OutList) :- listWithUpper(T, Num, OutList), !  Унификация успешна  Подстановка: {T = [5], Num = 4, OutList = OutList\_} | Формируется новое состояние резольвенты, прямой ход |
| 11 | listWithUpper([5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([5], 4, OutList\_) = append([H|T], SList, [H|AddTail]) :- append(T, SList, AddTail), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 12 | … | … | … |
| 13 | listWithUpper([5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([5], 4, OutList\_) = listWithUpper([H|T], Num, [H|AddTail]) :- H > Num, !, listWithUpper(T, Num, AddTail)  Унификация успешна  Подстановка: {H = 5, T = [], Num = 4, [5|AddTail] = OutList\_} | Формируется новое состояние резольвенты, прямой ход |
| 14 | 5 > 4,  !,  listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Унификация:  5 > 4  Результат: истина | Формируется новое состояние резольвенты, прямой ход |
| 15 | !,  listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Встречен системный предикат отсечения | Дальнейшие предложения процедуры рассматриваться не будут. Формируется новое состояние резольвенты, прямой ход |
| 16 | listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([], 4, AddTail) = append([H|T], SList, [H|AddTail]) :- append(T, SList, AddTail), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 17 | … | … | … |
| 18 | listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([], 4, AddTail) = listWithUpper([H|T], Num, [H|AddTail]) :- H > Num, !, listWithUpper(T, Num, AddTail)  Унификация неуспешна (первые аргументы не унифицируемы) | Откат, переход к следующему предложению |
| 19 | listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([], 4, AddTail) = listWithUpper([\_|T], Num, OutList) :- listWithUpper(T, Num, OutList), !  Унификация неуспешна (первые аргументы не унифицируемы) | Откат, переход к следующему предложению |
| 20 | listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([], 4, AddTail) = listWithUpper([], \_, [])  Унификация успешна  Подстановка: {AddTail = [], [5 | AddTail] = OutList\_} | Резольвента пуста.  Вывод: OutList\_ = [5 | []] (или [5])  Откат, переход к следующему предложению |
| 21 | listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([], 4, AddTail) = oddPosList([H|[\_|T]], [H|OutList]) :- oddPosList(T, OutList), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 22-31 | … | … | … |
| 33 | listWithUpper([], 4, AddTail)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([], 4, AddTail) = makeSet(List, List)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Откат с отсечением остаточных предложений процедуры относительно шага 13 |
| 34 | listWithUpper([5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([5], 4, OutList\_) = oddPosList([H|[\_|T]], [H|OutList]) :- oddPosList(T, OutList), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 35-45 | … | … | … |
| 46 | listWithUpper([5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([5], 4, OutList\_) = makeSet(List, List)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Формируется новое состояние резольвенты |
| 47 | !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Встречен системный предикат отсечения | Дальнейшие предложения процедуры рассматриваться не будут. Резольвента пуста, восстановление резольвенты относительно шага 10 |
| 48 | listWithUpper([4, 5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([4, 5], 4, OutList\_) = oddPosList([H|[\_|T]], [H|OutList]) :- oddPosList(T, OutList), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 49-59 | … | … | … |
| 60 | listWithUpper([4, 5], 4, OutList\_),  !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([4, 5], 4, OutList\_) = makeSet(List, List)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Формируется новое состояние резольвенты |
| 61 | !  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | Встречен системный предикат отсечения | Дальнейшие предложения процедуры рассматриваться не будут. Резольвента пуста, восстановление резольвенты относительно шага 5 |
| 62 | listWithUpper([1, 4, 5], 4, OutList\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([1, 4, 5], 4, OutList\_) = oddPosList([H|[\_|T]], [H|OutList]) :- oddPosList(T, OutList), !  Унификация неуспешна (несовпадение функторов) | Откат, переход к следующему предложению |
| 63-73 | … | … | … |
| 74 | listWithUpper([1, 4, 5], 4, OutList\_)  Резольвента непуста, запуск алгоритма унификации для подцели, извлекаемой из стека | listWithUpper([1, 4, 5], 4, OutList\_) = makeSet(List, List)  Унификация неуспешна (несовпадение функторов) | Конец БЗ достигнут. Резольвента пуста, завершение работы системы |