|  |  |
| --- | --- |
| 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 № 18**

|  |  |
| --- | --- |
| **Дисциплина** Функциональное и логическое программирование  **Тема** \_Рекурсия на Prolog\_  **Студент** \_Ильясов И. М.\_  **Группа** \_ИУ7-63Б\_  **Оценка (баллы)** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **Преподаватель** \_Толпинская Н. Б., Строганов Ю. В.\_ |  |

Москва, 2020 г.

**Цель работы** –изучить рекурсивные способы организации программ на Prolog, методы формирования эффективных рекурсивных программ и порядок реализации таких программ.

**Задачи работы**:

* приобрести навыки эффективного декларативного описания предметной области с использованием фактов и правил;
* изучить порядок использования фактов и правил в программе на Prolog, принципы и особенности сопоставления и отождествления термов, на основе механизма унификации. Способ формирования и изменения резольвенты. Порядок формирования ответа.

**Задание лабораторной работы**

**Используя хвостовую рекурсию, разработать программу, позволяющую найти**

1. **n!**,
2. **n-е** 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!

**Вопрос:…**

|  |  |  |  |
| --- | --- | --- | --- |
| № 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 (почему и к чему приводит?) |
| 1… | … | … | Комментарий, вывод… |
| … | … | … | … |

**Текст программы**

predicates

fact(integer, integer).

fib(integer, integer).

clauses

fib(1, Res) :- Res = 1, !.

fib(2, Res) :- Res = 1, !.

fib(N, Res) :-

A = N - 1, B = N - 2,

fib(A, A\_Res), fib(B, B\_Res),

Res = A\_Res + B\_Res.

fact(0, Res) :- Res = 1, !.

fact(N, Res) :-

Next = N - 1,

fact(Next, Next\_Res),

Res = N \* Next\_Res.

goal

write("fib(5) = "),

fib(5, Res);

write("3! = "),

fact(3, Res).

**Примеры работы программы**

1. Поиск n!, где n = 3.

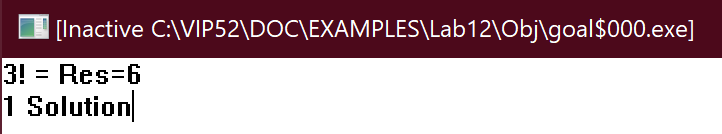


Рисунок 1. Поиск 3!

1. Поиск n-ого числа Фибоначчи, где n = 5.

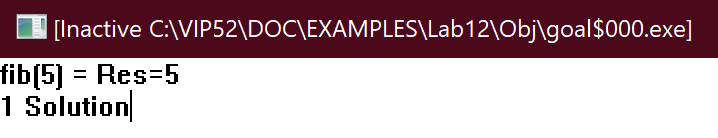


Рисунок 2. Поиск 5 числа Фибоначчи.

**Задание с таблицей**

fib(3, Res) – поиск n-ого числа Фибоначчи, где n = 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № 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 (почему и к чему приводит?) |
| 1 | fib(3, Res) | T1 = fib(3, Res).  T2 = fib(1, Res).  Неудача, разные функторы (3 != 1). | Прямой ход к следующему предложению. |
| 2 | fib(3, Res) | T1 = fib(3, Res).  T2 = fib(2, Res).  Неудача, разные функторы (3 != 2). | Прямой ход к следующему предложению. |
| 3 | fib(3, Res) | T1 = fib(3, Res).  T2 = fib(N, Res).  Успех, подстановка {N = 3, Res = Res}. | Прямой ход к следующему предложению. |
| 4 | A = N - 1, B = N - 2,  fib(A, A\_Res), fib(B, B\_Res), Res = A\_Res + B\_Res. | A = 2 | Прямой ход к следующему предложению. |
| 5 | B = N - 2,  fib(A, A\_Res), fib(B, B\_Res), Res = A\_Res + B\_Res. | B = 1 | Прямой ход к следующему предложению. |
| 6 | fib(A, A\_Res), fib(B, B\_Res), Res = A\_Res + B\_Res. | Подстановка {N = 2, Res = A\_Res}, | Прямой ход к следующему предложению. |
| 7 | Res = 1, fib(B, B\_Res), Res = A\_Res + B\_Res. | Подстановка {Res = 1} | Прямой ход к следующему предложению. |
| 8 | fib(B, B\_Res), Res = A\_Res + B\_Res. | Подстановка {N = 1, Res = B\_Res} | Прямой ход к следующему предложению. |
| 9 | Result = 1, Res = A\_Res + B\_Res. | Подстановка {Res = 1} | Прямой ход к следующему предложению. |
| 10 | Res = A\_Res + B\_Res. | Подстановка {Res = 2} | Прямой ход. |
| 11 |  | **Результат**: Res = 3. | Обратный ход. |

fact(3, Res) – поиск n!, где n = 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № 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 (почему и к чему приводит?) |
| 1 | fact(3, Res) | T1 = fact(3, Res).  T2 = fib(1, Res).  Неудача, разные функторы. | Прямой ход к следующему предложению. |
| 2 | fact(3, Res) | T1 = fact(3, Res).  T2 = fib(2, Res).  Неудача, разные функторы. | Прямой ход к следующему предложению. |
| 3 | fact(3, Res) | T1 = fact(3, Res).  T2 = fib(N, Res).  Неудача, разные функторы. | Прямой ход к следующему предложению. |
| 4 | fact(3, Res) | T1 = fact(3, Res).  T2 = fact(0, Res).  Неудача, разные функторы. | Прямой ход к следующему предложению. |
| 5 | fact(3, Res) | T1 = fact(3, Res).  T2 = fact(N, Res).  Успех, подстановка {N = 3, Res = Res} | Прямой ход к следующему предложению. |
| 6 | Next = N - 1,  fact(Next, Next\_Res),  Res = N \* Next\_Res. | Подстановка {Next = 2} | Прямой ход к следующему предложению. |
| 7 | fact(Next, Next\_Res),  Res = N \* Next\_Res. | Подстановка {N = 2, Res = Next\_Res} | Прямой ход к следующему предложению. |
| 8 | Next = N - 1,  fact(Next, Next\_Res),  Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {Next = 1} | Прямой ход к следующему предложению. |
| 9 | fact(Next, Next\_Res),  Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {N = 1, Res = Next\_Res} | Прямой ход к следующему предложению. |
| 10 | Next = N - 1,  fact(Next, Next\_Res),  Res = N \* Next\_Res,  Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {Next = 1} | Прямой ход к следующему предложению. |
| 11 | fact(Next, Next\_Res),  Res = N \* Next\_Res,  Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {N = 0, Res = Next\_Res} | Прямой ход. |
| 12 | Res = 1,  Res = N \* Next\_Res,  Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {Res = 1} | Прямой ход к следующему предложению. |
| 13 | Res = N \* Next\_Res,  Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {Res = 1} | Прямой ход. |
| 14 | Res = N \* Next\_Res,  Res = N \* Next\_Res. | Подстановка {Res = 2} | Прямой ход. |
| 15 | Res = N \* Next\_Res. | Подстановка {Res = 6} | Прямой ход. |
| 16 |  | **Результат**: Res = 6 | Обратный ход. |

**Ответы на вопросы**

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

Рекурсия – это ссылка на определяемый объект во время его определения. В Prolog рекурсия организуется правилом, в котором есть обращение к тому же правилу. Организовать выход из рекурсии можно при помощи отсечения.

1. **Какое первое состояние резольвенты?**

Если задан простой вопрос, то сначала он попадает в резольвенту.

1. **В каком случае система запускает алгоритм унификации? (Как эту необходимость на формальном уровне распознает система?) Каково назначение использования алгоритма унификации? Каков результат использования алгоритма унификации?**

Если есть что доказывать (цель), то процесс унификации запускается автоматически. Формально: если резольвента не пуста – запускается алгоритм унификации.

Назначение алгоритма унификации заключается в попарном сопоставлении термов и попытке построить для них общий пример.

Унификация может завершаться успехом или тупиковой ситуацией (неудачей).

1. **В каких пределах программы уникальны переменные?**

Переменные уникальны в пределах предложения, т.е. в рамках предложения одно и то же имя принадлежит одной и той же переменной. Исключение – анонимные переменные (обозначаются символом нижнего подчеркивания «\_») – каждая такая переменная является отдельной сущностью и применяется, когда ее значение неважно для данного предложения.

1. **Как применяется подстановка, полученная с помощью алгоритма унификации?**

Применение подстановки {*X1=T1, …, Xn=Tn*} заключается в замене каждого вхождения переменной *Xi* на соответствующий терм *Ti*.

1. **Как меняется резольвента?**

Изменение резольвенты происходит в 2 этапа:

1) из стека выбирается подцель (верхняя, т.к. стек) и для нее выполняется редукция, т.е. замена подцели на тело найденного правила;

2) к полученной конъюнкции целей применяется подстановка (наибольший общий унификатор выбранной цели и заголовка сопоставленного с этой целью правила).

1. **В каких случаях запускается механизм отката?**

Механизм отката запускается в 2 случаях:

1. Если алгоритм попал в тупиковую ситуацию.

2. Если резольвента не пуста и решение найдено, но в базе знание остались не отмеченные предложения.